MALICIOUS USER ATTACKS IN DECENTRALISED COGNITIVE RADIO NETWORKS

by

Arun Sivakumaran

Submitted in partial fulfilment of the requirements for the degree
Master of Engineering (Electronic Engineering)

in the

Department of Electrical, Electronic, and Computer Engineering
Faculty of Engineering, Built Environment, and Information Technology

UNIVERSITY OF PRETORIA

August 2020

SUMMARY

MALICIOUS USER ATTACKS IN DECENTRALISED COGNITIVE RADIO NETWORKS

by

Arun Sivakumaran

Supervisor:	Prof. B.T. Maharaj
Co-supervisor:	Prof. A.S. Alfa
Department:	Electrical, Electronic, and Computer Engineering
University:	University of Pretoria
Degree:	Master of Engineering (Electronic Engineering)
Keywords:	Decentralised cognitive radio networks, malicious user detection, cooperative spectrum sensing, data fusion

Cognitive radio networks (CRNs) have emerged as a solution for the looming spectrum crunch caused by the rapid adoption of wireless devices over the previous decade. This technology enables efficient spectrum utility by dynamically reusing existing spectral bands. A CRN achieves this by requiring its users – called secondary users (SUs) – to measure and opportunistically utilise the band of a legacy broadcaster – called a primary user (PU) – in a process called spectrum sensing. Sensing requires the distribution and fusion of measurements from all SUs, which is facilitated by a variety of architectures and topologies.

CRNs possessing a central computation node are called centralised networks, while CRNs composed of multiple computation nodes are called decentralised networks. While simpler to implement, centralised networks are reliant on the central node – the entire network fails if this node is compromised. In contrast, decentralised networks require more sophisticated protocols to implement, while offering greater robustness to node failure. Relay-based networks, a subset of decentralised networks, distribute the computation over a number of specialised relay nodes – little research exists on spectrum sensing using these networks.

CRNs are vulnerable to unique physical layer attacks targeted at their spectrum sensing functionality. One such attack is the Byzantine attack; these attacks occur when malicious SUs (MUs) alter their sensing reports to achieve some goal (e.g. exploitation of the CRN's resources, reduction of the CRN's sensing performance, etc.). Mitigation strategies for Byzantine attacks vary based on the CRN's network architecture, requiring defence algorithms to be explored for all architectures. Because of the sparse literature regarding relay-based networks, a novel algorithm – suitable for relay-based networks – is proposed in this work. The proposed algorithm performs joint MU detection and secure sensing by large-scale probabilistic inference of a statistical model.

The proposed algorithm's development is separated into the following two parts.

- The first part involves the construction of a probabilistic graphical model representing the likelihood of all possible outcomes in the sensing process of a relay-based network. This is done by discovering the conditional dependencies present between the variables of the model. Various candidate graphical models are explored, and the mathematical description of the chosen graphical model is determined.
- The second part involves the extraction of information from the graphical model to provide utility for sensing. Marginal inference is used to enable this information extraction. Belief propagation is used to infer the developed graphical model efficiently. Sensing is performed by exchanging the intermediate belief propagation computations between the relays of the CRN.

Through a performance evaluation, the proposed algorithm was found to be resistant to probabilistic MU attacks of all frequencies and proportions. The sensing performance was highly sensitive to the placement of the relays and honest SUs, with the performance improving when the number of relays was increased. The transient behaviour of the proposed algorithm was evaluated in terms of its dynamics and computational complexity, with the algorithm's results deemed satisfactory in this regard. Finally, an analysis of the effectiveness of the graphical model's components was conducted, with a few model components accounting for most of the performance, implying that further simplifications to the proposed algorithm are possible.

LIST OF ABBREVIATIONS

CH	Cluster Head
CR	Cognitive Radio
CRN	Cognitive Radio Network
EB	Exabyte
FC	Fusion Centre
ITU	International Telecommunication Union
KL	Kullback-Leibler
MAP	Maximum *a posteriori*
MDR	Majority Decision Rule
MRF	Markov Random Field
MU	Malicious User
PA	Proposed Algorithm
PU	Primary User
RA	Resource Allocation
RF	Radio Frequency
SU	Secondary User
TU	Trusted User
TVWS	Television White Space
UTU	Untrusted User

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

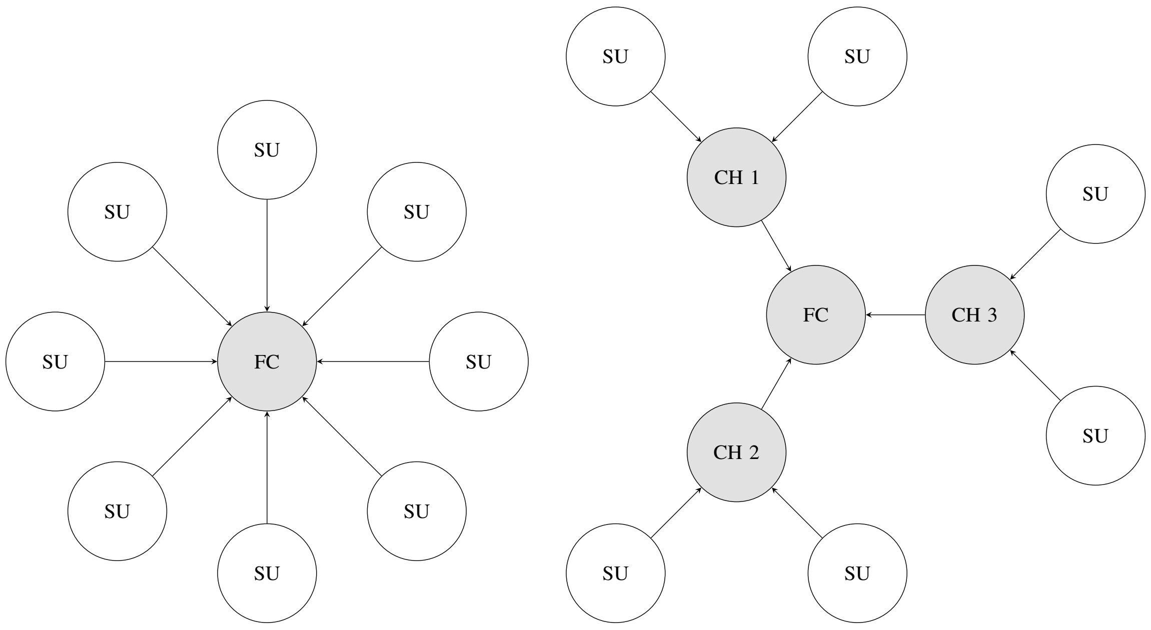
1.1 BACKGROUND

The ubiquity of wireless systems, owing to the integration of the technology into everyday consumer electronics, has led to the increased use of the frequency spectrum. The demand for the use of this spectrum is mismatched by the limited supply. The increase in mobile device traffic is estimated to be from 11.51 EB (exabytes) in 2017 to 77.49 EB in 2022, in response to the presence of high bandwidth services such as video-on-demand, mobile application downloads, and the growth of machine-to-machine devices [1], [2]. Furthermore, the number of mobile subscribers is projected to almost double over the next decade from 10.7 billion in 2020 to 17.1 billion in 2030, leading to concerns of a forthcoming spectrum crunch [2].

A number of solutions have been proposed to combat spectrum congestion. The accommodation of new users is currently facilitated by reducing the bandwidth of existing transmissions or making use of gaps in higher frequency bands. The former method can be achieved by designing more bandwidth-efficient modulation schemes or by increasing the information density of the source content (i.e. data compression). The latter can be achieved by developing sophisticated and expensive radio frequency (RF) equipment to exploit gaps in higher frequency bands. Mobile network capacity can also be improved by optimising the cell sizes [2]. The use of these methods can only marginally relieve spectrum congestion and delay the inevitable spectrum crunch.

Cognitive radio networks (CRNs) offer a long-term solution to spectrum congestion [3], [4]. A CRN attempts to intelligently reuse the frequency spectrum of a licensed legacy user – called a primary user (PU) – by allowing the CRN's own users – called secondary users (SUs) – to broadcast when the PU's spectrum is vacant. As a result, the dynamic nature of CRNs allows different transmissions to share the same portion of the spectrum in the frequency domain. The SUs are responsible for the measurement of the PU's occupancy in a process called spectrum sensing. It is typically beneficial to let the SUs cooperate with one another to increase the diversity of sensing reports in transmission environments. The fusion of the various SU reports to determine the PU channel occupancy is referred to as cooperative spectrum sensing.

The distribution and fusion of SU reports is enabled by a number of different network architectures and topologies. The centralised architecture is used when a single node performs all processing. There are two different arrangements, namely fully centralised and cluster-based topologies. All SU reports are sent directly to a single entity called a fusion centre (FC) for fully centralised topologies. SUs in networks that employ the cluster-based topology pass their reports to a cluster head (CH), which in turn sends the reports to an FC [5], [6]. Figure 1.1 illustrates the centralised topologies, where CH denotes a cluster head.

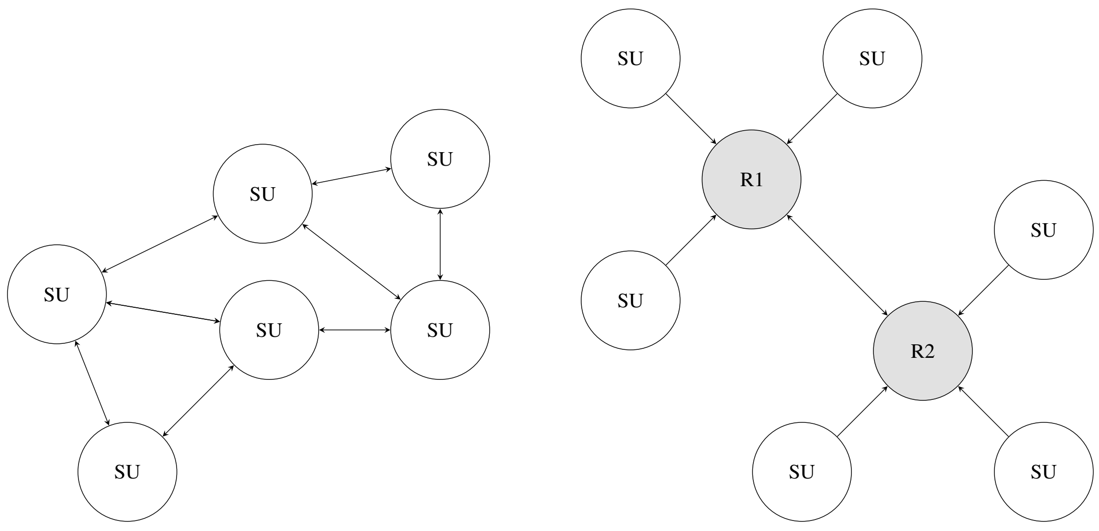


(a) Fully centralised topology.

(b) Cluster-based topology.

Figure 1.1. Centralised CRN topologies.

Networks employing multiple computation nodes use the decentralised architecture. Fully distributed and relay-based topologies exist within decentralised networks. The former is peer-to-peer, requiring all SUs to perform fusion and schedule the transmission of reports, while the latter distributes the computational burden across specialised relay nodes. Figure 1.2 illustrates the decentralised topologies, with R denoting a relay.



(a) Fully distributed topology. **(b)** Relay-based topology.

Figure 1.2. Decentralised CRN topologies.

The effectiveness of a specific architecture and topology is based on the context of the application; it is important to develop and investigate fusion schemes for all architectures, as this improves the variety of applications for CRNs.

CRNs are vulnerable to a number of unique attacks targeted at cooperative sensing, which pose a threat to the success of CRNs as a pioneering technology for spectrum efficiency [7]. For many applications, the honesty of the SUs in a CRN cannot be guaranteed prior to sensing and fusion. The Byzantine attack, where a malicious SU feeds the CRN incorrect information to maximise an objective that benefits a malicious party, is an example of one such attack [7]–[10]. SUs that report incorrect information are referred to as malicious or Byzantine users (MUs). For example, an MU may falsely report the presence of a PU to free the spectrum for its own transmission. Attack strategies can be even more sophisticated if many MUs collude to attain a specified objective [11]. Protection against Byzantine attacks in CRNs is enabled by the use of an appropriate defence algorithm. The choice of the algorithm is determined by the data fusion algorithm used for cooperative sensing, the architecture of the CRN, and the nature of the Byzantine attack employed by the MUs [8].

1.2 RESEARCH OBJECTIVE

In general, defence algorithms for centralised CRNs are simpler to implement in comparison to their decentralised counterparts. Physically realising networks employing centralised architectures requires fixed infrastructure in the form of an FC – the sensing capability of the network is nullified if the FC is compromised in some way. Decentralised networks can bypass infrastructure failure as multiple devices are used to perform fusion. Furthermore, the computational burden is distributed over the nodes performing fusion in a decentralised network, resulting in the need for less sophisticated hardware.

Fully distributed networks offer the possibility of *ad hoc* CRN support for everyday devices. Despite this, a major shortcoming of fully distributed CRNs is the adverse impact of MUs on the network. Since each SU plays a role in sensing, an MU can falsify interim algorithm iterates to change the consensus of the entire network [8], [12]. Relay-based CRNs are decentralised networks that serve as a compromise between centralised and fully distributed *ad hoc* CRNs – certain nodes act as distributed FCs that relay intermediate computations between each other. This topology is more secure than its fully distributed counterpart, as only trusted entities play a role in the fusion of information, bypassing the issue of algorithm iterate falsification. Furthermore, relay-based CRNs are more reliable than centralised CRNs, as computation is not dependent on a single entity. Fusion and data exchange techniques applied to relay-based networks can be extended to fully distributed networks.

Very few publications in the literature describe the formulation of Byzantine defence strategies in relay-based and distributed networks. The objective of this research project is therefore to develop a novel sensing algorithm that minimises the effect of MUs in relay-based CRNs while maintaining sufficient spectrum sensing accuracy.

1.2.1 Research questions

A number of research questions arise from the objectives defined above.

- Can probabilistic inference be used as an efficient cooperative sensing technique in relay-based networks?
- How will the sensing performance of the developed algorithm compare with simpler alternatives? Is it worth the additional overhead incurred?

- How sensitive is the developed algorithm to a change in model parameters (e.g. MU population and proportion, increased number of users, etc.)? Does the algorithm scale well, or is it limited to operating over a small range of parameters?

1.2.2 Research methodology

The aforementioned objectives and research questions are investigated by implementing the proposed algorithm in software. The algorithm uses measurements that are obtained from a simulation of a CRN in a noisy unobstructed environment. The effectiveness of the algorithm is determined by performing a number of tests, including the evaluation of the spectrum sensing performance when varying the proportion and maliciousness of the MUs, and increasing the number of SUs in the network. The dynamics and complexity of the algorithm, which include the computational overhead and time to convergence, are also evaluated. The sensing performance of the algorithm, determined by the overall sensing error rate, is compared against a control algorithm.

1.3 CONTRIBUTION

A novel secure sensing algorithm for relay-based CRNs is proposed in this work. The proposed algorithm uses probabilistic inference, and is formulated with computational efficiency and performance robustness in mind. The contribution is composed of the following parts:

- a comprehensive literature review on CRNs and security techniques,
- a statistical model of the sensing process for a relay-based CRN,
- a decentralised data exchange protocol based on the computation of the statistical model's inference, and
- simulation and analysis of the proposed algorithm's sensing performance.

1.4 RESEARCH OUTPUTS

The work in this dissertation produced a published conference article and a submitted journal article. The conference article concerned the empirical study of the effects of MUs on decentralised networks, and was published and presented in the Spring 2019 IEEE Vehicular Technology Conference (VTC2019-Spring) in Kuala Lumpur, Malaysia. The journal article was submitted to the *Wireless Networks* journal, with its content summarising this dissertation's core contributions. The research outputs are listed below:

[1] A. Sivakumaran, A.S. Alfa, and B.T. Maharaj, "An empirical analysis of the effect of malicious users in decentralised cognitive radio networks," in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, Apr. 2019, pp.1–5.

[2] A. Sivakumaran, A.S. Alfa, and B.T. Maharaj, "Secure spectrum sensing in relay-based cognitive radio networks," *Wireless Networks*, Jan. 2020, in review.

1.5 DISSERTATION OVERVIEW

This dissertation is composed of six chapters, with this introduction being the first chapter. The contents of the following chapters are summarised below.

- Chapter 2 outlines the background and research existing in this field, while introducing and contextualising the main contribution of this work. The concept of the cognitive cycle is introduced, and important aspects relating to the main contribution are highlighted. The advantages and disadvantages of the various network architectures are considered. Techniques involved in signal measurement, quantisation, and data fusion are reviewed. Physical layer security threats to CRNs are then introduced, with a focus on Byzantine attacks. A framework for Byzantine attack model formulation is presented, and some examples of attack models are discussed. Existing defence strategies against Byzantine attacks are then reviewed. Finally, the need for integrated defence and sensing algorithms in decentralised networks is argued, leading to the development of the proposed algorithm in the subsequent sections.
- Chapter 3 consists of the development of the statistical model that forms the backbone of the proposed algorithm. A system model of a relay-based CRN performing cooperative sensing is first developed; the conditional dependencies present between the variables of the model are observed. A probabilistic graphical model is used to encode the conditional dependencies. The graphical model's structure is developed by mathematically defining the relationship between the desired variables of the original system model.
- Chapter 4 describes the proposed algorithm, which is the decentralised implementation of the inference of the statistical model developed in Chapter 3. Efficient inference is achieved by applying belief propagation to the model; this computation is subsequently broken up and partitioned over the network's relays for the decentralised implementation.
- The evaluation and analysis of the proposed algorithm's performance is found in Chapter 5. Firstly, the simulation environment is described, including the propagation model, the layout of the CRN, and the channel noise model used. The performance metrics, which serve to evaluate

functioning of the algorithm, are then defined. This is followed by an explanation of the tests performed, along with a motivation for their use. The simulation methodology employed is then described. Finally, the results are presented; each set of results is accompanied by an analysis and commentary on its meaning.

- Chapter 6 concludes the thesis, with a retrospective discussion of what was achieved by this work. Limitations to the proposed algorithm, along with further avenues for research, are identified.

CHAPTER 2 COGNITIVE RADIO NETWORKS AND SECURITY

2.1 CHAPTER OVERVIEW

In this chapter, cognitive radio (CR) is introduced as a new paradigm for efficient spectrum management in terms of the cognition cycle. This cycle, explored in Section 2.2, is divided into a number of categories, each corresponding to distinct fields of research. The contribution of the work in the following chapters is contextualised in terms of the cognition cycle, and the constituent processes that lead to this contribution are identified. The basic architectures for CRNs are discussed in Section 2.3. Local sensing in the form of signal detection methods is found in Section 2.4, and cooperative spectrum sensing is covered in Section 2.5. Physical layer attacks, specifically Byzantine attacks, are discussed in Section 2.6, while Byzantine defence strategies are the subject of Section 2.7. Finally, the need for secure relay-based sensing algorithms is presented as a potential solution to cooperative sensing-based physical layer vulnerabilities in Section 2.8.

2.2 THE COGNITION CYCLE

CR was envisioned as a replacement for static, licensed radio systems. A fully realised CR, with the ability to function in dynamic environments, would require some form of autonomy on the part of the radio device/network. In Mitola's seminal paper, this autonomy is realised in the form of the cognition cycle [4]. A simplified version of this cycle is shown in Figure 2.1. The cognitive cycle is a high-level concept, with each portion of the cycle representing a vast field of research. The following steps describe the cognitive cycle.

- **Observe –** A CR device interacts with the outside environment by sensing the frequency spectrum. A CR was initially intended to measure a number of different phenomena such as light, humidity, position etc. (for detection of indoor/outdoor environments). Most current research, including this work, assumes that a CR only senses the frequency spectrum; the field of spectrum sensing concerns this operation [13]–[15].

- **Orient –** The orientation phase governs the alternation between the CR's modes of operation, based on the observation of the environment and the current performance of the CR. Reactions to fault detection or a drastic change in service quality are handled in this phase.

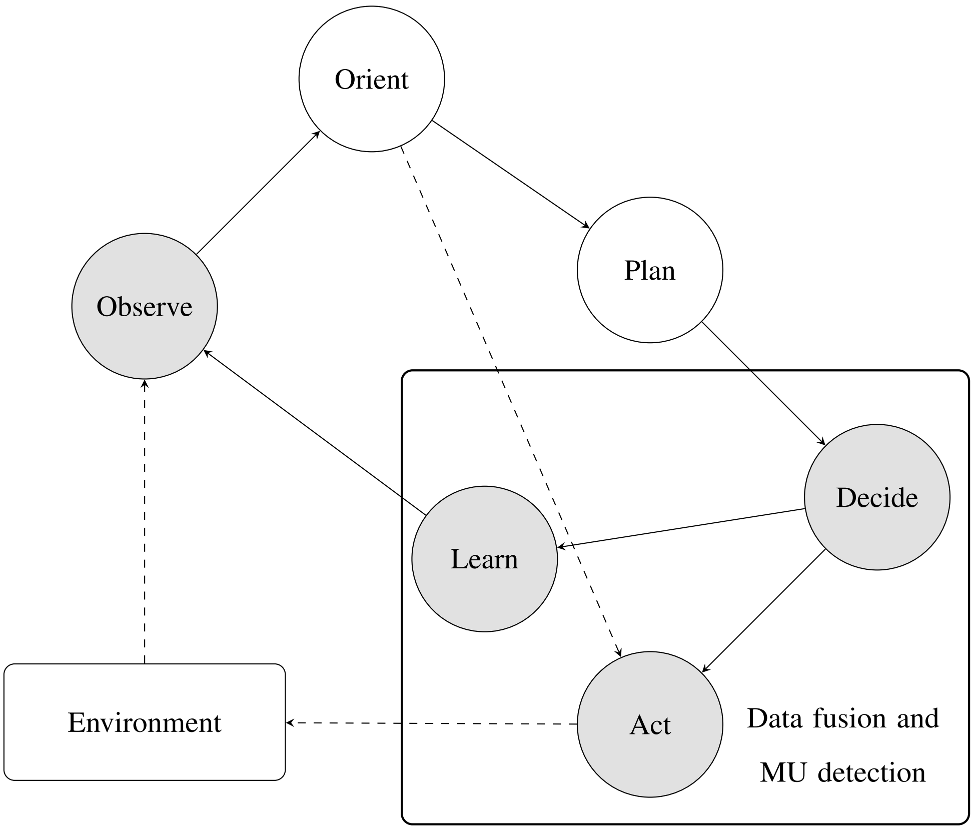


Figure 2.1. Simplified version of Mitola's proposed cycle of cognition for a CRN. Adapted from [4], © 2020 IEEE.

- **Plan –** The planning phase, intended to be fully dynamic in a true CR, involves the optimal allocation of a network's resources, given input from the orientation phase. The field of resource allocation (RA) comprehensively covers the functionality of this phase [16].
- **Decide –** This stage involves the combination of measurements from different devices experiencing unique environmental conditions. Cooperative sensing by data fusion involves the combination of various SU reports to determine the occupancy of a PU in an RF environment to form a decision; it is the primary field of research concerning this phase [17].

- **Act –** Different forms of actuation may take place, depending on the context of the operation. This includes allowing SUs to broadcast because of the vacancy of a PU or identifying and excluding potentially hostile users from the network. It may also involve the process of reallocating resources based on a change in the environment.
- **Learn –** The learning stage involves gaining new knowledge of the environment and the network after a number of iterations of sensing and decision. This knowledge can be used to improve sensing results or provide better network throughput, either of which is possible by tuning system parameters from the preceding phases [17].

The above functionalities can be realised using different network architectures, chosen based on the nature of the CRN's application. A discussion of the various architectures is found in Section 2.3. The research problem defined in Chapter 1, concerning secure cooperative sensing in the presence of MUs, involves consideration of the following topics from the cognition cycle (shaded in Figure 2.1):

- **Local spectrum sensing** involves the raw measurement process that is undergone by an individual SU in the CRN. A number of different techniques can be used to measure the state of the PU signal reliably. The measurement can be quantised to a binary result representing the PU's occupancy or vacancy in a process called signal detection. Detection can be done before or after fusion. The process of spectrum measurement forms part of the shaded **Observe** portion of the cognition cycle in Figure 2.1. Signal detection is part of the **Decision** node. Measurement is considered briefly in Section 2.4.1, and individual SU signal detection is discussed in Section 2.4.2.
- **Cooperative spectrum sensing** entails the fusion of various SU reports to enable an informed decision on the possible state of the PU and CRN. Data fusion schemes typically assume a predetermined SU report format. Soft fusion is applied when the SUs' raw measurements are combined. Hard fusion is applied when the SUs perform signal detection prior to combining. Cooperative sensing and data fusion are highly dependent on the network architecture employed, and form part of the shaded **Decision** node of Figure 2.1. This topic is given attention in Section 2.5, as the fusion of SU reports plays a critical role in the network's sensing performance.
- **MU detection** is the process of selecting SUs in the CRN that are deemed to give dishonest reports. The detection of MUs can be separated or integrated with data fusion, depending on the type of fusion algorithm that is used. Furthermore, the results of MU detection can be used to alter the contributions of potential MUs in a way that is beneficial to the sensing performance. As

a result, MU detection forms part of the **Decision** and **Learn** nodes of Figure 2.1. A discussion of physical layer attacks in CRNs is found in Section 2.6.

Note that actuation (the **Act** node of Figure 2.1) is dependent on the results from data fusion and MU detection. In order to capture the functionality of the **Decision**, **Learn**, and **Act** nodes effectively, it is desirable to formulate integrated fusion algorithms that perform their respective functions simultaneously.

2.3 COGNITIVE RADIO NETWORK ARCHITECTURES

A number of network architectures and arrangements can be used for spectrum sensing and data fusion [17]. Each architecture has its own advantages and disadvantages, and is determined by the needs of the sensing application. As mentioned in Section 1.1, the two major categories of network architecture are centralised and decentralised networks.

2.3.1 Centralised networks

The two topologies for centralised networks – fully centralised and cluster-based – are shown in Figure 1.1. Fusion algorithms can be implemented on centralised networks with almost no limitations, as all processing is performed by the FC. Fully centralised networks benefit from simple network management and low bandwidth requirements, since communication is limited to links between the SUs and the sole FC [18]. However, because of the inflexible structure of a fully centralised network, its size and the distance covered by the network are limited [8], [17]. Cluster-based networks offer greater scalability, allowing groups of SUs to send their reports to a local CH that sends the reports to the FC. The CHs may perform pre-processing on the reports to reduce the communication bandwidth, which will result in increased energy consumption. Employing this topology requires some form of cluster management, resulting in increased bandwidth use. Prior knowledge of the location and spread of the SUs may be required for cluster formation. Statistical approaches can be used if location information is unavailable, but additional overhead is incurred with these methods [6].

2.3.2 Decentralised networks

The reliance of centralised networks on a single computation node is a major drawback – fusion cannot take place if the FC is compromised. To this end, decentralised networks sacrifice the convenience of a centralised computation node in favour of improved network robustness [19]. Two possible topologies for decentralised networks – fully distributed and relay-based – are shown in Figure 1.2. Information can be transferred over a number of routes for fully distributed networks, as all SUs are responsible for fusion. In contrast, only a single path for information transfer exists for relay-based networks (across the relays themselves). The differences between these topologies result in a number of application-specific advantages for both networks.

Fully distributed networks have the advantage of requiring no specialised infrastructure, with the computation distributed over each SU in the CRN; this reduces the hardware burden on the network significantly. Fully *ad hoc* CRNs are only possible with the distributed topology; an arbitrary group of users with sensing capabilities can exchange information and form a functioning CRN [19]. A major disadvantage of these networks is the significant network management required and bandwidth used when multiple SUs communicate with one another. Furthermore, *ad hoc* distributed networks are extremely vulnerable to physical layer attacks, as there is no guarantee that an SU responsible for data fusion is not malicious [8].

Relay-based networks serve as a compromise between fully distributed and centralised networks. Instead of allowing all SUs to perform fusion, a number of specialised nodes relay neighbouring reports and interim fusion results among one another. The relays can be SUs themselves or external entities. The relay nodes are capable of viewing all the SU reports simultaneously, much like an FC in a centralised network. Any computations are distributed across each of the relay nodes, similar to a fully distributed network. The relaying of information from one node to the next is referred to as a hop [20]. Networks with many relays require more hops for information transfer, resulting in large delays across the nodes. Large networks do, however, reduce individual computational burden and possess greater scalability. Energy consumption increases with every relay added to the network, leading to an upper bound on the number of relays to be deployed in a network [17]. If one of the relays is compromised, the network can recover by establishing a connection to the next relay [21], [22]. Relay-based networks are more secure than their distributed counterparts if only trusted entities are used to form the relay network.

2.4 LOCAL SPECTRUM SENSING

Sensing takes place in the transmission space, which is composed of many dimensions, including space, time, and frequency [13]. In general, sensing is equivalent to finding regions in the transmission space that are not occupied by the PU. The simplest way to exploit a gap in the transmission space is to transmit in the PU's frequency band when the PU is not broadcasting. The individual estimation of the PU state is composed of signal measurement and – if hard fusion is used – signal detection.

2.4.1 Primary user signal measurement

It is assumed that the output of PU measurement is a continuous quantity E, with a large value of E implying the occupancy of the PU; the converse is true for a small value of E. If hard fusion is used after individual sensing, a local decision is made by quantising E. The following techniques can be used to obtain the measurement E:

- **Energy detection –** This is the simplest, least demanding signal measurement technique [23]. For this reason, it can be applied to many situations. Furthermore, the performance of this sensing technique can be modelled and evaluated effectively [24], [25]. It is also the most viable strategy to implement, as it requires no *a priori* information on the PU. Its simplicity comes at the cost of low detection accuracy. Mathematically, energy detection can be summarised as

$$E = \sum_{t=1}^{T} |y(t)|^2, \tag{2.1}$$

  with T representing the observation period, t the time index, and $y(t)$ the received signal.

- **Waveform-based sensing –** Some signals have known regularities in their structure, such as cyclic prefixes or pilot signals. Signals can be detected by observing these patterns. Correlating the received signal pattern $p_r(t)$, a subset of $y(t)$, with a template pattern $p_t(t)$ at the receiver results in [26]

$$E = p_r(t) \star p_t(t) + n(t) \star p_t(t), \tag{2.2}$$

  where $\star$ refers to the cross-correlation operation. The term $n(t) \star p_t(t)$ represents the correlation of received noise $n(t)$ with the template pattern $p_t(t)$. The value of E is large when a signal feature is detected, and is small otherwise. While this technique is more accurate than energy detection, it requires *a priori* knowledge of the PU signal.

- **Cyclostationary sensing –** Many signals express the phenomenon of cyclostationarity. A waveform exhibits this behaviour when the statistics describing the waveform vary over time, but repeat periodically [27]. Knowledge of this can be used to identify a PU signal; this comes at the cost of increased computational overhead and delays, and is limited to the class of signals that exhibit this property.

- **Matched filter methods –** Matched filters are, by definition, optimally designed to reconstruct the true signal $s(t)$ from a received signal $y(t)$ that travels through a channel $h(t)$ [28]. This high level of accuracy comes at the requirement of a large amount of *a priori* information about the PU signal, including the demodulation and carrier synchronisation protocols [29]. Matched filter receivers are specific to a single class of PU signals, and multiple receivers must be present at each sensor node for greater configurability. This limits the flexibility of matched filter-based receivers for spectrum sensing.

Regardless of the type of method used, PU measurement results in the continuous estimate E that describes the perceived occupancy status of the PU signal. This quantity is sent to neighbouring processing units for cooperative sensing and may be quantised to ease computational and bandwidth requirements.

2.4.2 Quantisation of the PU measurement

Most SU quantisation techniques use some form of hypothesis testing to make a decision on the spectrum occupancy of a PU. The generic form of a hypothesis test requires a likelihood ratio to be calculated [8], [10]:

$$F = \frac{p(E|\mathcal{H}_1)}{p(E|\mathcal{H}_0)}, \tag{2.3}$$

where F is the likelihood ratio. The null hypothesis $\mathcal{H}_0$ represents the case of a PU being absent, while the alternative hypothesis $\mathcal{H}_1$ implies that the channel is in use. The calculation of $p(E|\mathcal{H}_0)$ and $p(E|\mathcal{H}_1)$ is achieved by modelling the local sensing process. The hypothesis is most often determined using a decision threshold λ, resulting in a Neyman-Pearson test:

$$u = \begin{cases} \mathcal{H}_0, & \text{if } F < \lambda \\ \mathcal{H}_1, & \text{if } F \geq \lambda \end{cases}. \tag{2.4}$$

A sequential probability ratio test can be used to attain greater detection accuracy:

$$u = \begin{cases} \mathcal{H}_0, & \text{if } F \leq \eta_0 \\ \mathcal{H}_1, & \text{if } F \geq \eta_1 \end{cases}, \tag{2.5}$$

where the pair (η_0, η_1) denotes the respective upper and lower decision thresholds for either hypothesis. More sensing rounds are required if the test statistic lies in the range $\eta_0 < F < \eta_1$. Many iterations of (2.5) may be required to yield conclusive results. This may take a long time in unfavourable environmental conditions.

When soft fusion is used for cooperative sensing, the SUs send their raw measurements to the FC. A soft fusion rule is used to make a final decision directly. This increases communication overhead, as floating-point values are required for transmission between the FC-SU links of the CRN [30].

2.5 COOPERATIVE SPECTRUM SENSING

One of the principal challenges in the design of CRNs is the observation of a PU in a non-ideal communication environment. Channel effects such as path loss or shadow fading cause degradation of the communication links in the CRN, worsening the quality of the CRN's observations [11], [31]. Cooperative sensing by data fusion is employed to improve the accuracy of the individual PU occupancy measurements by exploiting the diversity of SUs experiencing unique environmental conditions. The CRN considers the results from all the SUs and makes a decision accordingly. Cooperative sensing also includes the protocol and scheduling of the information transfer between the SUs.

2.5.1 Non-ideal effects

Numerous non-ideal effects degrade the quality of SU reports, among others:

- channel effects due to environmental conditions (e.g. shadow fading and path loss),
- the different sensing characteristics of each SU (caused by variation in hardware and detection algorithms among SUs), and
- the honesty of all the SUs in the CRN.

Path loss and shadow fading present in the communication channel are common reasons for non-ideal sensing performance in CRNs [8], [31], [32]. Path loss is caused by the decay of signal intensity due to propagation [11]. The decay of the PU due to path loss can cause an SU to falsely assume that the channel is vacant. The nature of the landscape over which the CRN exists causes the received PU signal power to differ at each SU. Obstacles located in the landscape scatter the PU signal such that localised regions experience a similar channel gain, resulting in shadow fading. Shadow fading distorts the PU power measurements taken by the SUs, which in turn reduces the performance of cooperative sensing in a CRN.

Diversity in SU parameters and channel conditions is typically modelled using detection error probabilities. False alarms, denoted by p_{fa}, represent the event of measuring an occupied PU channel when it is actually vacant. Missed detections, given by p_{md}, occur when an occupied channel is incorrectly assumed to be vacant [11]. False alarms and missed detections only occur under a certain

PU hypothesis ($\mathcal{H}_0$ and $\mathcal{H}_1$ respectively). The quantities p_{fa} and p_{md} capture the perceived channel conditions and hardware reliability of each SU. The total probability of error is the weighted sum of the false alarm and missed detection probabilities (weighted by the frequency of either PU state). The probability that an SU makes a correct decision can be determined by the complements of p_{fa} and p_{md}. These results can be generalised to obtain the sensing performance of the entire CRN.

2.5.2 Data fusion

Data fusion involves combining different pieces of information to form a more accurate global estimation. Most data fusion algorithms for cooperative sensing can be divided into the following classes:

- K-out-of-N decision rule,
- consensus algorithm,
- diffusion algorithm, and
- belief propagation algorithm.

The K-out-of-N rule is typically applied to centralised networks, while the consensus and diffusion algorithms are applied to decentralised networks. Belief propagation algorithms, which are based on graphical inference, can be applied to both centralised and decentralised networks. Algorithms from most classes can be sufficiently altered to be compatible with the hard or soft fusion protocols.

2.5.2.1 K-out-of-N rule

A popular choice of centralised fusion rule, owing to its ease of implementation, is the K-out-of-N rule, which requires at least K SUs (from a total of N SUs) to decide in favour of a state for that state recognised as the consensus [8], [10]. Each SU, indexed by n, makes a decision u_n on the channel occupancy before sending its report for fusion. The fusion rule is expressed as

$$\hat{\mathcal{H}} = \begin{cases} \mathcal{H}_0, & \text{if } \sum_{n=1}^{N} u_n < K \\ \mathcal{H}_1, & \text{if } \sum_{n=1}^{N} u_n \geq K, \end{cases} \tag{2.6}$$

where $\hat{\mathcal{H}}$ is the global network PU state estimation. The AND and OR fusion rules represent the extreme cases of (2.6), where $K = N$ and $K = 1$ respectively. The simplicity of the basic fusion rule of (2.6) leads to lack of robustness against non-ideal channel effects and malicious CRN attacks. A soft fusion rule version of the K-of-N rule exists, where the contribution of the n^{th} SU is optimally weighted based on the error probabilities of the SU [33].

2.5.2.2 Consensus algorithm

Distributed networks, which lack the advantage of a central FC, can use the information provided by their neighbours to iteratively reach consensus [12], [34]. The consensus algorithm is given by

$$u_n^{(k+1)} = u_n^k + \epsilon \sum_{j \in \mathcal{N}_n} (u_j^k - u_n^k), \tag{2.7}$$

where k is the algorithm's current iteration index, ϵ is a weighting factor based on the topology of the network, and $\mathcal{N}_n$ is the collection of neighbouring nodes at SU n [12]. Equation (2.7) can be used for both hard and soft fusion, with u_n being discrete or continuous for each particular case. If hard fusion is used, the consensus algorithm becomes equivalent to the K-out-of-N rule in (2.6) with $K = \frac{N}{2}$. If soft fusion is used, a global threshold is required to determine the final decision $\hat{\mathcal{H}}$. The final decision of the decentralised network is the value of u_n^k when all the SU reports converge on the same result. The convergence rate of global consensus is linked to the delay between sensing and decision-making in the network.

2.5.2.3 Diffusion algorithm

The diffusion algorithm is the decentralised solution to an optimisation problem set up to minimise the difference between the raw individual SU measurements E_n and the average received power over the network P [35]–[37]:

$$\begin{aligned} J(\mathbf{E}) &= \sum_{n=1}^{N} \mathbb{E}\left[|E_n|^2 - P\right], \\ \mathbf{E}_o &= \underset{\mathbf{E}}{\operatorname{argmin}} J(\mathbf{E}), \end{aligned} \tag{2.8}$$

where $\mathbf{E} = [E_1, \cdots, E_n, \cdots, E_N]$, $\mathbf{E}_o$ is the optimal set of measurements, and $\mathbb{E}$ is the expectation operator. The goal of the diffusion algorithm is to alter the measurements E_n such that the mean square estimation error J is a minimum. Furthermore, the solution of (2.8) must be obtained without the use of an FC. The diffusion algorithm exclusively employs soft fusion, and is intended for use in decentralised networks.

2.5.2.4 Belief propagation algorithm

The sensing process, composed of unknown variables such as the PU state and SU measurements, can be represented as a joint distribution, where every combination of variable states is associated with a likelihood of occurrence. These likelihoods can be normalised such that they all sum to unity to form a probability distribution. The most likely states of the unknown variables correspond to where the probability density of the distribution is largest. Determining this set of optimal states is aided by inferring the joint distribution. Belief propagation offers an efficient way to perform this inference, and requires the joint distribution to be converted to a statistical graph. Inference is achieved by passing messages along the edges of the graph. Performing cooperative sensing in this manner allows multiple dependent variables to be estimated simultaneously, thus making probabilistic inference a suitable candidate for integrated fusion and defence algorithms. The message-passing nature of belief propagation allows it to be readily used for decentralised networks, as the interim algorithm computations become the information exchanged between the computation nodes.

2.6 PHYSICAL LAYER ATTACKS

Because of the unique nature of CRNs, they are sensitive to a number of physical layer attacks [38]. These attacks can be carried out at any phase of the cognitive cycle in Figure 2.1. Compromising any one of these phases can lead to poor performance and quality of service for the users of the network. Physical layer attacks are conducted by entities that can vary from external agents to malicious SUs in the CRN. These attacking entities are typically motivated by one of the following two reasons [7].

- **Greed** – The malicious entity wishes to broadcast in a vacant spectrum hole left behind by a missed detection error on the part of the CRN.
- **Degradation** – The malicious entity wishes to compromise the CRN; it attempts to do this by degrading the CRN's performance as much as possible.

The following physical layer attacks directly affect cooperative spectrum sensing:

- **Primary user emulation attacks** – Emulation attacks occur when a malicious entity broadcasts in the band of the PU, tricking the SUs into incorrectly detecting the PU [7], [8]. If the PU broadcasts using multiple channels, the malicious entity will transmit in one of those bands, allowing any users linked with the entity to broadcast in the other unused bands. A typical countermeasure to emulation attacks is to analyse the signal characteristics and dynamics of any incoming signal in the PU's bands; a signal that does not conform to the standard characteristics of the PU signal will be ignored by the SUs [39]. For example, if it assumed that the PU is a television white space (TVWS) broadcaster, an *ad hoc* malicious entity is unlikely to transmit a signal with a power as high as the TVWS PU. As a result, any significant drops in the PU's received power can be treated as a potential emulation attack [7].

- **Byzantine attacks** – Byzantine attacks occur when an MU alters its sensing data in order to change the spectrum-sensing consensus of the CRN [7], [8], [11]. The attackers can follow an attack strategy either to render the network ineffective or to force a missed detection error and broadcast in the resulting spectrum hole. A large variety of methods exist to detect and counteract Byzantine attacks, all of them involving the analysis of an SU's sensing results in the context of the CRN's performance.

The core contribution of this work concerns attack prevention in relay-based decentralised networks. Byzantine attacks are intrinsically more dangerous to relay-based networks than to their centralised counterparts, while primary user emulation attacks affect all CRN topologies equally. Since Byzantine attacks threaten the viability of decentralised networks specifically, they will be solely considered for the remainder of this work. The contribution of this work provides a base defence strategy for relay-based networks upon which improvements and alterations can be made, one of which could include the prevention of primary user emulation attacks. Byzantine attack strategies used by MUs are discussed further in Section 2.6.1, and the prevention and correction of Byzantine attacks is the focus of Section 2.7.

2.6.1 Byzantine attack strategies

Formulating pragmatic CRN defence algorithms requires the development of suitable attack models. These strategies must be able to capture the complex behaviour of an MU looking to accomplish one of the two goals: satisfying greed or accomplishing degradation. Attackers employing the former method are irrational, while the latter are more intelligent. Exploitative attackers may behave like ordinary SUs initially, attempting to gain the rest of the CRN's trust. The following parameters can be used to determine the type of MU attack to be used on the CRN [8]:

- the network architecture of the CRN,
- the ratio of MUs relative to the total network population,
- the statistical dependence of the MU attacks with respect to other SUs, and
- the determinism, periodicity, and stationarity of attacks.

2.6.1.1 Attacking based on the network architecture

In centralised and relay-based decentralised CRNs, MUs attempt to degrade the performance by sending incorrect reports to the FC or the relays. They are, in general, unable to access the defence algorithm used by the FC/relays. This fact can be used to detect and eliminate MUs from the sensing process. Fully distributed networks, which require all SUs to participate in the fusion process, cannot rely on withholding information on the defence algorithm. Since the MUs of a distributed network participate in decision fusion, they can also inject incorrect values during the fusion process [12]. This makes MU attacks in *ad hoc* CRNs a critical problem.

2.6.1.2 Population of MUs in attacks

The magnitude of an attack is governed by the number of MUs used in a CRN. The effectiveness of a population-based attack can be appraised by computing the Kullback-Leibler (KL) divergence between the probability distributions for the data received under either PU hypothesis: $\mathcal{H}_0$ or $\mathcal{H}_1$ [11]. A large KL divergence implies that the two distributions express vastly different scenarios. Conversely, a KL divergence of zero would mean that both distributions represent identical situations. The KL divergence is defined as [11]

$$D(a||b) = \sum_x \log \frac{a(x)}{b(x)}, \tag{2.9}$$

where $a(x)$ and $b(x)$ are the distributions for which the divergence is evaluated. Assuming hard fusion is employed, the following representations are used for a and b:

$$a(x) = p(u|\mathcal{H}_1), \quad b(x) = p(u|\mathcal{H}_0). \tag{2.10}$$

The critical proportion of MUs that reduces (2.9) to zero represents the “blind point” of the CRN. Decisions made by the CRN past this point are reduced to random guesses, resulting in a worst case performance scenario. CRN attacks can thus be classified by the number of attackers; a smaller MU population would be easier to detect than a large one, as less false information is fed to the network. A defence algorithm must maximise its blind point to ensure that large MU populations cannot degrade the network performance significantly.

2.6.1.3 Statistical dependence of attacks

Attacks that are statistically independent of any other variables in the CRN are the simplest to model. The dependence of each type of attack is illustrated in Figure 2.2. Fully independent attacks, which could be used for vandalism, do not depend on any state in the CRN. These types of attacks occur in the form of always on ($u_n = 1$), always off ($u_n = 0$), or random attacks [30]. This family of attacks is the simplest to defend against, as the sensing results from each MU will deviate from those of the honest SUs. Network-independent attacks include a class of more intelligent MUs that attack based on the state of the PU they perceive. For example, an MU could intentionally report on the absence of a PU in a channel when the PU is actually present. An MU could also deliberately try to increase the global sensing error probability.

Dependent attacks, which are the hardest to detect, rely on all the factors given in Figure 2.2. These attacks are cooperative in nature, with each MU considering its neighbours' results before sending its own responses. The complex attack patterns that can be formulated by these MUs require much more sophisticated detection algorithms [8], [40]. MUs participating in collusion attacks are rational, and will disrupt the network in subtle ways to maximise long-term damage.

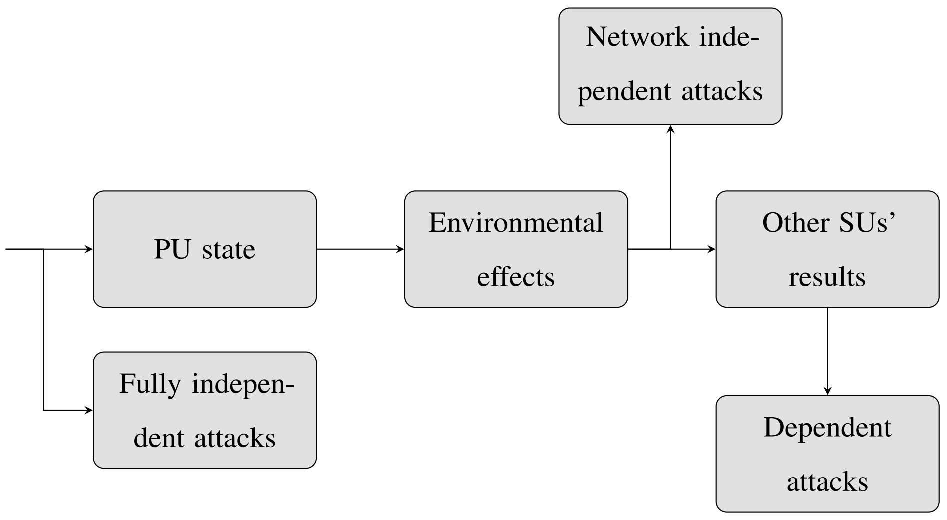


Figure 2.2. The statistical dependence of each type of attack.

2.6.1.4 Causality of MU attacks

The temporal pattern of MU attacks is an important part of their attack strategy. The statistical dependence of attacks discussed in Section 2.6.1.3 expresses the conditions required prior to the launching of an attack, while this section refers to how the attacks are carried out. A deterministic attack strategy would require the MUs to send reports obtained from a predetermined algorithm. These types of MUs could adjust their parameters to optimise an objective function (e.g. KL divergence, throughput). Attackers that consistently send false reports can be detected by the use of reputation-based defence schemes. To combat this, MUs can attack based on an assigned probability to reduce the likelihood of detection by the rest of the CRN [10]. Attacking based on probabilities still presents patterns to the CRN over long periods of time owing to the statistical stationarity of the random process. MUs employing a non-stationary probabilistic attack model would require the most complex detection algorithms, as the CRN would require a longer temporal window to commence detection.

2.6.1.5 Types of attack strategies

Many Byzantine attack models can be derived from the framework described above [8]. A combination of features from each of the four criteria (i.e. network architecture, population, statistical dependence, and causality) results in a given class of attack models. To demonstrate the variety in attack strategies, two attack models – each polarised in terms of complexity – are discussed.

- **Strategy 1** – Consider a group of MUs that attack a fully centralised CRN independently. The number of attackers is large in comparison to the total number of SUs in the CRN. The attackers constantly send randomly generated reports to the FC.
- **Strategy 2** – Now consider another group of MUs that attack a fully distributed network. Owing to the distributed nature of computation in the network, these users are fed reports from neighbouring honest users. The attackers use their individual sensing results, along with their neighbouring users' results, to alter their sensing reports. To reduce visibility, only a small of number of these malicious agents are deployed across the network.

Strategy 1 is a centralised, densely populated, independent, and probabilistic attack, similar to a denial-of-service attack in conventional networks. It intends to disrupt the functioning of the CRN by feeding numerous completely incorrect reports to the FC. In contrast, strategy 2 is a distributed, sparsely populated, dependent, and deterministic attack; the MUs alternate intelligently between malicious and honest modes to avoid detection by neighbouring users. Note that the variety in attack strategies causes the defence strategies to differ similarly; the attack model is more sophisticated in strategy 2, and the defence algorithm used to mitigate strategy 2 will be more complicated than the one used to mitigate strategy 1. Different strategies used to combat Byzantine attacks are discussed in Section 2.7.

2.7 BYZANTINE ATTACK MITIGATION

The detection methods employed by CRNs can be categorised according to

- the nature of the attack strategy,
- the network architecture of the CRN,

- the homogeneity of the CRN's SUs, and
- the approach used to mitigate the attacks.

2.7.1 Homogeneous and heterogeneous CRNs

Network homogeneity is a parameter, in addition to the network architecture, that defines the CRN. Homogeneous networks are a class of CRNs whose constituent SUs all contain identical sensing characteristics [8]. SUs in a homogeneous network experience identical channel conditions and possess the same hardware characteristics. The assumption of network homogeneity simplifies the formulation of the defence algorithms at the expense of reducing their effectiveness. Algorithms assuming homogeneous sensing include outlier, utility, and probability distribution-based methods.

In contrast, heterogeneous networks, which contain SUs with varying sensing characteristics, require more advanced defence methods. The main cause of heterogeneous conditions – environmental effects – can be directly modelled to account for the variation in SU parameters. Byzantine attack mitigation can then be performed by using the aforementioned homogeneous methods. The possibility of directly modelling each cause of variation is limited, as phenomena that are unknown at the time of modelling cannot be captured. A more robust approach to accounting for heterogeneity is to identify differences and similarities in SU reports by directly analysing the emitted data while being agnostic to the true cause of these differences. This is typically done by inferring the parameters that maximise the likelihood of some template statistical model. A hierarchy of the different classes of detection algorithms is shown in Figure 2.3.

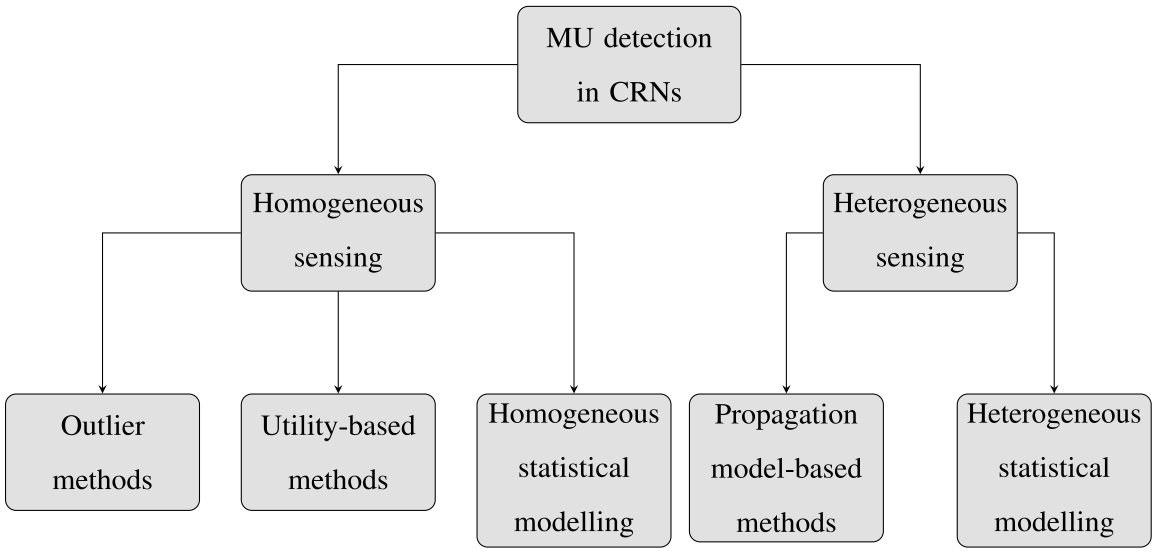

MU detection in CRNs
Homogeneous sensing
Heterogeneous sensing
Outlier methods
Utility-based methods
Homogeneous statistical modelling
Propagation model-based methods
Heterogeneous statistical modelling

Figure 2.3. Hierarchy of the various categories of MU detection. Adapted from [8], © IEEE 2020.

2.7.2 Outlier methods

Outlier detection methods that assume an SU's results are quantised immediately after measurement (i.e. hard fusion) are the simplest to implement. The CRN can compose a reliability metric by tracking the deviation of an SU's prediction from the network's consensus. Statistical tests, such as the Kruskal-Wallis test, can be used to determine if a set of samples originated from the same distribution [41]. In the context of sensing, an SU measurement originates from one of two distributions due to PU vacancy ($\mathcal{H}_0$) or occupancy ($\mathcal{H}_1$), which should be mutually exclusive. The Conover-Iman method is used to determine whether the difference between two samples is significant; if the Kruskal-Wallis test signals that a group of samples originated from conflicting distributions, the Conover-Iman method can then be used to determine which report was maliciously altered.

If soft fusion is used, outliers can be determined by extracting the statistical moments of the reports [9]. The magnitude of an outlier can be measured by

$$o_n = \frac{E_n - \mu}{\sigma}, \tag{2.11}$$

where o_n is the outlier test statistic, and E_n is the raw measurement from SU n. The mean and variance of the PU signal's power sensed by the neighbouring SUs are given by μ and σ respectively. The inherent skewness of the distribution of received energy readings implies that more robust statistical metrics are required for a pragmatic implementation. Furthermore, coalitions can be formed to improve the outlier estimations such that only SUs with correlated information (over a time window) are considered for the calculation of μ and σ in (2.11). An extension of this protocol involves the association of each SU with a penalty score. The reputation of an SU steadily decreases when its reports conflict with those of its neighbours. Cooperative diversity is exploited when employing outlier detection since the readings of one SU are compared with all other SUs, each of which has a unique perspective of the CRN.

Reputation metrics, which track the behaviour of an SU over a fixed time window, can be used by the CRN to quantify the perceived trust associated with an SU. The deviations from a global decision can be tracked over T time steps, with an SU considered malicious as soon as it has deviated more than η times [11]. Outlier detection schemes have been reasonably successful when tested in models that use a small MU population employing independent attacks. These types of methods tend to fail if the MU employs a more sophisticated attack model [11].

2.7.3 Utility-based methods

Utility-based detection methods consider cooperative sensing and MU attacks as a complex interaction between two parties. Honest SUs (and the FC for centralised networks) attempt to maximise the utility of the entire CRN by successfully predicting the occupancy of a PU. Rational MUs attempt to disrupt the CRN by falsifying their measurements. In order to remain undetected, the MUs must still report some correct measurements. Thus, the utility of an MU can be split into two parts: an MU's usefulness to the CRN, and the MU's own effectiveness in exploiting the CRN's resources [42]. A CRN can discourage an MU from falsifying reports by making cooperation a greater incentive (to the MU) than exploitation. Mechanism design theory, also referred to as reverse game theory, has been applied to dissuade and detect MUs in a CRN. The FC is given the ability to sense the spectrum, and an optimisation problem based on minimising the FC's sensing probability while maximising the cost associated with an MU falsifying attacks is developed [30].

2.7.4 Homogeneous statistical modelling

Detection of MUs via statistical modelling requires a distribution that models the data emitted by an SU. The statistical behaviour of each SU can be inferred during cooperative spectrum sensing [43]. The parameters that describe the probability distribution of incoming reports are refined with each sensing round.

The SUs that falsify reports may contain statistical characteristics that are very different from those of their honest counterparts; this can be used to identify the MUs. The statistics of the alternation of an SU's reports (i.e. an SU choosing between PU occupancy and vacancy) can be modelled as a Markov process [44]. The transition between the two measurement states $u_n = 1$ and $u_n = 0$ is represented by probabilities. The deviation between each SU's transition probabilities provides a metric for SU maliciousness. This is because honest SUs will tend to choose decisions in a pattern that matches the true PU occupancy. In contrast, MUs may occasionally feed the FC incorrect results to free the spectrum for their own transmission, resulting in a different report sequence.

Note that the above methods rely on the assumption of homogeneity; the models assume that all honest SUs behave similarly. An extension of these methods for heterogeneous networks is discussed in Section 2.7.6.

2.7.5 Propagation model-based methods

Individual SU reports are conditionally independent of one another if the only cause of CRN heterogeneity is the channel propagation model. Adjusting each SU's error probabilities to reflect the effect of the environment on the sensing results accounts for the variation, provided that the environment is the only cause of this variation. Methods have been developed to improve sensing robustness by modelling the shadow fading present in the CRN environment [31]. A distribution that captures the pattern of the incoming arrivals can be approximated on condition that the characteristics of the shadow fading field are known. These distributions can then be used to track sensing reports that are outliers. The formation of coalitions to aid in MU detection is possible by grouping SUs experiencing similar environmental conditions [32]. Optimal coalitions that jointly minimise power consumption due to sensing and maximise the probability of detection can also be formed.

2.7.6 Heterogeneous statistical modelling

In many cases, channel effects are not the sole contributors of misleading measurements. Determining all the true factors and subsequently modelling them are not desirable, as doing this would lead to a system model that is too specialised. A sufficiently generalisable detection technique can be constructed by letting the measurements fit the model. This is typically achieved by attempting to maximise the likelihood of the received measurements. The model parameters that yield the maximum likelihood or maximum *a posteriori* (MAP) probabilities are determined.

The process of an SU sensing the spectrum, falsifying a measurement, and sending a message to the FC can be broken down into a set of functions that alters the state of a set of desired variables [10]. The interaction between these functions can be represented using a graphical model, and belief propagation can be used to infer the desired variables. The application of this technique results in probability distributions of the true hypothesis and the perceived maliciousness of each SU in the CRN. Techniques used to detect MU collusion in centralised networks can also be derived using belief propagation [40]. Correlations in SU reports are used to cluster similar SUs together. The structure of the graphical model is determined by these clusters. Yet again, belief propagation is applied to determine the distributions over the desired variables. Detection of correlated reports without the need for prior assumptions is an important step towards developing a universal MU detection algorithm.

Expectation maximisation can be used to autonomously select the model parameters that maximise a specified likelihood function [45]. Expectation maximisation has been used in a variety of different fields, most notably for clustering datasets in machine learning. The algorithm can be used to partition SUs based on classes representing different behaviours [46]. These characteristics are derived from the heterogeneity of the CRN, where groups of SUs can be formed based on their sensing measurements.

2.7.7 Defence strategies in decentralised networks

The advanced defence strategies mentioned in this section are only suitable for centralised implementations. Most existing decentralised attack mitigation algorithms are reputation-based, as sophisticated schemes require additional information transfer protocols to be implemented, further complicating the formulation of these strategies. As a result, there is a need for the development of robust decentralised defence algorithms. In decentralised networks, minimal research has been done on the implementation of defence algorithms in relay-based networks. As a result, a novel relay-based defence algorithm is developed and analysed in the remainder of this work.

2.8 INTEGRATED SENSING AND DEFENCE ALGORITHMS

The requirements of the CRN, as implied by the cognitive cycle, result in the need for many complicated algorithms to execute together. In the physical layer alone, the CRN needs to sense, fuse, allocate resources, distribute messages, and run security algorithms. Furthermore, some of these algorithms must run between sensing time increments. As a result, there is a need for the simplification of computation while sacrificing as little of the intended functionality as possible. This can be achieved by integrating similar processes into one algorithm, thus limiting bandwidth use and computation delays across the network. Since Byzantine attack mitigation is closely linked to cooperative spectrum sensing, these two processes can be combined for more efficient implementation in a CRN.

Belief propagation has been used in performing either spectrum sensing or Byzantine attack mitigation exclusively. It can also be used to perform both tasks simultaneously. The combination of a model-based and data-driven approach – provided by the framework of graphical modelling and inference – allows a great degree of expressive power. In addition, the message-passing structure of belief propagation naturally allows implementations of integrated fusion and defence algorithms to decentralised networks.

2.9 CHAPTER SUMMARY

The functionality of CR was considered in terms of the cognitive cycle. Spectrum sensing and decision-making strategies were discussed, along with the vulnerabilities to these processes through physical layer attacks. One of these attacks, termed the Byzantine attack, was analysed in detail; a framework for attack strategy formulation was presented, and various defence techniques were discussed. The literature's lack of decentralised defence algorithms, along with the need for integrated sensing-defence schemes, was highlighted. The proposed algorithm detailed in the remainder of this work serves to address this gap, and the formulation and realisation of the proposed algorithm is considered in the following chapters. Chapter 3 focuses on the formulation of the statistical model of the relay-based sensing process. Chapter 4 concerns the application of belief propagation on the developed graphical model, along with how this is realised in a relay-based CRN.

CHAPTER 3 PROBABILISTIC INFERENCE FOR SECURE SENSING

3.1 CHAPTER OVERVIEW

In the previous chapter, lack of research into integrated sensing and defence algorithms for decentralised networks was identified. As a result, a novel secure sensing algorithm for relay-based networks is developed and tested over the remainder of this work. This chapter describes the development of the statistical model forming the foundation of the proposed algorithm. This model allows the PU state and the maliciousness of any SUs to be determined simultaneously with the aid of probabilistic inference, allowing the proposed algorithm to be an integrated sensing and defence scheme. The determination of the desired phenomena is facilitated by joint estimations obtained from the statistical model. A system model of the relay-based network is first defined in Section 3.2. The notation is introduced, and the sensing procedure applied by the SUs is presented. The system variables, including the SU measurements, reports, and maliciousness of the MUs are defined and discussed. The processes by which a measurement is made by an SU (and subsequently falsified if it is an MU) are explained. Historical information that is useful to the sensing process is identified.

In Section 3.3, the utility of probabilistic inference in determining joint estimations is discussed. The variables introduced in Section 3.2 are partitioned based on whether they are directly observable or latent. The general form of a joint distribution describing all outcomes of the sensing process is obtained using this partition. Marginal inference can be performed on this distribution to obtain the joint estimates needed for the proposed algorithm; the brute-force computation associated with marginal inference is too inefficient for use in real-time spectrum sensing. Thus, an alternative technique – belief propagation – is chosen to perform efficient inference on the distribution.

Applying belief propagation requires the formulation of a graphical model, which is an alternative representation of the joint distribution. A graphical model visually expresses the joint distribution in terms of the conditional relationships of the system variables. The discovery and definition of these relationships are found in Section 3.4. A graphical model based on the conditional dependencies is then constructed in Section 3.5. Candidate graphical models are explored first, with an appropriate model chosen to capture the interactions between the variables sufficiently. A graphical model of the sensing process is then built, allowing the mathematical form of the corresponding joint distribution to be obtained. This concludes the development of the statistical model, with the efficient inference and implementation of this model considered in Chapter 4.

3.2 SYSTEM MODEL

The proposed algorithm aims to combine the sensing reports from the SUs of the network to perform the following two tasks:

1. determine the true state of the PU, and
2. identify which SUs are malicious.

The above two tasks are to be performed on a relay-based CRN, with any computations distributed across the relays of the network. The following two types of SUs are used to achieve this:

1. **Trusted users (TUs)** – These SUs function as the relays of the network, and are responsible for data fusion. These users have established mutual trust; sensing reports made by these users are always assumed to be honest (i.e. TUs never maliciously alter their reports).
2. **Untrusted users (UTUs)** – These SUs, forming the bulk of the network, include any devices with some sensing capability that are added to the network on an *ad hoc* basis. Prior to sensing, they are assumed to be untrusted and may be dishonest (i.e. some UTUs may maliciously alter their reports). In other words a subset of malicious UTUs exists.

All SUs contribute to spectrum sensing by measuring and reporting their PU occupancy estimates, but only TUs perform data fusion. Initially, the TUs are unaware of which UTUs are malicious. Thus, the proposed algorithm needs to discern the subset of MUs from the UTUs while maintaining an accurate PU state estimation. These tasks are dependent upon each other, and it is desirable to perform them jointly.

There are N_k TUs (indexed by k) and N_m UTUs (indexed by m) in the relay-based network. If SUs are referred to generally, they are indexed by n. The total number of SUs in the CRN is $N_{su} = N_k + N_u$. The sensing process is segmented over discrete time increments $t = 1, t = 2, \cdots, t = T$. The true state of the PU at time t is represented by $\mathcal{H}^{(t)} = 1 = \mathcal{H}_1^{(t)}$ when the PU is broadcasting in the environment, and $\mathcal{H}^{(t)} = 0 = \mathcal{H}_0^{(t)}$ when the PU is absent. At time t, an SU senses the spectrum by measuring the power in the PU's allocated band. It collects R samples (denoted by s), storing them as a vector:

$$\mathbf{s} = [s_1, s_2, \cdots, s_R]. \tag{3.1}$$

Hard fusion is used to save bandwidth; the nth SU's measurement $\mathbf{s}_n^{(t)}$ is quantised to a PU state estimate $u_n^{(t)} \in \{\mathcal{H}_0^{(t)}, \mathcal{H}_1^{(t)}\}$. An SU broadcasts its sensing reports in the form $y_n^{(t)} \in \{\mathcal{H}_0^{(t)}, \mathcal{H}_1^{(t)}\}$. A measurement made by a TU is denoted by $u_{t_k}^{(t)}$, and reports made by TUs are always the same as their measurements (i.e. $u_{t_k}^{(t)} = y_{t_k}^{(t)}$). A measurement is denoted by $u_m^{(t)}$ for UTUs. Reports made by UTUs ($y_m^{(t)}$) are not always the same as their measurements, as malicious UTUs may alter their sensing results. The global PU state estimate, formed by fusing all SU reports, is given by $\hat{\mathcal{H}}^{(t)}$. For the remainder of this work, notation will be abused for clarity, and the superscript "(t)" will be dropped when referring to sensing in the current time instant.

3.2.1 Information transfer in the network

Figure 3.1 shows an example of the described network, with three TUs and five UTUs[1]. The TUs are enumerated prior to sensing. Each UTU sends its sensing reports to the most conveniently located TU. The kth TU shares and processes the information with either of its adjacent neighbours in the form of $M_{k,k+1}$ or $M_{k,k-1}$. The proposed algorithm is implemented by sending this information across the TUs.

3.2.2 Obtaining u_n – individual estimation of the PU state

The conversion of the spectral power $\mathbf{s}_n$ to a PU state estimate u_n is facilitated by a likelihood test. The magnitude of $\mathbf{s}_n$ is compared against a threshold that is based on the noise characteristics of the environment. Assuming that additive white Gaussian noise is present in the environment, the following threshold can be used [47]:

$$\tau(\epsilon) = \sigma_v^2 \left[1 + \sqrt{R} Q^{-1}(\epsilon)\right], \tag{3.2}$$

[1]Note that these numbers are chosen for clarity. The number of UTUs is typically much larger than the number of TUs.

where ϵ is the desired false alarm rate, σ_v^2 is the noise power in the environment, and $Q^{-1}(x)$ is the inverse Q-function. It is assumed that the environment in which the CRN exists is statistically stationary; the threshold remains the same for all SUs over the sensing period. The likelihood test is formulated using the defined threshold $\tau(\epsilon)$:

$$F_n = \frac{1}{R}||\mathbf{s}_n||^2 - \tau(\epsilon), \tag{3.3}$$

where F_n is the n^{th} SU's test statistic, which is converted to u_n as follows:

$$u_n = \begin{cases} 0, & \text{if } F_n < 0 \\ 1, & \text{if } F_n \geq 0 \end{cases}. \tag{3.4}$$

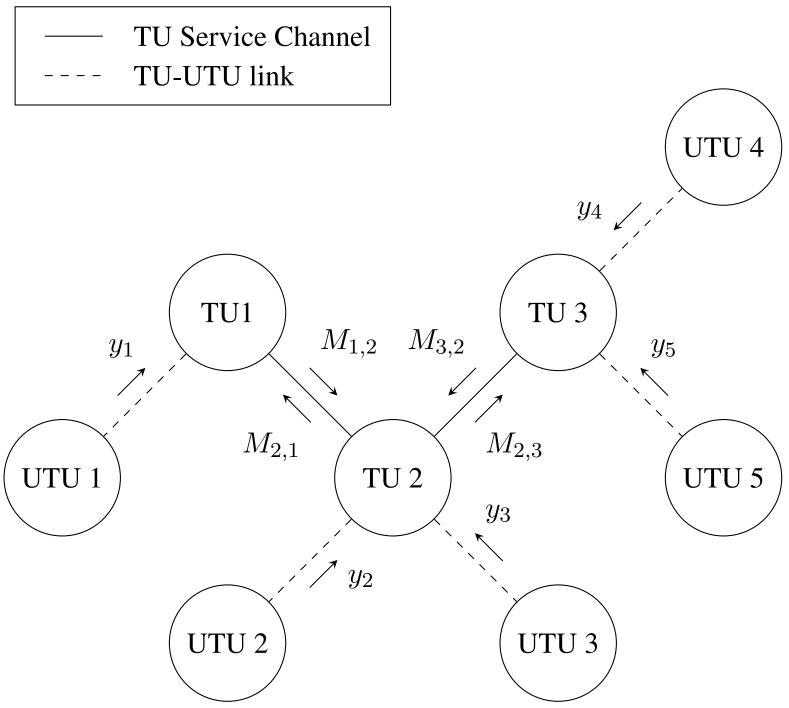


Figure 3.1. Example of the described relay-based network.

3.2.3 Obtaining y_n – reported PU state estimations

In general, it is assumed that a malicious group exists within the group of UTUs. Maliciousness, for the scope of this work, is defined as the frequency with which an MU changes its sensing reports. The maliciousness of a UTU m is determined by the variable $r_m \in [-1, 1]$. When $r_m < 0$, the MU tends to alter its reports in favour of enforcing missed detections. Conversely, $r_m > 0$ results in a skew towards false alarm reports. The larger the magnitude of r_m, the greater the maliciousness of the UTU m [10]. In the event that an MU with $r_m < 0$ measures $u_m = 1$, the probabilities of the MU altering or keeping its sensing result are governed by

$$\begin{aligned} &p(y_m = 0|u_m = 1) = |r_m|, \text{ or} \\ &p(y_m = 1|u_m = 1) = 1 - |r_m|, \end{aligned} \tag{3.5}$$

respectively. Similarly, if an MU with $r_m > 0$ measures $u_m = 0$, the probabilities associated with the MU's reports are

$$\begin{aligned} &p(y_m = 0|u_m = 0) = 1 - r_m, \text{ or} \\ &p(y_m = 1|u_m = 0) = r_m. \end{aligned} \tag{3.6}$$

It is assumed that the MUs do not change their maliciousness over time. For TUs, $y_{t_k} = u_{t_k}$ in all cases. Note that, using the framework described in Section 2.6.1, the MU strategy employed here is a decentralised, population-variable, network-independent, and probabilistic attack.

3.2.4 Storage of historical information

The variables defined thus far are only specific to a single time instant. Historical information can improve the robustness of the PU state estimation. The behaviour of the SUs can be averaged over a time period to account for possible outliers and variations in the reports for the current time instant y_n. Information from time instants $t = 1$ to $t = T$ can be represented in the form of previous PU state estimations

$$\hat{\boldsymbol{\mathcal{H}}}^{(T)} = [\hat{\mathcal{H}}^{(1)}, \hat{\mathcal{H}}^{(2)}, \cdots, \hat{\mathcal{H}}^{(T)}], \tag{3.7}$$

and sensing reports broadcast by all SUs:

$$\mathbf{Y}^{(T)} = \begin{bmatrix} y_1^{(1)} & \dots & y_n^{(1)} & \dots & y_{N_{su}}^{(1)} \\ y_1^{(2)} & \dots & y_n^{(2)} & \dots & y_{N_{su}}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_1^{(T)} & \dots & y_n^{(T)} & \dots & y_{N_{su}}^{(T)} \end{bmatrix}. \tag{3.8}$$

The historical information is assumed to be known by the TUs. The tth row of $\mathbf{Y}^{(T)}$ represents the measurements made by all SUs at time increment t. The nth column is given by $\mathbf{y}_n^{(T)}$, and represents all measurements made by SU n until time $t = T$.

3.2.5 Combining reports to estimate $\mathcal{H}$ – cooperative spectrum sensing

The task of the proposed algorithm is now redefined in terms of the newly introduced notation. The proposed algorithm combines the individual sensing reports of each SU for the current time instant $\mathbf{y} = [y_1, \ldots, y_n, \ldots, y_{N_{su}}]$, the sensing reports from the previous T time instants $\mathbf{Y}^{(T)}$, and the historical PU state estimations $\hat{\boldsymbol{\mathcal{H}}}^{(T)}$ to perform the following tasks:

1. determine the current PU state $\mathcal{H}$ by use of an estimate $\hat{\mathcal{H}}$, and
2. determine the subset of MUs by estimating their maliciousness $\mathbf{r} = [r_1, r_2, \cdots, r_m, \cdots, r_{N_m}]$.

An integrated fusion and defence algorithm can be formulated to obtain all estimates simultaneously. Probabilistic inference provides a very convenient framework to achieve this by allowing probability distributions over the latent variables of the system model to be determined. These latent variables cannot be measured directly by the TUs and include the PU state $\mathcal{H}$, UTU maliciousness values $\mathbf{r}$, and UTU measurements $\mathbf{u} = [u_1, u_2, \cdots, u_m, \cdots, u_{N_m}]$. Furthermore, applying inference in this manner leads to the graphical representation of a probability distribution that can be used to compute the inference in a distributed manner – a property probabilistic inference methods hold over their alternatives (i.e. consensus and diffusion algorithms).

3.3 COOPERATIVE SPECTRUM SENSING BY PROBABILISTIC INFERENCE

A joint distribution of a system associates every possible variable state combination of that system with a probability density. Inference can be performed on this distribution to extract useful information, which requires a full description of the distribution's mathematical form. In formulating the joint distribution, it is first important to define the variables that exist in this system, and subsequently to partition those variables into known (i.e. measurable) and desired (i.e. latent) subsets. The joint distribution is then formulated in terms of the desired variables and conditioned on the known variables. Let $\mathcal{P}_s$ represent the joint distribution of the sensing process composed of the variable set $\mathbf{X}_s$. The desired variables are in the set $\mathbf{Z}_s$, and the observed variables are in the set $\mathbf{E}_s$. The following partitions are present in the sensing model:

1. The system of variables is

$$\mathbf{X}_s = \left\{ \mathbf{u}, \mathbf{r}, \mathcal{H}, \hat{\boldsymbol{\mathcal{H}}}^{(T)}, \mathbf{Y}^{(T)} \right\}. \tag{3.9}$$

2. The set of variables known by the TUs at time t, which functions as the evidence for the statistical model, is

$$\mathbf{E}_s = \left\{ \hat{\boldsymbol{\mathcal{H}}}^{(T)}, \mathbf{Y}^{(T)} \right\}. \tag{3.10}$$

3. The set of unknown variables desired for inference is

$$\mathbf{Z}_s = \{\mathbf{u}, \mathbf{r}, \mathcal{H}\}. \tag{3.11}$$

Thus, the form of the observed joint distribution at time t is

$$p(\mathbf{Z}_s|\mathbf{E}_s) = p(\mathbf{u}, \mathbf{r}, \mathcal{H}|\hat{\boldsymbol{\mathcal{H}}}^{(T)}, \mathbf{Y}^{(T)}). \tag{3.12}$$

The following notation is used to condense and imply the conditional relationship between the desired and known variables:

$$\mathcal{P}_s = p_s(\mathbf{u}, \mathbf{r}, \mathcal{H}) = p(\mathbf{u}, \mathbf{r}, \mathcal{H}|\hat{\boldsymbol{\mathcal{H}}}^{(T)}, \mathbf{Y}^{(T)}). \tag{3.13}$$

The raw information contained in $\mathcal{P}_s$ requires extraction to provide usable information for sensing. This is done by the inference of (3.13).

3.3.1 Inference of the joint distribution $\mathcal{P}_s$

Inference is enabled by the computation of the marginals associated with (3.13). Marginals are distributions with respect to subsets of variables belonging to the parent joint distribution. For the problem at hand, the TUs require estimates of the PU state, the maliciousness of all UTUs, and the original measurements of all UTUs. Thus, the marginals are of the form

$$\begin{aligned} b(u_m) &\approx p_s(u_m) = p(u_m|\hat{\boldsymbol{\mathcal{H}}}^{(T)}, \mathbf{Y}^{(T)}), \\ b(r_m) &\approx p_s(r_m) = p(r_m|\hat{\boldsymbol{\mathcal{H}}}^{(T)}, \mathbf{Y}^{(T)}), \text{ and} \\ b(\mathcal{H}) &\approx p_s(\mathcal{H}) = p(\mathcal{H}|\hat{\boldsymbol{\mathcal{H}}}^{(T)}, \mathbf{Y}^{(T)}), \end{aligned} \tag{3.14}$$

where $b(u_m)$, $b(r_m)$, and $b(\mathcal{H})$ are estimates of the marginals called “beliefs”.

The marginals in (3.14) are a set of posterior probability distributions, each over a single desired variable. Obtaining these marginals gives insight into the states that are favoured by the model, given the evidence $\hat{\boldsymbol{\mathcal{H}}}^{(T)}$ and $\mathbf{Y}^{(T)}$. These favoured states are determined using the MAP estimate; this is the state in which the posterior marginal distribution is at its maximum:

$$\begin{aligned} \hat{u}_m &= \underset{u_m}{\operatorname{argmax}}\, b(u_m), \\ \hat{r}_m &= \underset{r_m}{\operatorname{argmax}}\, b(r_m), \\ \hat{\mathcal{H}} &= \underset{\mathcal{H}}{\operatorname{argmax}}\, b(\mathcal{H}). \end{aligned} \tag{3.15}$$

The MAP estimates $\hat{u}_m$, $\hat{r}_m$, and $\hat{\mathcal{H}}$ jointly represent the most likely variable states. This is because $\mathcal{P}_s$ considers the interaction between the variables multilaterally. These estimates become the final decisions made by the network (i.e. the network concludes that the PU band is vacant if $\hat{\mathcal{H}} = \mathcal{H}_0 = 0$ and *vice versa*). Thus, marginal inference of (3.13) accomplishes the following tasks simultaneously:

1. Detection of MUs – If the evidence deems it likely that a UTU report has been altered (i.e. $y_m \neq u_m$), the belief $b(r_m)$ is shifted such that the UTU is considered more malicious.
2. Weighting of UTU contributions – The contribution of a report coming from a UTU deemed malicious is lowered. In some cases, if the MU is observed to be extremely malicious (i.e. $|\hat{r}_m| >> 0$), its report is reversed (i.e. $y_m = \mathcal{H}_0$ is changed to $y_m = \mathcal{H}_1$). These changes affect the PU state prediction.
3. Determining the most likely PU state – The MAP estimate $\hat{\mathcal{H}}$ is the peak of the belief $b(\mathcal{H})$. This estimate is the amalgamation of MU detection and UTU report weighting/combining. For example, the model would deem $\hat{\mathcal{H}} = \mathcal{H}_1 = 1$ more likely when a large number of re-weighted UTU measurement estimates are $\hat{u}_m = 1$.

In summary, the model variables are combined to form the joint distribution in (3.13); this is done prior to sensing. The estimates $\hat{\mathbf{u}}$, $\hat{\mathbf{r}}$, and $\hat{\mathcal{H}}$ are then obtained from the inference of the joint distribution, which is done during sensing. This process is illustrated in Figure 3.2. Thus, cooperative spectrum sensing is equivalent to performing marginal inference on (3.13) to obtain the estimates in (3.15).

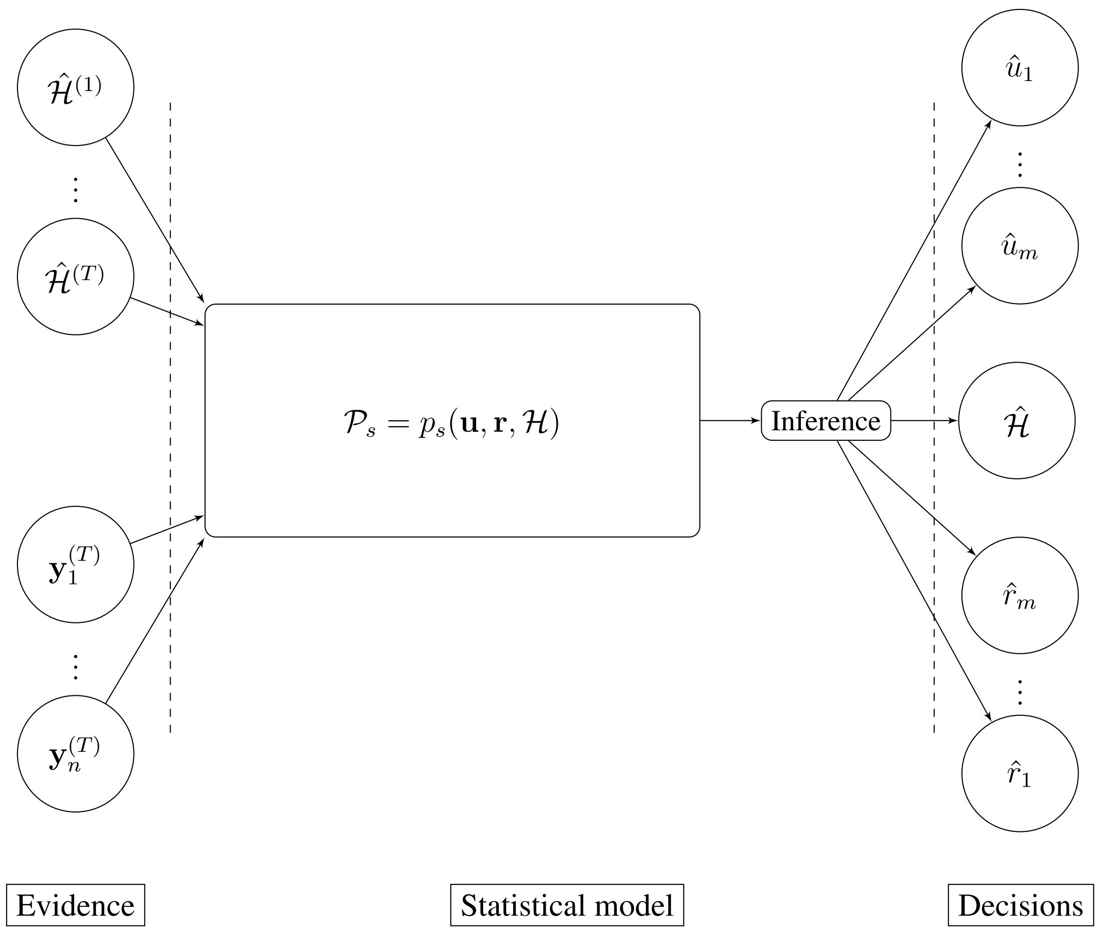


Figure 3.2. High-level view of how the variables are combined and extracted to perform cooperative spectrum sensing.

3.3.2 Computation of the marginals

The brute-force computation of the marginals in (3.14) involves summing (for a discrete variable) or integrating (for a continuous variable) $\mathcal{P}_s$ over all unknown variables, with the exception of the variable that is the subject of the marginal distribution. For the problem at hand, the brute-force approach yields the following expressions for the marginals:

$$\begin{aligned} p_s(u_m) &\propto \int_{-1}^{1} \sum_{\mathbf{u}\setminus u_m} \sum_{\mathcal{H}} \mathcal{P}_s d\{\mathbf{r}\}, \\ p_s(r_m) &\propto \int_{-1}^{1} \sum_{\mathbf{u}} \sum_{\mathcal{H}} \mathcal{P}_s d\left\{\mathbf{r} \setminus r_m\right\}, \text{ and} \\ p_s(\mathcal{H}) &\propto \int_{-1}^{1} \sum_{\mathbf{u}} \mathcal{P}_s d\{\mathbf{r}\}, \end{aligned} \tag{3.16}$$

where $d\{\mathbf{r}\}$ denotes an iterated integration over the elements of the vector $\mathbf{r}$. The notation "$\mathbf{u} \setminus u_m$" and "$\mathbf{r} \setminus r_m$" represents the set of $\mathbf{u}$ and $\mathbf{r}$ with the exception of u_m and r_m respectively. Applying the brute-force approach becomes prohibitively costly for joint distributions of many variables. The general computational cost of the brute-force approach is $\mathcal{O}(L^N)$, where L is the number of states for each variable (assuming all variables have the same number of states) and N is the total number of variables [45]. The brute-force approach assumes that no variables in $\mathcal{P}_s$ are conditionally independent – an assumption that is not generally true.

3.4 VARIABLE INTERACTIONS IN THE SENSING MODEL

Probabilistic graphical models are representations of joint distributions in terms of their conditional dependencies. The dependence structure of a distribution, exposed by representing it as a graphical model, can be exploited to perform marginal inference much more efficiently in comparison to the brute-force case. For the special class of graphical models that represent Markov chains, the computational cost can be reduced to $\mathcal{O}(L^2 N)$. This approach, which involves scheduling and saving intermediate summations and products in the computation of the marginal distribution, is called belief propagation [48], [49]. Thus, if efficient marginal inference is desired, $\mathcal{P}_s$ needs to be expressed graphically. Before doing this, it is important to identify and define the conditional dependencies present between the variables in $\mathbf{X}_s$, as this will allow the most appropriate graphical model to be chosen. This section explores and defines the variable interactions present in the sensing model of Section 3.2. These interactions will be defined in terms of likelihood functions. These functions are eventually combined to form a graphical model $\mathcal{G}_s$ that corresponds to $\mathcal{P}_s$.

3.4.1 PU state $\mathcal{H}$, and SU measurements u_m and u_{t_k}

When the PU is broadcasting (i.e. $\mathcal{H} = \mathcal{H}_1 = 1$), an SU's original measurement is more likely to correspond to the PU state (i.e. $u_n = \mathcal{H}_1 = 1$). This is also true when the PU is vacant. Thus, knowledge of $\mathcal{H}$ changes the expectation of an SU's measurement. A reliable SU that is placed in a favourable location is more likely to measure the state of the PU correctly. A simple relationship between an SU n's measurement and the PU state $\mathcal{H}$ can be determined by the abstraction of an SU's reliability using the error probabilities p_{fa}^n and p_{md}^n. An SU with a large false alarm probability is prone to incorrectly measuring a PU signal when there is none, and an SU with a large missed detection probability will often fail to sense a PU signal's presence.

Table 3.1 shows the function $\alpha_{t_k}(u_{t_k}, \mathcal{H})$ that expresses the relationship between a TU's measurement u_{t_k} and the PU state $\mathcal{H}$. A "$*$" denotes a variable that is unknown prior to sensing (i.e. a desired variable). For α_{t_k}, u_{t_k} is known since the SU k is trusted ($u_{t_k} = y_{t_k}$).

Table 3.1. Description of the likelihood function α_{t_k} for a TU k.

$\alpha_{t_k}(u_{t_k}, \mathcal{H}^*)$	$\mathcal{H}^* = 0$	$\mathcal{H}^* = 1$
$u_{t_k} = 0$	$1 - p^k_{md}$	p^k_{md}
$u_{t_k} = 1$	p^k_{fa}	$1 - p^k_{fa}$

Table 3.2 shows the function $\alpha_{u_m}(u_m, \mathcal{H})$ expressing the relationship between a UTU's measurement u_m and the PU state $\mathcal{H}$. This function is mostly similar to α_{t_k}, with the exception that u_m is unknown, since it is not guaranteed to be the same as y_m.

Table 3.2. Description of the likelihood function α_{u_m} for a UTU m.

$\alpha_{u_m}(u^*_m, \mathcal{H}^*)$	$\mathcal{H}^* = 0$	$\mathcal{H}^* = 1$
$u^*_m = 0$	$1 - p^m_{md}$	p^m_{md}
$u^*_m = 1$	p^m_{fa}	$1 - p^m_{fa}$

In both cases, a higher likelihood for a measurement corresponding to the PU state is assigned to an SU with smaller error probabilities. Each TU is associated with a single α_{t_k} factor, and each UTU is associated with a single α_{u_m} factor.

3.4.2 TU reports y_{t_k} and UTU reports y_m

Shared spatial and environmental factors result in a correlation between TU reports $y_{t_k} = u_{t_k}$ and UTU reports y_m, both of which are linked to the PU state $\mathcal{H}$. This correlation can be leveraged to refine the estimation of the PU state. Consider a pair of two SUs: one is a TU k, and the other is a UTU m. The correlation between these SUs' reports can be represented by two probabilities $\zeta_{mk} = p(y_m = 0|y_{t_k} = 0)$ and $\tau_{mk} = p(y_m = 1|y_{t_k} = 1)$. Large values of ζ_{mk} and τ_{mk} mean that UTU m and TU k make similar reports. A small value of ζ_{mk} or τ_{mk} implies that UTU m makes dissimilar reports to TU k when $y_{t_k} = 0$ or $y_{t_k} = 1$ respectively. The function $\gamma_{mk}(u_{t_k}, y_m, \mathcal{H})$ relates the relevant variables to these quantities, as shown in Table 3.3. Note that y_{t_k} is replaced by u_{t_k} for uniformity in notation with the other factors.

Table 3.3. Description of the likelihood function γ_{mk}.

$\gamma_{mk}(u_{t_k}, y_m, \mathcal{H}^*)$		$\mathcal{H}^* = 0$	$\mathcal{H}^* = 1$
$y_m = 0$	$u_{t_k} = 0$	ζ_{mk}	$1 - \zeta_{mk}$
	$u_{t_k} = 1$	τ_{mk}	$1 - \tau_{mk}$
$y_m = 1$	$u_{t_k} = 0$	$1 - \zeta_{mk}$	ζ_{mk}
	$u_{t_k} = 1$	$1 - \tau_{mk}$	τ_{mk}

The probabilities ζ_{mk} and τ_{mk} are assumed to be unknown and need to be determined during sensing. This can be achieved in various ways. The simplest technique to estimate ζ_{mk} and τ_{mk} is to take the sum of all reports where $y_m = u_{t_k} = 1$ and $y_m = u_{t_k} = 0$ until time $t = T$:

$$\begin{aligned} \hat{\zeta}_{mk} &= \frac{\sum_{t=1}^{T} \{u_{t_k} = y_m\}^0}{T}, \\ \hat{\tau}_{mk} &= \frac{\sum_{t=1}^{T} \{u_{t_k} = y_m\}^1}{T}, \end{aligned} \tag{3.17}$$

where $\{u_{t_k} = y_m\}^i = 1$ when $u_{t_k} = y_m = i$, and is 0 otherwise. These estimates $\hat{\zeta}_{mk}$ and $\hat{\tau}_{mk}$ are used in place of the true values during sensing. Every pair of connections between the UTUs m and the TUs k is associated with a γ_{mk} function.

3.4.3 Historical information, UTU reports y_m, and maliciousness r_m

If a UTU m is malicious, it cannot be guaranteed that $u_m = y_m$. When $|r_m|$ is large, there is a high probability that $u_m \neq y_m$, with the converse true for small $|r_m|$. A function $\delta_{a_m}(y_m, u_m, r_m)$ can be used to capture the relationship between a UTU's measurement, report, and maliciousness – this function is shown in Table 3.4 [10]. The output of δ_{a_m} gives the likelihood of the original measurement and maliciousness given the UTU's report.

Table 3.4. Description of the likelihood function δ_{a_m}.

$\delta_{a_m}(y_m, u_m^*, r_m^*)$		$y_m = 0$	$y_m = 1$
$u_m^* = 0$	$r_m^* > 0$	$1 - r_m^*$	r_m^*
	$r_m^* \leq 0$	1	0
$u_m^* = 1$	$r_m^* \geq 0$	0	1
	$r_m^* < 0$	$\|r_m^*\|$	$1 - \|r_m^*\|$

Another similar function can be used to refine the estimation of r_m further by using the UTU's historical sensing performance. In this case, the PU state estimate $\hat{\mathcal{H}}$ and UTU report from historical time instant $i \leq T$ can be used to infer the maliciousness. The function $\delta_{b_m}^i(y_m^{(i)}, \hat{\mathcal{H}}^{(i)}, r_m)$, shown in Table 3.5, fulfils this task.

Table 3.5. Description of the likelihood function $\delta_{b_m}^i$.

$\delta_{b_m}^i(y_m^{(i)}, \hat{\mathcal{H}}^{(i)}, r_m^*)$		$y_m^{(i)} = 0$	$y_m^{(i)} = 1$
$\hat{\mathcal{H}}^{(i)} = 0$	$r_m^* > 0$	$1 - r_m^*$	r_m^*
	$r_m^* \leq 0$	1	0
$\hat{\mathcal{H}}^{(i)} = 1$	$r_m^* \geq 0$	0	1
	$r_m^* < 0$	$\|r_m^*\|$	$1 - \|r_m^*\|$

Each UTU is associated with a single δ_{a_m} function and a set of $\delta_{b_m}^i$ functions for each historical time instant $i \leq T$.

Figure 3.3 summarises how the variables are related to the functions. This diagram is specific to a subject TU k and UTU m. The clear and hatched nodes represent the known and unknown variables respectively.

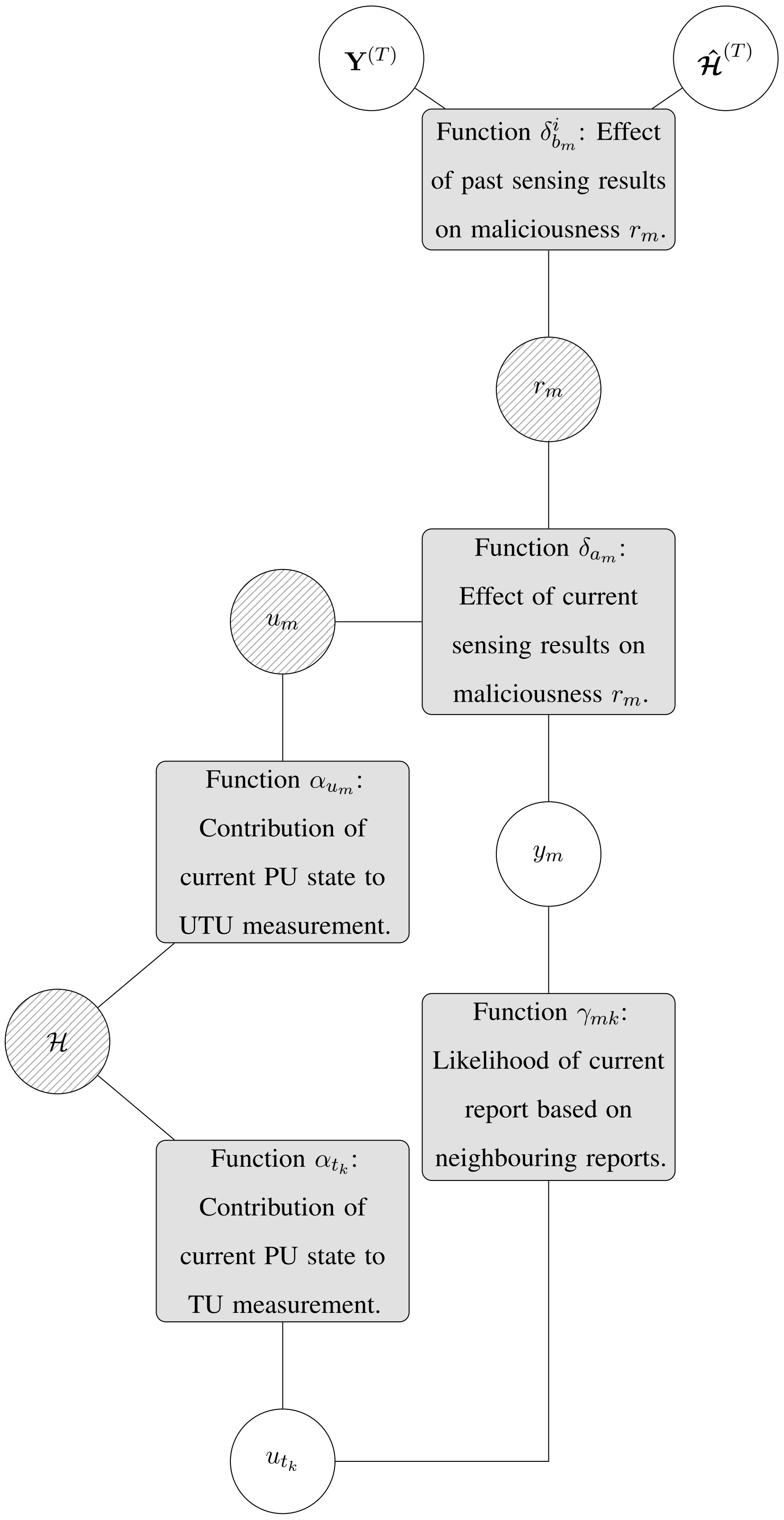

$\mathbf{Y}^{(T)}$
$\hat{\boldsymbol{\mathcal{H}}}^{(T)}$
Function $\delta^{i}_{b_m}$: Effect of past sensing results on maliciousness r_m.
r_m
Function δ_{a_m}: Effect of current sensing results on maliciousness r_m.
u_m
Function α_{u_m}: Contribution of current PU state to UTU measurement.
y_m
$\mathcal{H}$
Function γ_{mk}: Likelihood of current report based on neighbouring reports.
Function α_{t_k}: Contribution of current PU state to TU measurement.
u_{t_k}

Figure 3.3. Diagram showing how the variables in the sensing model are related to one other.

Note that the conditional dependencies between the variables have already been exposed in Figure 3.3 – observing y_m changes the knowledge of r_m which influences the knowledge of u_m, which in turn influences the knowledge of $\mathcal{H}$. Note that arrows are not present between the nodes as these conditional relationships work both ways, and for any path across the variables. The joint distribution $\mathcal{P}_s$ is mathematically expressed as the product of the functions α_{t_k}, α_{u_m}, γ_{mk}, δ_{a_m}, and $\delta^i_{b_m}$ – the order in which the product is structured is highlighted using an appropriate graphical model.

3.5 FORMULATION OF THE GRAPHICAL MODEL

Since the variable interactions have been established, a graphical model $\mathcal{G}_s$ representing $\mathcal{P}_s$ can now be determined. As mentioned previously, the graph captures the conditional relationships between the variables in $\mathbf{X}_s$. This is essential for the efficient computation of the marginal distributions of $\mathcal{P}_s$. Furthermore, the mathematical expression of $\mathcal{P}_s$ can be obtained from $\mathcal{G}_s$ in a straightforward and intuitive manner. A suitable type of graphical model must first be chosen to encode the interactions of Section 3.4 accurately in terms of conditional dependencies.

3.5.1 Types of graphical models

A variety of graphical models exist, including Bayesian networks, Markov random fields (MRFs), and factor graphs. These models are explored in this section, and a suitable model is selected based on the aforementioned variable interactions. A generic graphical model $\mathcal{G}$ and its associated joint distribution $\mathcal{P}$ will be used here. The model $\mathcal{G}$ contains a variable set $\mathbf{X}$.

3.5.1.1 Bayesian networks

Bayesian networks are directed acyclic graphs that represent unilateral conditional relationships between variables. The graph consists of variable nodes (each representing a variable in $\mathbf{X}$) connected by arrows pointing at each neighbour. An arrow pointing from a variable x_i at another variable x_j represents the direction of influence between the variables. In other words, knowledge of x_i leads to an update in the knowledge of x_j:

$$p(x_j, x_i) = p(x_j|x_i)p(x_i). \tag{3.18}$$

As a result, the Bayesian network is a graph $\mathcal{G}$ that expresses a joint distribution $\mathcal{P}$ in terms of its conditional relationships. Consider the following example distribution, with $\mathbf{X} = \{x_1, x_2, x_3, x_4, x_5\}$:

$$\mathcal{P} = p(\mathbf{X} = \{x_1, x_2, x_3, x_4, x_5\}) = p(x_1|x_2, x_3)p(x_3)p(x_2|x_4, x_5)p(x_4)p(x_5). \tag{3.19}$$

The equivalent Bayesian network of the distribution in (3.19) is shown in Figure 3.4.

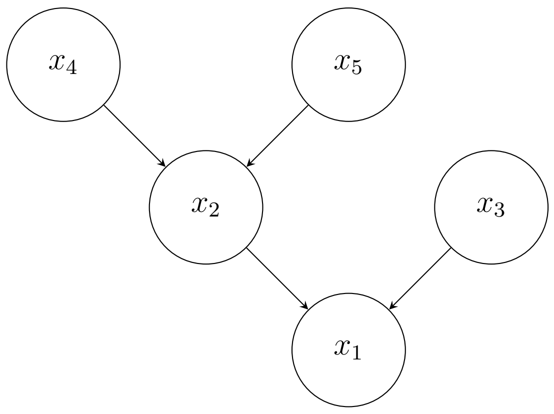


Figure 3.4. Example of a Bayesian network with $\mathbf{X} = \{x_1, x_2, x_3, x_4, x_5\}$.

Note that Bayesian networks cannot represent bilateral dependence structures where $p(x_j|x_i)$ and $p(x_i|x_j)$ both exist; these networks are appropriate for modelling systems of variables that display clear unilateral interactions between each other.

3.5.1.2 Markov random fields

MRFs are undirected graphs that can contain cycles. This allows them to capture cyclic bilateral influences between systems of variables. Similarly to Bayesian networks, MRFs contain variable nodes corresponding to each element of $\mathbf{X}$. The edge connecting a node x_i to another node x_j is associated with a potential function $\psi_{ij} = \psi_{ji}$. The function ψ_{ij} represents the mutual likelihood of x_i and x_j. The potential is formulated such that $\psi_{ij} >> 0$ if x_i and x_j are observed, and these observations are deemed likely by the model.

The joint distribution corresponding to an MRF, with a system of variables $\mathbf{X}$, is the product of all edge potentials in the graph:

$$\mathcal{P} = p(\mathbf{X} = \{x_1, \cdots, x_N\}) = \frac{1}{Z} \prod_{\{i,j\} \in \mathcal{G}} \psi_{ij}(x_i, x_j), \tag{3.20}$$

where $\{i, j\}$ refers to a pair of connected variables x_i and x_j in $\mathcal{G}$, and Z is the normalising constant required to make (3.20) a valid probability distribution. An example of an MRF is shown in Figure 3.5 below.

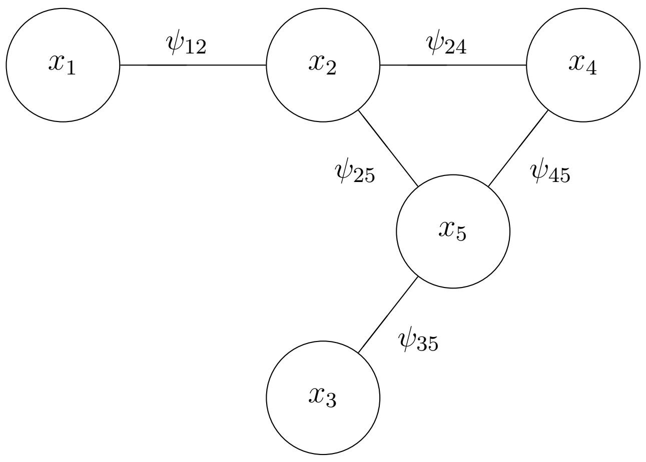


Figure 3.5. Example of an MRF with $\mathbf{X} = \{x_1, x_2, x_3, x_4, x_5\}$.

The joint distribution corresponding to $\mathcal{G}$ in Figure 3.5 is

$$\mathcal{P} = p(\mathbf{X} = \{x_1, x_2, x_3, x_4, x_5\}) = \frac{1}{Z}\psi_{12}(x_1, x_2)\psi_{24}(x_2, x_4)\psi_{25}(x_2, x_5)\psi_{35}(x_3, x_5)\psi_{45}(x_4, x_5). \tag{3.21}$$

MRFs are limited to capturing models with only bilateral influences between a pair of variables, as the edge potentials are only functions of two variables. These models are suitable for applications where correlations are present between system variables.

3.5.1.3 Factor graphs

Factor graphs have a fundamentally different graphical architecture in comparison to Bayesian networks and MRFs. Factor graphs consist of two different types of nodes: variables and factors. The variable nodes represent each element of the set $\mathbf{X}$, and define the skeleton of the graph. The factor nodes, which are elements of the set $\mathbf{F} = \{f_1, f_2, \cdots\}$, connect the variable nodes. Factors represent likelihood functions that relate the variables to one other, and are a function of a subset of variables $\mathbf{x}_i$ from $\mathbf{X}$. Thus, a factor graph has the following form: $\mathcal{G} = [\mathbf{X}, \mathbf{F}]$. In $\mathcal{G}$, a factor $f_i(\mathbf{x}_i)$ is connected to all the variables of $\mathbf{x}_i$. The joint distribution represented by $\mathcal{G}$ is the product of all the factors in $\mathbf{F}$:

$$\mathcal{P} = p(\mathbf{X} = \{x_1, \cdots, x_N\}) = \frac{1}{Z}\prod_{f_i \in \mathbf{F}} f_i(\mathbf{x}_i). \tag{3.22}$$

Factor graphs allow for cycles and undirected bilateral relationships between variables, unlike Bayesian networks. Furthermore, factor graphs have greater expressive power than MRFs, as the presence of factor nodes allows more than two variables to be influenced by the same factor. The generality of factor graphs allows Bayesian networks and MRFs to be converted to factor graphs, with the converse not always being possible.

The factor graph representation of the Bayesian network in Figure 3.4 is given in Figure 3.6. In this case, the variable set is $\mathbf{X} = \{x_1, x_2, x_3, x_4, x_5\}$, and the factor set is $\mathbf{F} = \{f_1, f_2, f_3, f_4\}$, with $f_1 = p(x_1|x_2, x_3)$, $f_2 = p(x_3)$, $f_3 = p(x_2|x_4, x_5)$, $f_4 = p(x_4)$, and $f_5 = p(x_5)$. Similarly, the factor graph representation of the MRF in Figure 3.5 is given in Figure 3.7. In this case, the edge potentials make up the factor set such that $f_1 = \psi_{12}(x_1, x_2)$, $f_2 = \psi_{24}(x_2, x_4)$, $f_3 = \psi_{35}(x_3, x_5)$, and $f_4 = \psi_{45}(x_4, x_5)$.

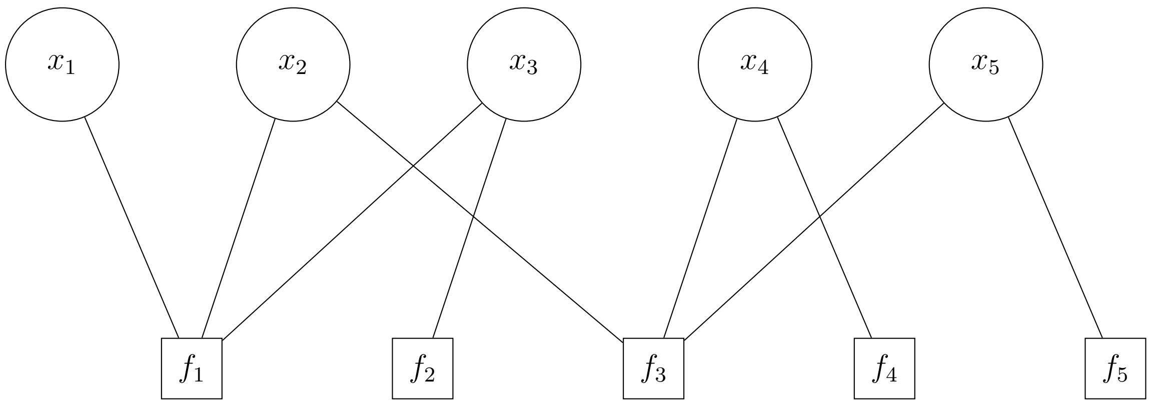


Figure 3.6. Factor graph corresponding to the Bayesian network of Fig. 3.4.

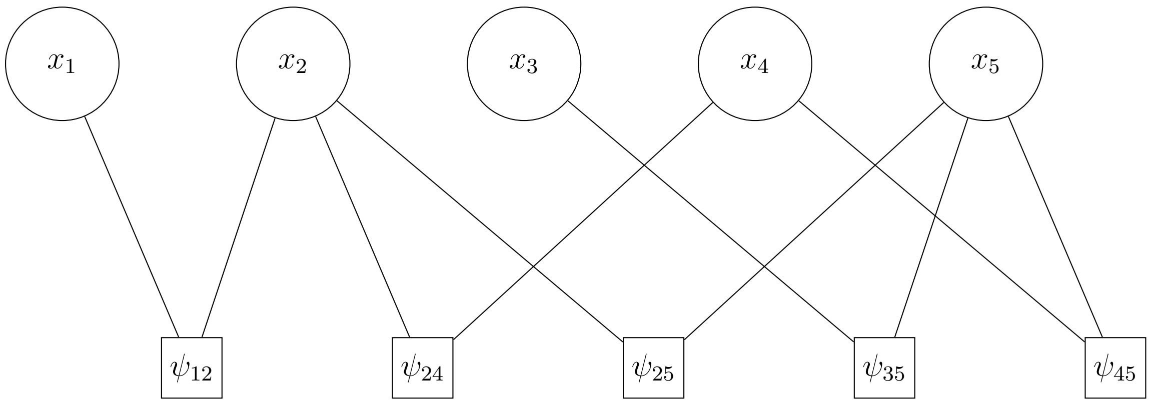


Figure 3.7. Factor graph corresponding to the MRF of Fig. 3.5.

Consider the following joint distribution, assuming that f_1, f_2, and f_3 cannot be factorised any further:

$$\mathcal{P} = p(\mathbf{X} = \{x_1, x_2, x_3, x_4, x_5\}) = f_1(x_1, x_2, x_3)f_2(x_2, x_4, x_5)f_3(x_4, x_5). \tag{3.23}$$

The factor graph corresponding to (3.23) is shown in Figure 3.8. Note that many variables in this graph are influenced by the same factor (i.e. f_1 and f_2), meaning that an MRF would be unable to capture this behaviour. The graph is also cyclic and undirected, properties that a Bayesian network cannot handle. Thus, the behaviour of the system of variables in (3.23) can only be captured by a factor graph.

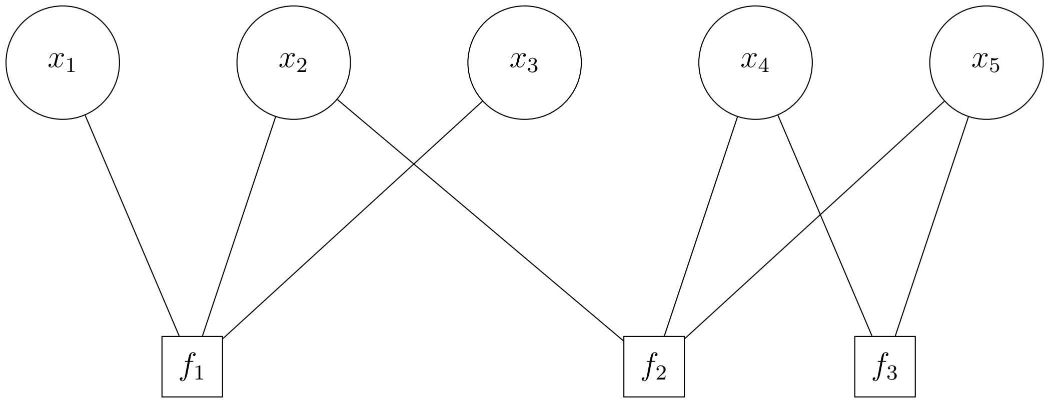


Figure 3.8. Example of a unique factor graph that captures the cyclic multilateral influences between variables.

3.5.2 Choice of graphical model for $\mathcal{P}_s$

The interactions listed in Section 3.4 contain cyclic and multilateral conditional dependence structures. This is highlighted in Figure 3.3, where knowledge of u_{t_k} and u_m changes the knowledge of $\mathcal{H}$, with the reverse scenario also true. These interactions disqualify Bayesian networks as a suitable candidate for the graphical sensing model, as they cannot handle the bilateral influences of the variables in $\mathbf{X}_s$. Now consider the function δ_{a_m}, which accounts for a UTU's maliciousness r_m, its sensing reports y_m, and its true measurements u_m. All three variables are influenced by δ_{a_m} simultaneously, an interaction that cannot be captured by an MRF. Thus, the only graphical model that can capture the interactions between the variables of $\mathbf{X}_s$ is the factor graph.

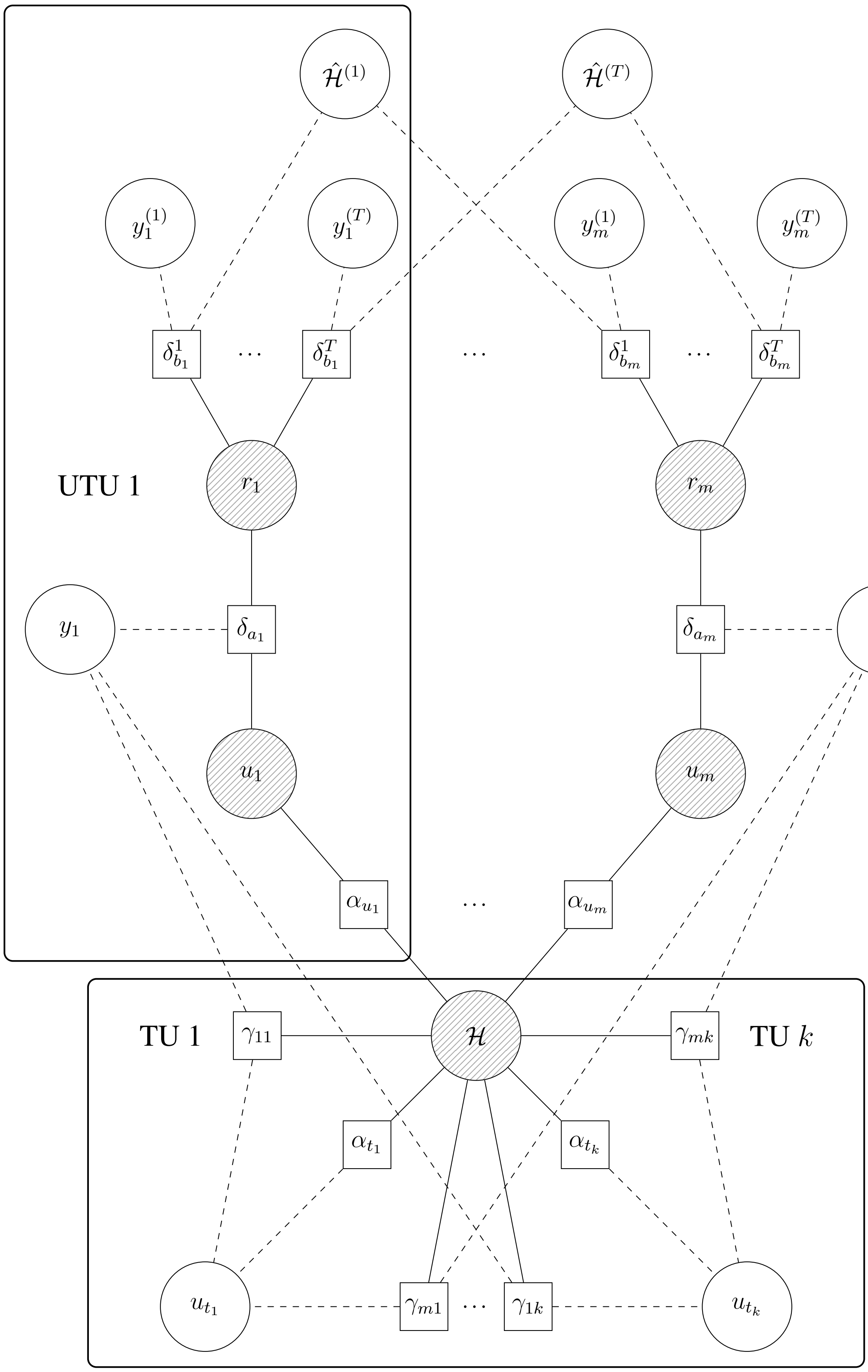


Figure 3.9. Factor graph $\mathcal{G}_s$ showing the relationship between the variables of $\mathbf{X}_s$.

3.5.3 Construction of the factor graph $\mathcal{G}_s$

The interactions of $\mathbf{X}_s$ are summarised as a factor graph, denoted by $\mathcal{G}_s = [\mathbf{X}_s, \mathbf{F}_s]$ in Figure 3.9. The variables and factors are represented by circles and squares respectively. The graph $\mathcal{G}_s$ is formed by considering the functions α_{t_k}, α_{u_m}, δ_{a_m}, $\delta^i_{b_m}$, and γ_{mk} as members of the factor set $\mathbf{F}_s$. An edge in $\mathcal{G}_s$ exists between a given variable in $\mathbf{X}_s$ and all the factor functions associated with that variable. Note that the graph represents the distribution $\mathcal{P}_s$ in which a certain set of variables has already been observed. The goal is to infer the "hidden" unobserved variables. The hatched circles in Figure 3.9 denote the unknown variables in the set $\mathbf{Z}_s$. A solid line denotes the connection between an unknown variable and a factor, and a dashed line denotes the connection between a known variable and a factor.

3.5.4 Construction of the joint distribution

The mathematical form of the joint distribution $\mathcal{P}_s$ can be determined by applying (3.22) to all factors in $\mathcal{G}_s$. Before doing so, it is important to note that $\mathcal{G}_s$ in Figure 3.9 can be conveniently partitioned into subgraphs with

1. a subgraph for every UTU m in the network:

$$\mathcal{G}_m = \left[\mathbf{X}_m = \{u_m, r_m, y_m, \mathbf{y}^{(T)}_m, \mathcal{H}, \hat{\boldsymbol{\mathcal{H}}}^{(T)}\}, \mathbf{F}_m = \{\delta^1_{b_m}, \cdots, \delta^T_{b_m}, \delta_{a_m}, \alpha_{u_m}\}\right], \tag{3.24}$$

2. and a single subgraph for the all TUs combined:

$$\mathcal{G}_k = \left[\mathbf{X}_k = \{\mathbf{u}_{t_k}, \mathcal{H}\}, \mathbf{F}_k = \{\alpha_{t_1}, \cdots, \alpha_{t_k}, \gamma_{m1}, \cdots, \gamma_{mk}, \}\right]. \tag{3.25}$$

3.5.4.1 Local distribution for UTU m

The subgraph $\mathcal{G}_m$ is responsible for the estimation of the m^{th} UTU's maliciousness and the weighting of the UTU's contribution to the PU state estimation. Using Figure 3.9, the corresponding joint distribution for $\mathcal{G}_m$ is

$$\mathcal{P}_m \propto \alpha_{u_m}(u_m, \mathcal{H})\, \delta_{a_m}(u_m, y_m, r_m) \prod_{i \in \{1,2,\ldots,T\}} \delta^i_{b_m}\left(y^{(i)}_m, r_m\right). \tag{3.26}$$

3.5.4.2 Distribution for TUs

The subgraph $\mathcal{G}_t$ combines all the weighted reports and performs the estimation of $\mathcal{H}$. The effects of correlations between neighbouring SU reports, and their impact on $\mathcal{H}$, are also handled by this portion of $\mathcal{G}_s$. The corresponding joint distribution for $\mathcal{G}_k$ is

$$\mathcal{P}_k \propto \prod_{k\in\{1,2,\ldots,N_k\}} \alpha_{t_k}\left(u_{t_k},\mathcal{H}\right) \prod_{m\in\{1,2,\ldots,N_m\}} \gamma_{mk}\left(u_{t_k},y_m,\mathcal{H}\right). \tag{3.27}$$

3.5.4.3 Global joint distribution

Since the combination of the subgraphs $\mathcal{G}_k$ and $\mathcal{G}_m$ forms the complete graph $\mathcal{G}_s$, the global joint distribution is formed by multiplying $\mathcal{P}_k$ (representing all TUs in the network) and $\mathcal{P}_m$ (for each UTU in the network):

$$\mathcal{P}_s = p_s(\mathbf{u},\mathbf{r},\mathcal{H}) \propto \mathcal{P}_k \prod_{m\in\{1,2,\ldots,N_m\}} \mathcal{P}_m. \tag{3.28}$$

Note that the proportionality sign $\propto$ is used, as the individual factors are likelihood functions; a normalising constant is required to ensure that $\mathcal{P}_s$, $\mathcal{P}_t$, and $\mathcal{P}_m$ are valid probability distributions. Equation (3.28) completes the mathematical description of $\mathcal{P}_s$. Now that $\mathcal{G}_s$ and $\mathcal{P}_s$ are fully defined, belief propagation can be applied to infer $\mathcal{P}_s$ efficiently and find the estimates in (3.15).

3.6 CHAPTER SUMMARY

The objectives of the proposed algorithm were identified, and a statistical model forming the foundation of the algorithm was developed in this chapter. The system model for the sensing process of a relay-based CRN was first developed. Probabilistic inference was proposed as a sensing technique to perform joint estimations on the desired variables. Belief propagation was introduced as an efficient technique to infer graphical models, especially in comparison to the brute-force alternative. This required a graphical model representing the conditional relationships of the system to be developed. The conditional relationships between the variables of the system model were determined, and various candidate graphical models were explored. Factor graphs were identified as a sufficient model to capture the behaviour of the system variables. Once the graphical model had been developed, the mathematical form of the statistical model was obtained. The application of belief propagation, along with the implementation of the subsequent marginal computations on a relay-based network, are considered in Chapter 4.

CHAPTER 4 IMPLEMENTATION OF THE SECURE SENSING ALGORITHM

4.1 CHAPTER OVERVIEW

The statistical model developed in Chapter 3 is used to formulate a real-time spectrum-sensing algorithm in this chapter, which is divided into two major sections. Section 4.2 entails the efficient inference of the statistical model by the application of belief propagation, which results in analytical expressions for the computation of the desired marginals. Section 4.3 contains the formulation of the proposed algorithm, which involves the distributed computation of the marginal expressions.

Section 4.2 introduces the rules for belief propagation by considering an arbitrary factor graph first. Once the general rules have been obtained, they are applied to the graphical model developed in Chapter 3. The application of the rules of belief propagation yields a set of sub-computations called "messages", which can be combined to form the desired marginal distributions (and in turn the joint estimates).

The proposed algorithm determines the desired joint estimations by computing the messages obtained in a distributed manner across the relays of the relay-based CRN. This is formulated in Section 4.3 by allowing the TUs to expose only specific messages to neighbouring TUs. The choice of which messages to expose is facilitated by considering the local and global utility of these messages to data fusion. In this context, local utility refers to knowledge gained about a specific device in the CRN (i.e. the UTU m), and global utility refers to knowledge gained concerning the entire network (i.e. the current PU state). Only messages providing global utility are chosen to be exchanged among the relays. The transfer of this information is realised using sequential hops across the relay nodes.

Since historical information needs to be accumulated over time, phases are introduced to the proposed algorithm in Section 4.3.4. A learning phase and execution phase are used to divide the functionality of the algorithm to allow for adaptation to time-varying conditions. Finally, the complexity of the algorithm is analysed in Section 4.3.5.

4.2 APPLICATION OF BELIEF PROPAGATION TO THE GRAPHICAL MODEL

Expressions for the marginal probabilities of the sensing model $\mathcal{P}_s$ in (3.14) are efficiently determined by the application of belief propagation to the graphical model $\mathcal{G}_s$. Belief propagation entails passing interim computations called messages across the edges of $\mathcal{G}_s$; these messages and expressions are obtained in this section.

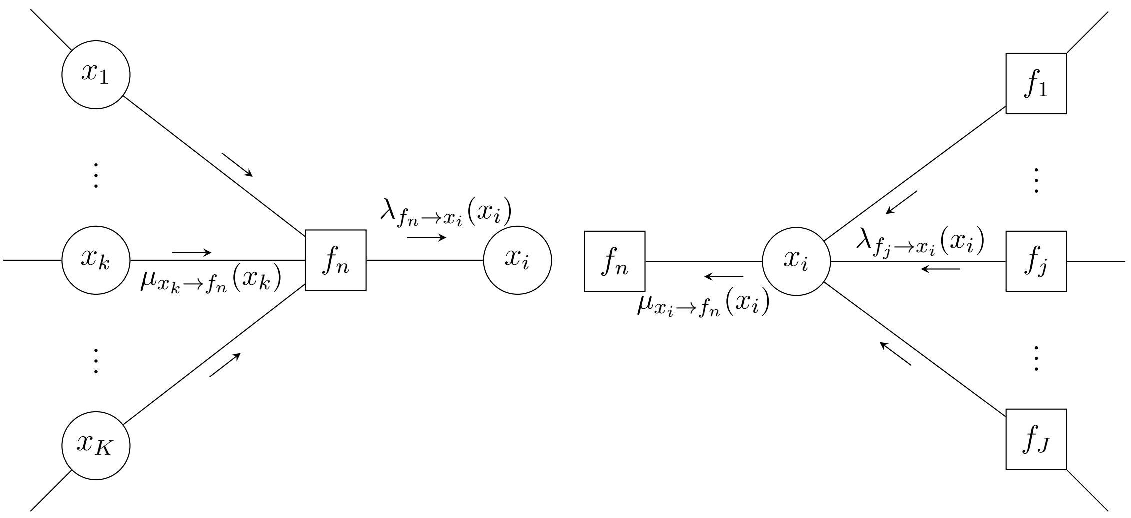
4.2.1 Belief propagation on factor graphs

Consider an arbitrary joint distribution $\mathcal{P}$ – assume that $\mathcal{P}$ can be expressed as a product of functions such that an equivalent factor graph $\mathcal{G}$ exists. The distribution $\mathcal{P}$ is a function of V variables in $\mathbf{X} = \{x_1, x_2, \cdots, x_V\}$; an arbitrary variable in the set is denoted by $x_i \in \mathbf{X}$, with $1 \leq i \leq V$. The mathematical expression of $\mathcal{P}$ is defined by the product of L factor functions $\mathbf{F} = \{f_1, f_2, \cdots, f_L\}$; an arbitrary factor in the set is denoted by f_n, with $1 \leq n \leq L$. Each factor f_n is a function of a subset of variables from $\mathbf{X}$; the subset associated with f_n is denoted by $\mathbf{x}_n \subset \mathbf{X}$. Similarly, the factors that are a function of a variable x_i form a subset $\mathbf{f}_i \subset \mathbf{F}$. Thus, the factor graph $\mathcal{G}$ possesses a variable set $\mathbf{X}$ and a factor set $\mathbf{F} = \{f_1(\mathbf{x}_1), f_2(\mathbf{x}_2), \cdots, f_L(\mathbf{x}_L)\}$.

Belief propagation reduces the overall computational burden of marginalisation by reusing interim computations for future ones; this concept is termed “dynamic programming”. Furthermore, application of belief propagation results in the simultaneous computation of all the marginals in $\mathcal{P}$. The following two types of interim computations exist:

- the contribution from a factor f_n to the marginalisation of an associated variable x_i if $x_i \in \mathbf{x}_n$, and
- the amalgamation of all the marginalisation contributions from the factors $\mathbf{f}_i$ associated with a specific variable x_i.

These computations are called “messages” and are expressed as the information transferred from a factor to a variable (in the former case) and a variable to a factor (in the latter case). A message passed from a factor f_n to a variable x_i is denoted by $\lambda_{f_n \rightarrow x_i}$, and a message passed from a variable x_i to a factor f_n is denoted by $\mu_{x_i \to f_n}$. Both types of messages represent probability distributions, and are functions of the variable node with which they are associated. Figure 4.1 shows two instances of message passing in the factor graph $\mathcal{G}$. Figure 4.1(a) is focused on a factor $f_n(\mathbf{x}_n)$ passing a message to x_i, with $\mathbf{x}_n = \{x_1, x_i, \cdots, x_k, \cdots, x_K\}$, $k \in \{1, 2, \ldots, K\}, K \leq V$, and $k \neq i$. Figure 4.1(b) is focused on a variable x_i passing a message to $f_n(\mathbf{x}_n)$, with $\mathbf{f}_i = \{f_1, f_n, \cdots, f_j, \cdots, f_J\}$, $j \in \{1, 2, \ldots, J\}$, $J \leq L$, and $j \neq n$.



(a) Factor-to-variable message passing. **(b)** Variable-to-factor message passing.

Figure 4.1. Message passing along the factor graph $\mathcal{G}$.

The mathematical forms of the messages are obtained in terms of the notation and structure of the factor graph fragments of Figure 4.1. The messages are determined iteratively; a message from the next iteration is dependent on the messages of the current iteration. The iteration index is given by l, and the total number of iterations is denoted by η. Note that if any of the variables in $\mathbf{X}$ are continuous, the summations in the message computations change to integrations over the domains of the continuous variables.

4.2.1.1 Factor-to-variable messages

The factor-to-variable message $\lambda_{f_n \to x_i}(x_i)$ is the marginalised product of the factor f_n and all variable-to-factor messages entering f_n, with the exception of $\mu_{x_i \to f_n}(x_i)$ (i.e. the message coming from x_i). The summation/integration is performed over all variables except x_i. The expression for $\lambda_{f_n \to x_i}(x_i)$ is obtained in the context of the graph fragment in Figure 4.1(a):

$$\lambda^{(l+1)}_{f_n \to x_i}(x_i) \propto \sum_{\mathbf{x}_n \setminus x_i} f_n(\mathbf{x}_n) \prod_{\substack{x_k \in \mathbf{x}_n \\ k \neq i}} \mu^{(l)}_{x_k \to f_n}(x_k). \tag{4.1}$$

Equation (4.1) passes the marginal computation of f_n and all marginal computations associated with the clique of f_n (besides x_i) to the variable x_i.

4.2.1.2 Variable-to-factor messages

The mathematical form of the variable-to-factor message $\mu_{x_i \to f_n}(x_i)$ of the next iteration is obtained by taking the product of all factor-to-variable messages entering the variable x_i in the current iteration, with the exception of $\lambda_{f_n \to x_i}$ (i.e. the message from f_n). The expression for $\mu_{x_i \to f_n}(x_i)$ is obtained in the context of the graph fragment in Figure 4.1(b).

$$\mu^{(l+1)}_{x_i \to f_n}(x_i) \propto \prod_{\substack{f_j \in \mathbf{f}_i \\ j \neq n}} \lambda^{(l)}_{f_j \to x_i}(x_i). \tag{4.2}$$

This message combines marginalisation information from the factors associated with x_i (besides factor f_n), and passes this to factor f_n.

4.2.1.3 Marginal distributions

The marginal distributions associated with a specific variable are determined by computing the product of the factor-to-variable messages entering that variable:

$$p(x_i) \approx b^{(l)}(x_i) \propto \prod_{f_j \in \mathbf{f}_i} \lambda^{(l)}_{f_j \to x_i}(x_i). \tag{4.3}$$

The proportionality sign $\propto$ is used to highlight that all messages must be normalised prior to computation. When the factor graph has no cycles/loops, the value of $b(x_i)$ can be guaranteed to converge to $p(x_i)$ (given enough iterations) [50]. All messages are computed in parallel, but acyclic graphs can commence message passing on an arbitrary root node to simplify the computation.

4.2.2 Applying the message-passing rules to $\mathcal{G}_s$

The message-passing rules of (4.1) and (4.2), along with the expression for the beliefs in (4.3), can now be applied to the graphical model $\mathcal{G}_s$ developed in Chapter 3 to obtain the marginals of (3.14). Detailed portions of Figure 3.9, with focus on the TUs (i.e. the subgraph $\mathcal{G}_k$) and UTUs (i.e. the subgraphs $\mathcal{G}_m$), are shown in Figures 4.2 and 4.3 respectively; these figures also show the messages passed along the edges of the graphs.

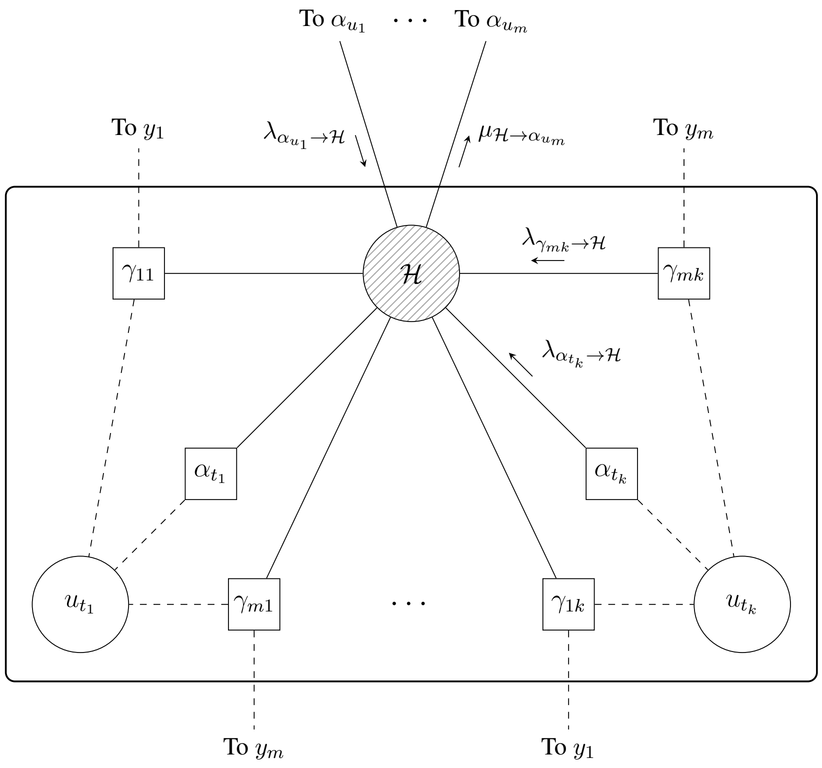

To α_{u_1} $\cdots$ To α_{u_m}
To y_1
$\lambda_{\alpha_{u_1}\to\mathcal{H}}$
$\mu_{\mathcal{H}\to\alpha_{u_m}}$
To y_m
γ_{11}
$\mathcal{H}$
$\lambda_{\gamma_{mk}\to\mathcal{H}}$
γ_{mk}
$\lambda_{\alpha_{t_k}\to\mathcal{H}}$
α_{t_1}
α_{t_k}
u_{t_1}
γ_{m1}
$\cdots$
γ_{1k}
u_{t_k}
To y_m
To y_1

Figure 4.2. Messages passed along the edges of subgraph $\mathcal{G}_k$.

Using the rules in (4.1), along with the factor graph fragments in Figures 4.2 and 4.3, results in the following set of expressions for the factor-to-variable messages:

$$\begin{aligned}
\lambda^{(l+1)}_{\alpha_{u_m}\to u_m}(u_m) &\propto \sum_{\mathcal{H}} \alpha_{u_m}(\mathcal{H}, u_m)\mu^{(l)}_{\mathcal{H}\to\alpha_{u_m}}(\mathcal{H}),\\
\lambda^{(l+1)}_{\alpha_{u_m}\to\mathcal{H}}(\mathcal{H}) &\propto \sum_{u_m} \alpha_{u_m}(\mathcal{H}, u_m)\mu^{(l)}_{u_m\to\alpha_{u_m}}(u_m),\\
\lambda^{(l+1)}_{\alpha_{t_k}\to\mathcal{H}}(\mathcal{H}) &\propto \alpha_{t_k}(\mathcal{H}, u_k),\\
\lambda^{(l+1)}_{\delta_{a_m}\to r_m}(r_m) &\propto \sum_{u_m} \delta_{a_m}(u_m, r_m, y_m)\mu^{(l)}_{u_m\to\delta_{a_m}}(u_m),\\
\lambda^{(l+1)}_{\gamma_{mk}\to\mathcal{H}}(\mathcal{H}) &\propto \gamma_{mk}(u_k, y_m, \mathcal{H}),\\
\lambda^{(l+1)}_{\delta_{a_m}\to u_m}(u_m) &\propto \int_{-1}^{1} \delta_{a_m}(u_m, r_m, y_m)\mu^{(l)}_{r_m\to\delta_{a_m}}(r_m)dr_m,\\
\lambda^{(l+1)}_{\delta^i_{b_m}\to r_m}(r_m) &\propto \delta^i_{b_m}(\hat{\mathcal{H}}^{(i)}, r_m, y^{(i)}_m).
\end{aligned} \tag{4.4}$$

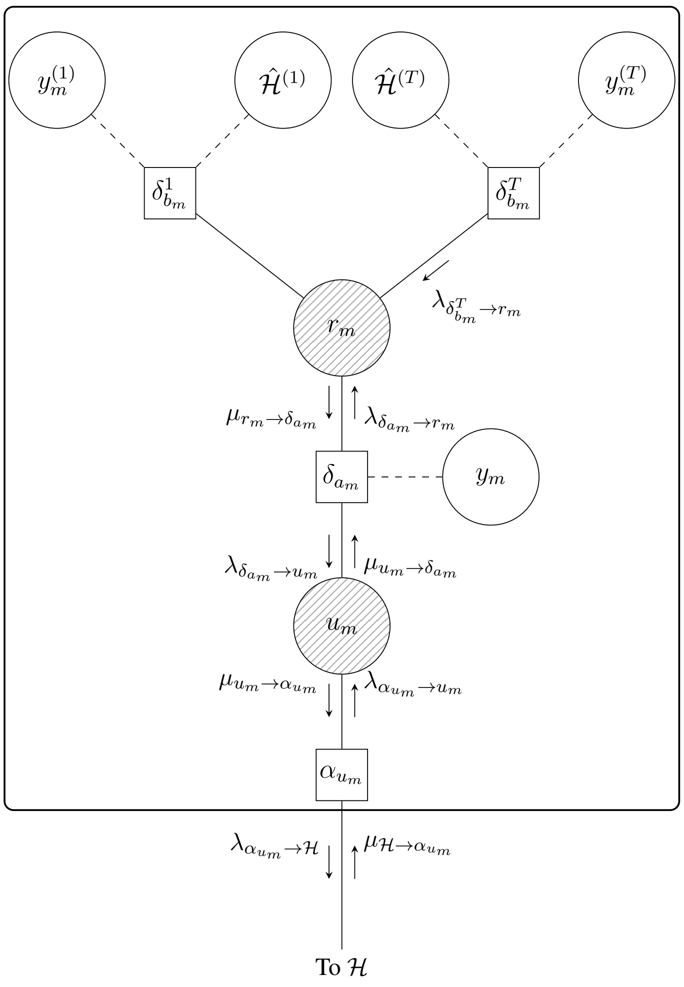

$y_m^{(1)}$
$\hat{\mathcal{H}}^{(1)}$
$\hat{\mathcal{H}}^{(T)}$
$y_m^{(T)}$
$\delta_{b_m}^1$
$\delta_{b_m}^T$
$\lambda_{\delta_{b_m}^T \to r_m}$
r_m
$\mu_{r_m \to \delta_{a_m}}$
$\lambda_{\delta_{a_m} \to r_m}$
δ_{a_m}
y_m
$\lambda_{\delta_{a_m} \to u_m}$
$\mu_{u_m \to \delta_{a_m}}$
u_m
$\mu_{u_m \to \alpha_{u_m}}$
$\lambda_{\alpha_{u_m} \to u_m}$
α_{u_m}
$\lambda_{\alpha_{u_m} \to \mathcal{H}}$
$\mu_{\mathcal{H} \to \alpha_{u_m}}$
To $\mathcal{H}$

Figure 4.3. Messages passed along the edges of subgraph $\mathcal{G}_m$.

Similarly, application of the rules in (4.2), along with the factor graph fragments in Figures 4.2 and 4.3, results in the following set of expressions for the variable-to-factor messages:

$$\begin{aligned}
\mu^{(l+1)}_{\mathcal{H}\to\alpha_{u_m}}(\mathcal{H}) &\propto \prod_{\substack{j=1\\ j\neq m}}^{N_u} \lambda^{(l)}_{\alpha_{u_j}\to\mathcal{H}}(\mathcal{H}) \times \prod_{j=1}^{N_k} \lambda^{(l)}_{\alpha_{t_j}\to\mathcal{H}}(\mathcal{H}),\\
\mu^{(l+1)}_{u_m\to\alpha_{u_m}}(u_m) &\propto \lambda^{(l)}_{\delta_{a_m}\to u_m}(u_m),\\
\mu^{(l+1)}_{u_m\to\delta_{a_m}}(u_m) &\propto \lambda^{(l)}_{\alpha_{u_m}\to u_m}(u_m),\\
\mu^{(l+1)}_{r_m\to\delta_{a_m}}(r_m) &\propto \prod_{i=1}^{T} \lambda^{(l)}_{\delta^i_{b_m}\to r_m}(r_m).
\end{aligned} \tag{4.5}$$

Each message must be initialised such that the probabilities associated with each state are equal and sum to unity. If a variable x_i has L states, an initial state s of the message entering/exiting each variable/factor is

$$\begin{aligned}\mu^{(0)}_{x_i \rightarrow f_n(\mathbf{x}_n)}(x_i = s) &= \frac{1}{L}, \text{ and} \\ \lambda^{(0)}_{f_n(\mathbf{x}_n) \rightarrow x_i}(x_i = s) &= \frac{1}{L}.\end{aligned} \tag{4.6}$$

Note that only the messages between an unknown variable and factor need to be computed iteratively, as messages between known variables and factors remain the same over the iterations. The beliefs associated with the unknown variables $\mathcal{H}$, u_m, and r_m can be computed by applying (4.3):

$$b^{(l)}(\mathcal{H}) \propto \prod_{k=1}^{N_k} \lambda^{(l)}_{\alpha_{t_k} \rightarrow \mathcal{H}}(\mathcal{H}) \times \prod_{m=1}^{N_m} \lambda^{(l)}_{\alpha_{u_m} \rightarrow \mathcal{H}}(\mathcal{H}) \times \prod_{k=1}^{N_k} \prod_{m=1}^{N_m} \lambda^{(l)}_{\gamma_{mk} \rightarrow \mathcal{H}}(\mathcal{H}), \tag{4.7}$$

$$b^{(l)}(u_m) \propto \lambda^{(l)}_{\alpha_{u_m} \rightarrow u_m}(u_m) \times \lambda^{(l)}_{\delta_a \rightarrow u_m}(u_m), \tag{4.8}$$

$$b^{(l)}(r_m) \propto \lambda^{(l)}_{\delta_{am} \rightarrow r_m}(r_m) \times \prod_{i=1}^{T} \lambda^{(l)}_{\delta^i_{bm} \rightarrow r_m}(r_m). \tag{4.9}$$

The factor-to-variable and variable-to-factor messages in (4.4) and (4.5), along with the beliefs of (4.7), (4.8), and (4.9), are computed by the TUs; Section 4.3 provides details on how this is achieved. The above beliefs are proportional (and not equal) to the product of the factors as the beliefs are approximates of probabilities, and must sum to unity. A normalising constant can be applied after the computation of the products to ensure that the beliefs sum to unity.

4.3 FORMULATING THE PROPOSED ALGORITHM

This section details the formulation of the proposed algorithm, which performs marginal inference on $\mathcal{P}_s$ (expressed in (4.7), (4.8), and (4.9)) using the relay-based network model of Section 3.2. This is done by letting the TUs expose only specific messages from (4.4) and (4.5) to neighbouring TUs. The messages that are selected for exposure are determined based on their utility to local (UTU m maliciousness) and global (PU state detection) estimations. A TU sends information to a neighbouring TU over a single hop; the final beliefs are recursively computed with each hop through the network.

4.3.1 Hop-based implementation

A sequential hop-based protocol is used to facilitate this implementation. A hop occurs when TU k passes information $M_{k,k+1}$ to TU $k+1$ (or *vice versa*, with $M_{k-1,k}$). The aim of the hop-based protocol is to supply every TU with only a subset of the messages derived in Section 4.2 (i.e. (4.4) and (4.5)), such that each TU is able to compute the belief $b(\mathcal{H})$ independently. The computation of the beliefs $b(u_m)$ and $b(r_m)$ associated with UTU m is done locally by the TU that is connected to the UTU. It is assumed that only one path is taken from TU 1 to TU N_k and *vice versa*. The order of the hops is represented as a sequence $\mathcal{C}$ over the enumeration of the TUs; the TUs are cycled in an alternating ascending-descending order based on the TU enumeration:

$$\mathcal{C} = \{\overbrace{1, 2, \cdots, N_k - 1, N_k}^{l=0}, \underbrace{N_k - 1, \cdots, 2, 1}_{l=1}, \overbrace{2, \cdots, N_k - 1, N_k}^{l=2}, N_k - 1, \cdots\}, \tag{4.10}$$

where an element of $\mathcal{C}$ corresponds to the index k of a TU. A single iteration of belief propagation (i.e. l) corresponds to a single ascending pass (i.e. information is transferred across all TUs from TU 1 to TU N_k) or descending pass (i.e. information is transferred across all TUs from TU N_k to TU 1). The beliefs are fully computed once $M_{k,k+1}^{(l)}$ or $M_{k-1,k}^{(l)}$ has been propagated to either TU 1 or TU N_k.

4.3.2 Grouping the messages based on utility

The beliefs $b(u_m)$, $b(r_m)$, and $b(\mathcal{H})$ can be grouped based on the local or global utility provided by their knowledge. The belief $b(\mathcal{H})$ provides global utility, as every SU in the CRN must be aware of its state. Since $b(u_m)$ and $b(r_m)$ are specific to a single UTU m, they provide local utility to sensing. Note that the subgraph $\mathcal{G}_k$ in Figure 4.2 is linked to global utility, as this graph contains the variable node $\mathcal{H}$. Similarly, the subgraph $\mathcal{G}_m$ in Figure 4.3 is linked to local utility, as this graph contains the variables u_m and r_m. The messages can be partitioned into the following groups, based on the subgraphs and variables with which they are associated:

1. **Group 1** – All messages that are enclosed in $\mathcal{G}_m$ are responsible for the inference of the variables associated with the UTU m. These messages provide local utility and are computed and used internally by the TUs. No information exchange between the TUs is required for their computation.
2. **Group 2** – Messages that are directed towards the variable $\mathcal{H}$ form this group. These messages amalgamate all local fusion results to provide a global prediction on the state of the PU. The messages of this group form the information that is exchanged between TUs.
3. **Group 3** – Messages that are directed away from the variable $\mathcal{H}$ form this group. These messages provide local feedback by updating the group 1 messages based on global fusion results. Messages in this group require information from neighbouring TUs (i.e. group 2

messages) to update the local inferences of UTU m. Only one class of messages ($\mu_{\mathcal{H}\to\alpha_{u_m}}$) exists in $\mathcal{G}_s$ that qualifies for this criterion.

Table 4.1 shows the messages that belong to these groups. The second column refers to the SU type that is the subject of the messages. A short description of the utility of the messages to sensing and detection is also given.

Table 4.1. Grouping of the messages in (4.4) and (4.5).

Group	Subject	Message	Utility
1	UTU m	$\mu_{u_m\to\alpha_{u_m}}$	Contains the estimation of the corrected UTU measurement in the form of a probability distribution over u_m.
		$\mu_{u_m\to\delta_{a_m}}$	Transfers the adjusted feedback measurement of $\lambda_{\alpha_{u_m}\to u_m}$ to δ_{a_m} in order to update the estimate of r_m.
		$\mu_{r_m\to\delta_{a_m}}$	Combines UTU maliciousness estimates over all T historical sensing periods.
		$\lambda_{\delta_{a_m}\to u_m}$	Adjusts the UTU m's report to discern the true measurement. The adjustment is based on the historical performance of the UTU and the current sensing result.
		$\lambda_{\delta_{a_m}\to r_m}$	Adjusts the UTU maliciousness r_m based on the sensing results from other SUs using the adjusted feedback measurement from $\mu_{u_m\to\delta_{a_m}}$.
		$\lambda_{\delta^i_{b_m}\to r_m}$	Updates the UTU m's maliciousness based on the predictions and reports of time $t = i \leq T$.
		$\lambda_{\alpha_{u_m}\to u_m}$	Adjusts and converts the feedback measurement from other SUs to a distribution over u_m.
2	UTU m	$\lambda_{\alpha_{u_m}\to\mathcal{H}}$	Converts the corrected UTU measurement to a probability distribution over $\mathcal{H}$. This is sent to the neighbouring TUs for fusion.
	TU k	$\lambda_{\alpha_{t_k}\to\mathcal{H}}$	Contribution of the TU k's measurement to data fusion. The measurement is converted to a distribution over $\mathcal{H}$.
		$\lambda_{\gamma_{mk}\to\mathcal{H}}$	Conversion of the correlation between y_m and u_{t_k} to a distribution over $\mathcal{H}$; this is used directly for data fusion.
3	UTU m	$\mu_{\mathcal{H}\to\alpha_{u_m}}$	Feedback measurement to a UTU m's variables based on the global consensus of the likely state of $\mathcal{H}$.

4.3.3 Data exchange in the decentralised network

Two of the three beliefs (i.e. $b(u_m)$ in (4.8) and $b(r_m)$ in (4.9)) can be determined using group 1 messages. As a result, these beliefs only require a single local TU for computation. In contrast, $b(\mathcal{H})$ in (4.7) requires the group 2 messages, spread over different TUs, for computation. The TUs must broadcast their group 2 messages to other TUs in order to fully determine $b(\mathcal{H})$.

In an ascending pass of $\mathcal{C}$, a TU k sends information to the next TU $k+1$ in the form of $M_{k,k+1}^{(l)}$. The information is represented by $M_{k,k-1}^{(l)}$ during a descending pass in $\mathcal{C}$. The content of $M_{k,k+1}^{(l)}$ or $M_{k,k-1}^{(l)}$ changes with the belief propagation iteration index l and can be composed of the following information:

1. At the beginning of the fusion process (i.e. $l = 0$), the messages are initialised and all UTUs send their occupancy reports to their local TUs. The reports are stored and transferred cumulatively from TU k to TU $k+1$. The information transferred in this phase is

$$M_{k,k+1}^{(l=0)} = \left[\mathbf{y}_{\mathcal{N}_1}, \cdots, \mathbf{y}_{\mathcal{N}_k}\right], \tag{4.11}$$

where $\mathcal{N}_k$ is the neighbour set of UTUs connected to TU k. All UTU reports are available to the final TU when (4.11) is applied over all TUs. Since iteration $l = 0$ always begins on an ascending pass over the TUs, some UTU reports must be sent back to the other TUs in iteration $l = 1$. Figure 4.4 illustrates the information transfer between TUs at iteration $l = 0$.

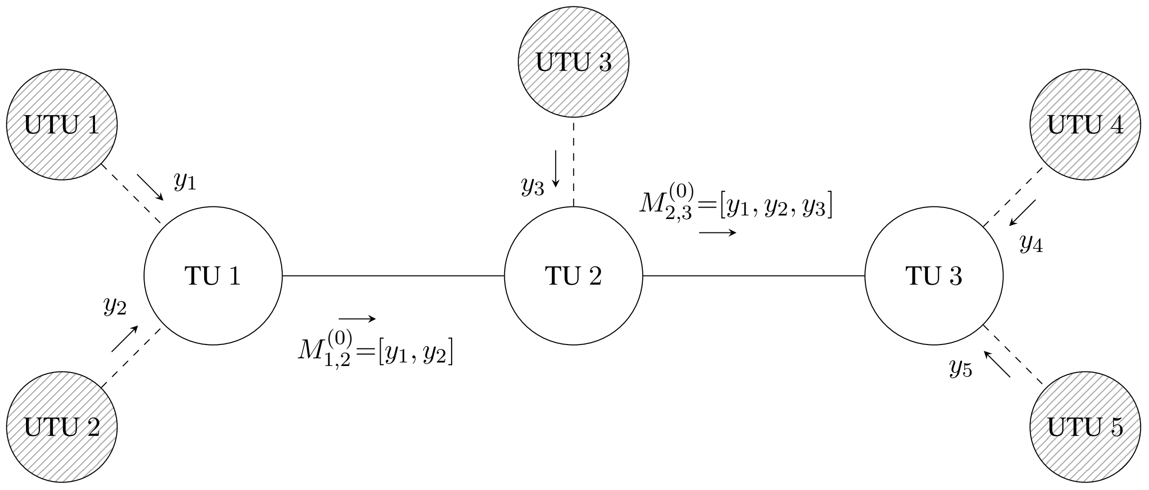


Figure 4.4. Transfer of information between TUs at $l = 0$. Note that y_4 and y_5 must be transferred back to TUs 1 and 2 during the descending pass in $l = 1$.

2. The main fusion phase occurs when $1 < l \leq \eta$ and the beliefs need to be determined, with all UTU reports made (or being made) available to the TUs. A TU k sends $M^{(l)}_{k,k+1}$ to TU $k+1$ if l is even (ascending pass), or $M^{(l)}_{k,k-1}$ to TU $k-1$ if l is odd (descending pass). Only a single TU k is needed to compute the beliefs $b(u_m)$ and $b(r_m)$ using the group 1 and 2 messages from the previous iteration. In contrast, computation of the messages associated with $b(\mathcal{H})$ in (4.7) requires information from all TUs. With each hop, the product of the group 3 messages associated with TU k is computed and broadcast to a neighbouring TU. The neighbouring TU multiplies this product with its own product of group 3 messages. This computation is propagated over all TUs. The content of $M^{(l)}_{k,k+1}$ becomes the recursive product of the contributions to $b(\mathcal{H})$ from all TUs up to k:

$$M^{(l)}_{k,k+1} = M^{(l)}_{k-1,k} \times \lambda^{(l)}_{\alpha_{t_k}\to\mathcal{H}}(\mathcal{H}) \times \prod_{m\in\mathcal{N}_k} \lambda^{(l)}_{\alpha_{u_m}\to\mathcal{H}}(\mathcal{H}) \times \prod_{m\in\{1,2,\ldots,N_m\}} \lambda^{(l)}_{\gamma_{mk}\to\mathcal{H}}(\mathcal{H}), \tag{4.12}$$

where $M^{(l)}_{k,k+1}$ changes to $M^{(l)}_{k,k-1}$, and $M^{(l)}_{k-1,k}$ changes to $M^{(l)}_{k+1,k}$ for a descending pass. Note that in this case, the information $M^{(l)}_{k,k+1}$ is a function of $\mathcal{H}$, which only has two states. Thus, only two floating-point numbers are required to be transferred between TUs for this phase. Application of (4.12) over all TUs in the network is equivalent to computing (4.7), with $b(\mathcal{H})$ yielded by $M^{(l)}_{1,2}$ or $M^{(l)}_{N_k-1,N_k}$ for an ascending or descending pass respectively. Figures 4.5 and 4.6 illustrate the information transfer between TUs at iterations $l = 1$ (descending pass) and $l = 2$ (ascending pass) respectively. The transfer of UTU reports during the descending pass is omitted from Figure 4.5 for clarity.

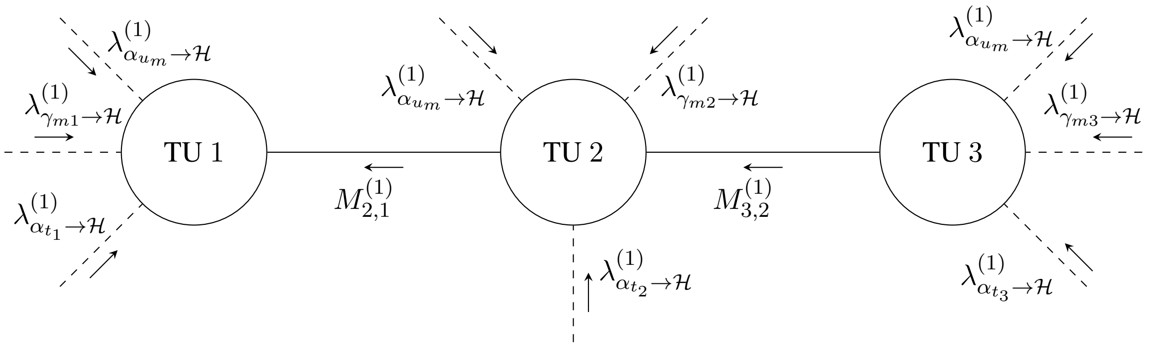
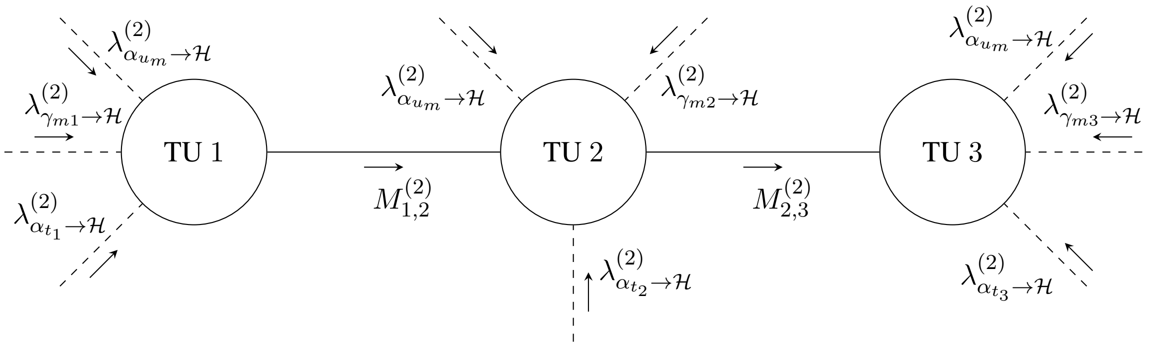

$\lambda^{(1)}_{\alpha_{u_m}\to\mathcal{H}}$
$\lambda^{(1)}_{\gamma_{m1}\to\mathcal{H}}$
TU 1
$\lambda^{(1)}_{\alpha_{t_1}\to\mathcal{H}}$
$\lambda^{(1)}_{\alpha_{u_m}\to\mathcal{H}}$
$M^{(1)}_{2,1}$
TU 2
$\lambda^{(1)}_{\gamma_{m2}\to\mathcal{H}}$
$\lambda^{(1)}_{\alpha_{t_2}\to\mathcal{H}}$
$M^{(1)}_{3,2}$
$\lambda^{(1)}_{\alpha_{u_m}\to\mathcal{H}}$
TU 3
$\lambda^{(1)}_{\gamma_{m3}\to\mathcal{H}}$
$\lambda^{(1)}_{\alpha_{t_3}\to\mathcal{H}}$

Figure 4.5. Transfer of information between TUs at $l = 1$.



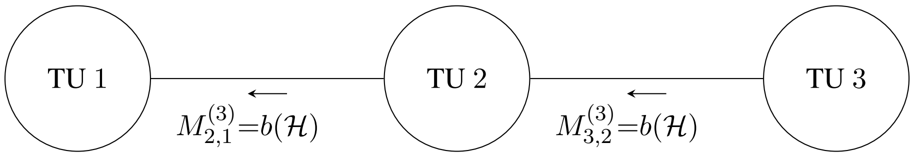
TU 1
TU 2
TU 3
$\lambda^{(2)}_{\alpha_{u_m}\to\mathcal{H}}$
$\lambda^{(2)}_{\gamma_{m1}\to\mathcal{H}}$
$\lambda^{(2)}_{\alpha_{t_1}\to\mathcal{H}}$
$\lambda^{(2)}_{\alpha_{u_m}\to\mathcal{H}}$
$M^{(2)}_{1,2}$
$\lambda^{(2)}_{\gamma_{m2}\to\mathcal{H}}$
$\lambda^{(2)}_{\alpha_{t_2}\to\mathcal{H}}$
$M^{(2)}_{2,3}$
$\lambda^{(2)}_{\alpha_{u_m}\to\mathcal{H}}$
$\lambda^{(2)}_{\gamma_{m3}\to\mathcal{H}}$
$\lambda^{(2)}_{\alpha_{t_3}\to\mathcal{H}}$

Figure 4.6. Transfer of information between TUs at $l = 2$.

3. When $l = \eta$, and the final beliefs have been determined, only TU 1 or TU N_k will know the value of $b(\mathcal{H})$. The final belief must be sent to the other TUs in the network for a final pass, with $l = \eta + 1$. For an ascending pass, the information is

$$M^{(\eta+1)}_{k,k+1} = b(\mathcal{H}), \tag{4.13}$$

where $M^{(\eta+1)}_{k,k+1}$ changes to $M^{(\eta+1)}_{k,k-1}$ for a descending pass. The TUs can now independently perform the joint estimations in (3.15), allowing the CRN to start or cease broadcasting in the PU's band. The information transfer between TUs at $l = \eta + 1 = 3$ is shown in Figure 4.7.



TU 1
TU 2
TU 3
$M^{(3)}_{2,1}=b(\mathcal{H})$
$M^{(3)}_{3,2}=b(\mathcal{H})$

Figure 4.7. Transfer of information between TUs at $l = \eta + 1 = 3$.

4.3.4 Phases of spectrum sensing

At $t = 1$, there is not enough prior information to use the distribution $\mathcal{P}_s$ meaningfully. The quantities $\hat{\boldsymbol{\mathcal{H}}}^{(T)}$ and $\mathbf{Y}^{(T)}$ require a few time steps (denoted by T_l) of learning to provide the TUs with sufficient information. As a result, the proposed algorithm is broken up into the following two phases:

1. The learning phase – from $t = 1$ to $t = T_l$, (4.11) is used to accumulate all the sensing reports for each TU and UTU in the form of $\mathbf{Y}^{(T_l)}$. A record of all the PU state predictions until $t = T_l$, in the form of $\hat{\boldsymbol{\mathcal{H}}}^{(T_l)}$, is also kept in this phase. When $t = 1$, any component of the

graphical model $\mathcal{G}_s$ requiring historical data does not contribute to the inference of the unknown probabilities. Because of the lack of data available, the prediction quality in this phase is poor. The learning phase serves to bootstrap the algorithm.

2. The execution phase – once enough SU reports are received, and a sufficient number of PU predictions are made, the algorithm is ready for sensing and MU detection. No further information needs to be acquired in order to make predictions. If the environment is time-varying, new SU reports can overwrite old ones. In this case, $\hat{\boldsymbol{\mathcal{H}}}^{(T)}$ and $\mathbf{Y}^{(T)}$ represent reports and predictions over a time window of length T_l.

A flow diagram summarising the proposed algorithm, from message passing to the phases of spectrum sensing, is given in Figure 4.8.

4.3.5 Complexity of the proposed algorithm

The proposed algorithm requires $2 + \eta$ ascending or descending passes to run: UTU reports are propagated in the first pass, η iterations of belief propagation are performed, and the beliefs are sent back to all TUs in the final pass. The complexity of the proposed algorithm depends on either N_k or N_m.

- An increase in N_k (number of TUs) affects the global complexity, and results in more hops being required for the computation of the final beliefs. Increasing N_k may result in a more accurate estimation, but this comes at the cost of increased latency, as more hops must be performed for a single pass through the TUs. Note that an increase in N_k results in more parallel computation taking place, thus having no net effect on the local complexity.
- An increase in N_m (number of UTUs) affects the local and global complexity, as a greater number of UTUs will require verification by the TUs, resulting in more group 1 messages being computed. The local computational overhead incurred by each TU increases linearly ($\mathcal{O}(N_m)$) in this case. This may increase the delay between sensing and prediction, depending on the hardware used for the TUs.

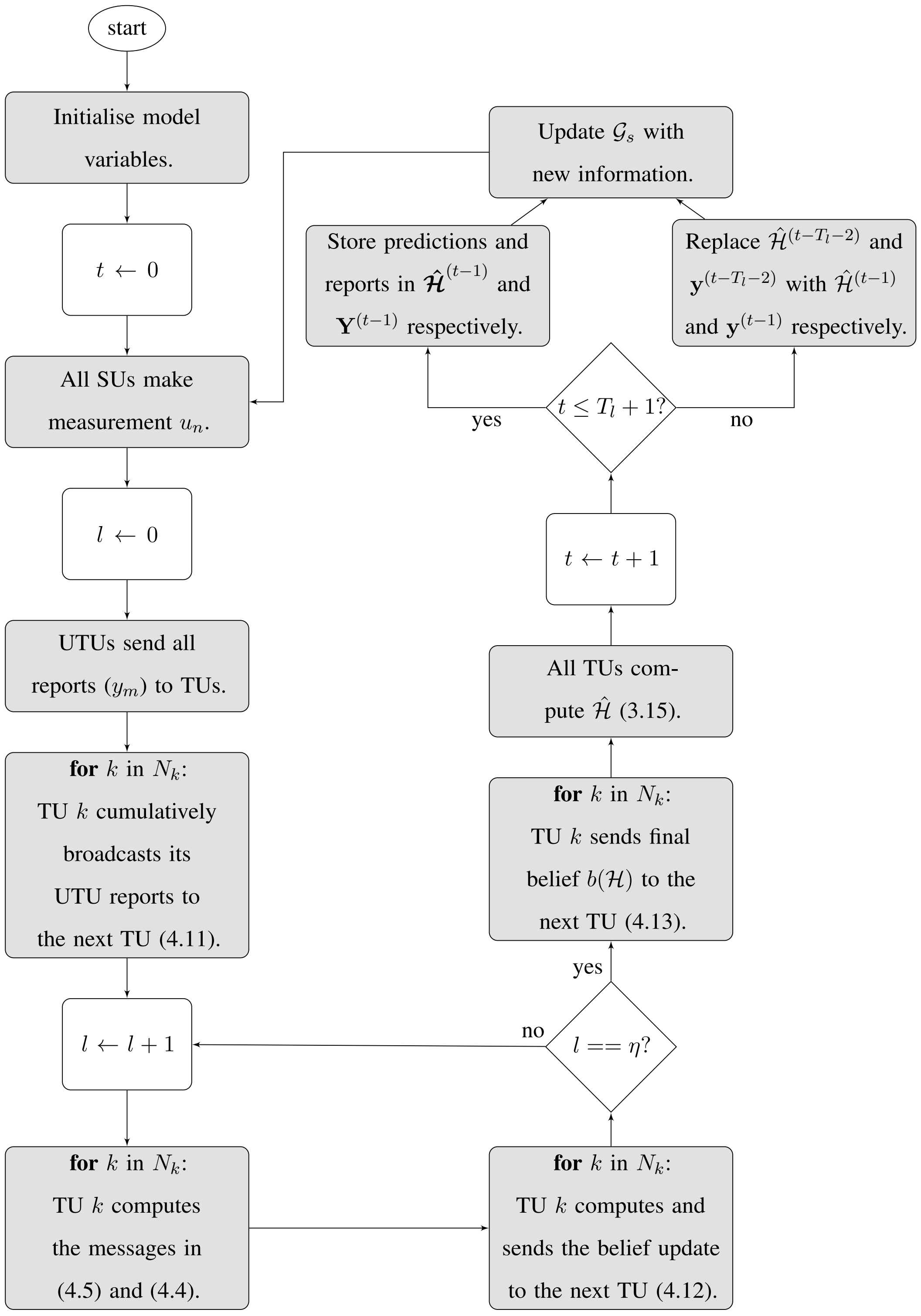

start
Initialise model variables.
$t \leftarrow 0$
All SUs make measurement u_n.
$l \leftarrow 0$
UTUs send all reports (y_m) to TUs.
for k in N_k: TU k cumulatively broadcasts its UTU reports to the next TU (4.11).
$l \leftarrow l + 1$
for k in N_k: TU k computes the messages in (4.5) and (4.4).
for k in N_k: TU k computes and sends the belief update to the next TU (4.12).
$l == \eta$?
no
yes
for k in N_k: TU k sends final belief $b(\mathcal{H})$ to the next TU (4.13).
All TUs compute $\hat{\mathcal{H}}$ (3.15).
$t \leftarrow t + 1$
$t \leq T_l + 1$?
yes
no
Store predictions and reports in $\hat{\boldsymbol{\mathcal{H}}}^{(t-1)}$ and $\mathbf{Y}^{(t-1)}$ respectively.
Replace $\hat{\mathcal{H}}^{(t-T_l-2)}$ and $\mathbf{y}^{(t-T_l-2)}$ with $\hat{\mathcal{H}}^{(t-1)}$ and $\mathbf{y}^{(t-1)}$ respectively.
Update $\mathcal{G}_s$ with new information.

Figure 4.8. Flow diagram summarising the proposed algorithm.

4.4 CHAPTER SUMMARY

In this chapter, belief propagation was used to determine the marginals associated with the joint distribution $\mathcal{P}_s$ and the graphical model $\mathcal{G}_s$ efficiently. Expressions for the messages passed along the edges of $\mathcal{G}_s$, and in turn the marginals, were determined. The proposed algorithm was formulated by establishing a protocol of information transfer where only a certain group of belief propagation messages is exchanged between the TUs. This subset of messages was determined by grouping all the messages according to their utility. Use of this protocol allows the TUs to determine all beliefs and estimations by transferring information using sequential hops across the CRN. In order to facilitate the accumulation of historical sensing information, the proposed algorithm was divided into two phases: a learning phase and an execution phase. The interaction between these two phases was summarised by the flow diagram in Figure 4.8. Finally, the complexity of the proposed algorithm with respect to the number of SUs was considered, with the computational cost for each TU scaling linearly with the number of UTUs, and the total number of hops increasing with the number of TUs.

CHAPTER 5 RESULTS AND EVALUATION

5.1 CHAPTER OVERVIEW

The performance of the proposed algorithm developed in Chapter 4 is evaluated in this section. The methodology is described first in Section 5.2, including details on the structure of the simulation platform used, the parameters selected for the characterisation of the environment, the metrics and control used to evaluate the algorithm's performance, and an explanation of the tests used to obtain the results. The results corresponding to the tests are then presented in Section 5.3, with each set of results accompanied by an interpretation. In Section 5.4, a holistic view of the algorithm's performance is obtained by summarising all the results.

5.2 METHODOLOGY

This section begins with a description of the software simulation used to obtain the results shown later in the chapter. The environment parameters, performance evaluation metrics, control algorithm, and tests used are also documented here.

5.2.1 Simulation description

The proposed algorithm in Figure 4.8 was simulated using the Python programming language. A flow diagram of the structure of the simulation is shown in Figure 5.1. The function `main.py` is used to select the input parameters for the simulation, including the size of the environment, the noise power, and the number of SUs in the CRN. The class `CrnEnvironment` is initialised and run using these parameters, and the SU positions and UTU maliciousness values are assigned. The SU measurements are sent to a class `MeasurementHandler`, which handles the buffering of the measurements for learning. A script called `TopologyGenerator` is used to generate the topology of the factor graph based on the input parameters. The factor graph topology is then sent to the `FactorGraph` class, which contains the implementation of belief propagation. The buffered measurements are also sent to the

`FactorGraph` class, and the final PU occupancy predictions are sent back to `main.py` for performance appraisal and plotting. Note that all results are obtained through Monte Carlo simulations.

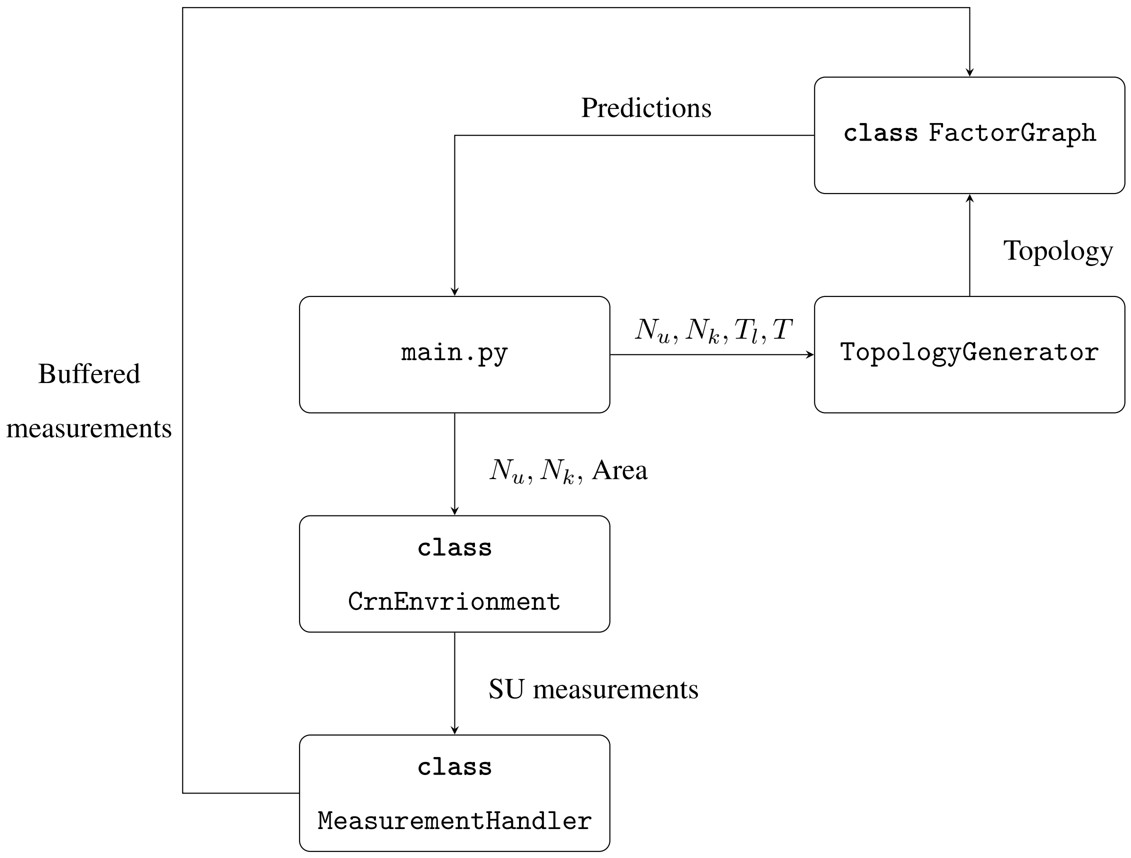


Figure 5.1. High-level diagram of the simulation and test environment.

5.2.2 Environment

The system model described in Section 3.2 was simulated in an environment characterised by the following properties (unless stated otherwise):

- An area $A_{crn} = 50 \text{ m} \times 50 \text{ m}$ with no obstacles. A simple environment was chosen for the experiment to limit the variation in placement possibilities and to improve the accuracy of the Monte Carlo simulation.
- A PU transmit power $P_t = 20 \text{ dB}$ at the centre of the area, with a carrier frequency $f_c = 150 \text{ MHz}$.
- A noise power of -10 dB, which was assumed to be known by the SUs prior to fusion. The desired false alarm rate of (3.2) was set to $\epsilon = 0.1$.
- Transmitter and receiver antennas with $G_t = 12 \text{ dBi}$ and $G_r = 2.5 \text{ dBi}$.

- A free space path loss model, with the relationship between the transmission power P_t and received power P_r as follows:

$$P_r = G_t G_r \left(\frac{\lambda}{4\pi d} \right), \tag{5.1}$$

with G_t and G_r representing the transmitter and receiver antenna gains respectively. The distance between the antennas is given by d. The wavelength λ corresponds to the PU's carrier frequency f_c.

- The false alarm and missed detection probabilities for factors α_{u_m} and α_{t_k} in Tables 3.1 and 3.2 were assumed to be $p_{fa}^{(n)} = 0.01$ and $p_{md}^{(n)} = 0.99$. Note that these assumptions are not truly accurate, as the true error rates would vary depending on the relative position of the SUs and the PU. The error in these assumptions is mitigated by the use of the γ_{mk} factors, which adaptively change over the learning phase, accounting for the variation in sensing performance among the SUs without requiring location information.

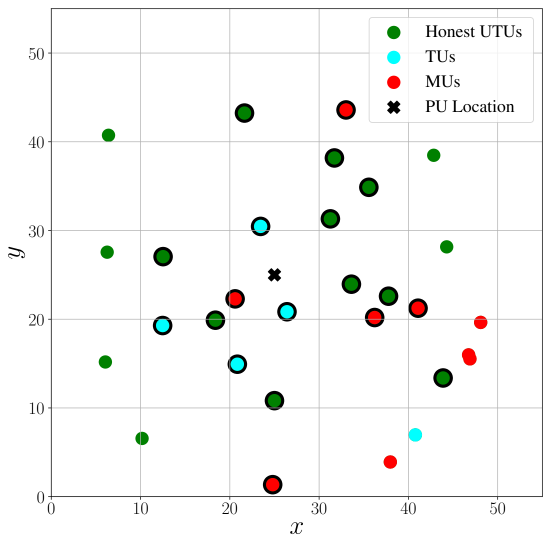


Figure 5.2. Example of a CRN placed in a 50 m × 50 m area. Nodes with black outlines measure $u_n = 1$.

An example of a CRN in a generated simulation environment is shown in Figure 5.2, where $N_{su} = 30$. The location of the PU, placed in the centre of the area, is assumed to be unknown by all SUs. The SUs were also unaware of the distances between neighbouring SUs. When the PU was broadcasting in the given scenario, the outlined nodes were likely to report PU occupancies ($u_n = 1$), while nodes without outlines were likely to report PU vacancies ($u_n = 0$). Note that, owing to the cause-agnostic nature of the proposed algorithm, the sensing accuracy is solely dependent on the quality of the received reports, and not on the specific details of the environmental model used.

5.2.3 Performance metrics

The spectrum-sensing performance of the proposed algorithm was evaluated using one of the following metrics:

- Global false alarm rate p_{fa} – a measure of how likely an algorithm is to detect the presence of the PU incorrectly when it is not broadcasting. This is determined by counting the total number of false alarms and averaging the total over the number of times the PU does not broadcast:

$$\begin{aligned} p_{fa} &= \frac{\sum_{t=1}^{T} e_{fa}^{(t)}}{T_{fa}}, \\ e_{fa}^{(t)} &= \begin{cases} 1, & \text{if } \hat{\mathcal{H}}^{(t)} \neq \mathcal{H}^{(t)} = 0 \\ 0, & \text{otherwise} \end{cases}, \end{aligned} \tag{5.2}$$

  where T_{fa} is the number of times the PU does not broadcast in the sensing time period (i.e. the total number of times a false alarm can occur). If p_{fa} is high, the algorithm tends to be too sensitive in its use of sensing reports.

- Global missed detection rate p_{md} – a measure of how likely an algorithm is to fail in the detection of the PU when it is broadcasting. This is determined by counting the total number of missed detections and averaging the total over the number of times the PU broadcasts:

$$\begin{aligned} p_{md} &= \frac{\sum_{t=1}^{T} e_{md}^{(t)}}{T_{md}}, \\ e_{md}^{(t)} &= \begin{cases} 1, & \text{if } \hat{\mathcal{H}}^{(t)} \neq \mathcal{H}^{(t)} = 1 \\ 0, & \text{otherwise} \end{cases}, \end{aligned} \tag{5.3}$$

  where T_{md} is the number of times the PU broadcasts in the sensing time period (i.e. the total number of times a missed detection can occur). If p_{md} is high, the algorithm is either receiving reports of poor quality (e.g. the network could be in a location where the PU power is low) or it is not prioritising accurate sensing reports.

- Spectrum sensing error rate p_e – the sum of missed detection and false alarm rates, and a measure of the overall performance of the algorithm. If it is assumed that $p(\mathcal{H}_0) = p(\mathcal{H}_1) = 0.5$, then the error is

$$p_e = \frac{p_{fa} + p_{md}}{2}. \tag{5.4}$$

5.2.4 Control algorithm

A control is required to compare and contextualise the performance of the proposed algorithm. The K-out-of-N algorithm was used:

$$\hat{\mathcal{H}} = \begin{cases} \mathcal{H}_0, & \text{if } \sum_{n=1}^{N_{su}} u_n < K = \frac{N_{su}}{2} \\ \mathcal{H}_1, & \text{if } \sum_{n=1}^{N_{su}} u_n \geq K = \frac{N_{su}}{2} \end{cases}, \tag{5.5}$$

where K is the decision threshold. Since $K = \frac{N_{su}}{2}$, the technique is termed the majority decision rule (MDR). It is equivalent to the distributed consensus fusion rule, assuming that the reports are quantised prior to fusion. This algorithm was chosen because of the lack of assumptions required prior to fusion.

5.2.5 Tests

The following test categories were used for the performance appraisal of the proposed algorithm.

- **Resistance to MU attacks** – This is the basic functionality of the algorithm; testing involved the exposure of the CRN to malicious agents of varying maliciousness and proportion.
- **Effect of increasing the number of SUs** – This involved the analysis of the change in performance when either the number of TUs or UTUs was changed.
- **Dynamics and complexity** – The transient behaviour of the algorithm was analysed here. The speed of the algorithm's response to changing network conditions, the computational overhead incurred by increasing the size of the network, and the time taken for the algorithm to converge were considered.
- **Factor effectiveness analysis** – Finally, the effectiveness of the likelihood functions defined in Section 3.4 was evaluated by assessing the sensing performance after the functions' sequential removal.

5.2.5.1 Resistance to MU attacks

The following tests were performed in the evaluation of the proposed algorithm's resistance to MU attacks.

- **Varying the node maliciousness of MUs** – The network was exposed to a fixed population of MUs; a Monte Carlo simulation was performed to estimate the global p_{fa} and p_{md} of the CRN in these conditions. The simulation was repeated with the maliciousness of the MU population varied in the range $-1 \leq r \leq 1$ with steps of 0.1. The results of this simulation, plotted in Figure 5.3, show how well the algorithm reacted to groups of users attacking at a given frequency (proportional to $|r|$). Furthermore, the algorithm's operating limits were tested with the consideration of extreme maliciousness values (i.e. $r = \pm 1$).
- **Varying the proportion of MUs** – The network was exposed to an increasing population of MUs with fixed maliciousness. Again, a Monte Carlo simulation estimating p_{fa} and p_{md} was performed for a given MU proportion. A proportion of MUs in the UTU population was considered in the range of 0% to 100% with steps of 10%. The sensing results associated with these MU proportions are plotted in Figure 5.4. This simulation shows how well the proposed algorithm handles large groups of attackers.
- **MU detection performance** – A snapshot of a scenario where a set of MUs with varying degrees of maliciousness attack the CRN was considered. The final beliefs associated with the malicious nodes were obtained, and are plotted in Figure 5.5. Since the proposed algorithm was formulated to perform joint estimations on u_n, $\hat{\mathcal{H}}$, and r_m, the information from $b(r_m)$ is automatically fed back to augment the other beliefs (i.e. (4.7) and (4.8)).

5.2.5.2 Increasing the number of SUs

Since the network contains two types of SUs (i.e. TUs and UTUs), a test involving the sensing performance associated with increasing either type of SU exclusively was performed.

- **Increasing the number of TUs in the network** – The network was initialised with only one TU, along with a number of UTUs (of which a fixed proportion were MUs). Once a complete Monte Carlo simulation involving the estimation of p_{fa} and p_{md} had been performed, an honest UTU was converted to a TU; this process was iterated until an upper bound on the number of TUs was reached. The results of this simulation for three scenarios – plotted in Figures 5.6, 5.7, and 5.8 – show how relay-based networks can augment performance by increasing the number of users providing a “ground truth” reference.

- **Increasing the number of UTUs in the network** – The network was initialised with a fixed number of TUs and a small number of UTUs (of which a fixed proportion were MUs). Once a complete Monte Carlo simulation involving the estimation of p_{fa} and p_{md} had been performed, a UTU was added to a random position in the network. This process was iterated until an upper bound on the number of UTUs was reached. The results of this simulation are plotted in Figure 5.9. An appropriate number of MUs were added to keep the MU proportion constant throughout the UTU population increase. This simulation shows how effectively the proposed algorithm leverages the additional data it receives, and how many UTUs are required in a network to attain a desired sensing performance.

5.2.5.3 Dynamics and complexity

The learning speed and computational complexity govern the temporal dynamics of the proposed algorithm.

- **Learning speed** – Multiple instances of the learning phase (see Section 4.3.4) were simulated for increasing periods of time (i.e. from $t = 1$ to $t = T_l$). The average p_e was determined from these simulations and plotted against time in Figure 5.10. Since the likelihoods $\hat{\zeta}_{mk}$ and $\hat{\tau}_{mk}$ are updated and approach their true values with each time instant, p_e is expected to improve over time.
- **Computational complexity** – The entire sensing process was run many times, and the average time taken for the proposed algorithm to make a final prediction was measured; this process was iterated over an increasing N_m, and is plotted in Figure 5.11. The purpose of this simulation was to provide empirical verification of the proposed algorithm’s linear computational cost ($\mathcal{O}(N_m)$).
- **Iterations and convergence** – A Monte Carlo simulation of the sensing process was performed, and the number of iterations taken for the proposed algorithm to converge on the beliefs was plotted in Figure 5.12. Since the sensing graph $\mathcal{G}_s$ contains no loops, belief propagation will converge to the true marginal distributions given a number of iterations η. The speed of convergence is an important factor when considering the algorithm’s implementation in a multi-hop network. As mentioned in Section 4.3.5, η is proportional to the number of hops in the network – it is desirable to keep this as low as possible.

5.2.5.4 Factor effectiveness analysis

The importance of the information provided by the factors defined in Section 3.4 was evaluated by nullifying each factor's contribution separately; the drop in performance caused by this was subsequently measured. The contribution of the factors was nullified as follows:

- for α_{t_k} and α_{u_m}, the error probabilities $p_{fa}^{(n)}$ and $p_{md}^{(n)}$ were set to zero,
- for factor γ_{mk}, the mutual measurement likelihood estimates $\hat{\gamma}_{mk}$ and $\hat{\zeta}_{mk}$ were set to 0.5, and
- for factors δ_{a_m} and $\delta_{b_m}^{i}$, the perceived maliciousness r_m was set to zero.

Once the contribution from a factor had been removed, a Monte Carlo simulation was performed to obtain p_{fa} and p_{md} for the proposed algorithm; this process was repeated with an increasing MU proportion. The drop in sensing performance due to the removal of each factor is plotted against the MU proportion in Figure 5.13. The results are interpreted as follows: consider two arbitrary factors – if the first factor's removal results in a larger sensing performance drop than that of the second factor's removal, the first factor is deemed more important.

5.3 RESULTS

The plots of the corresponding tests mentioned in Section 5.2.5 are found here. Tables 5.1 - 5.5 are sets of CRN parameters that are used in the simulations below.

Table 5.1. CRN parameter set 1.

Parameter	**Value**
N_{su}	30
N_k	5
T_l	30
η	3
A_{crn}	$50\ \mathrm{m} \times 50\ \mathrm{m}$

Table 5.2. CRN parameter set 2.

Parameter	**Value**
N_{su}	10
N_k	5
T_l	30
η	3
A_{crn}	$50 \text{ m} \times 50 \text{ m}$

Table 5.3. CRN parameter set 3.

Parameter	**Value**
N_{su}	30
N_k	$\{1, 2, 3, \cdots, 15\}$
T_l	30
η	3
A_{crn}	$65 \text{ m} \times 65 \text{ m}$

Table 5.4. CRN parameter set 4.

Parameter	**Value**
N_{su}	$\{10, 12, 15, \cdots, 30\}$
N_k	5
T_l	30
η	3
A_{crn}	$50 \text{ m} \times 50 \text{ m}$

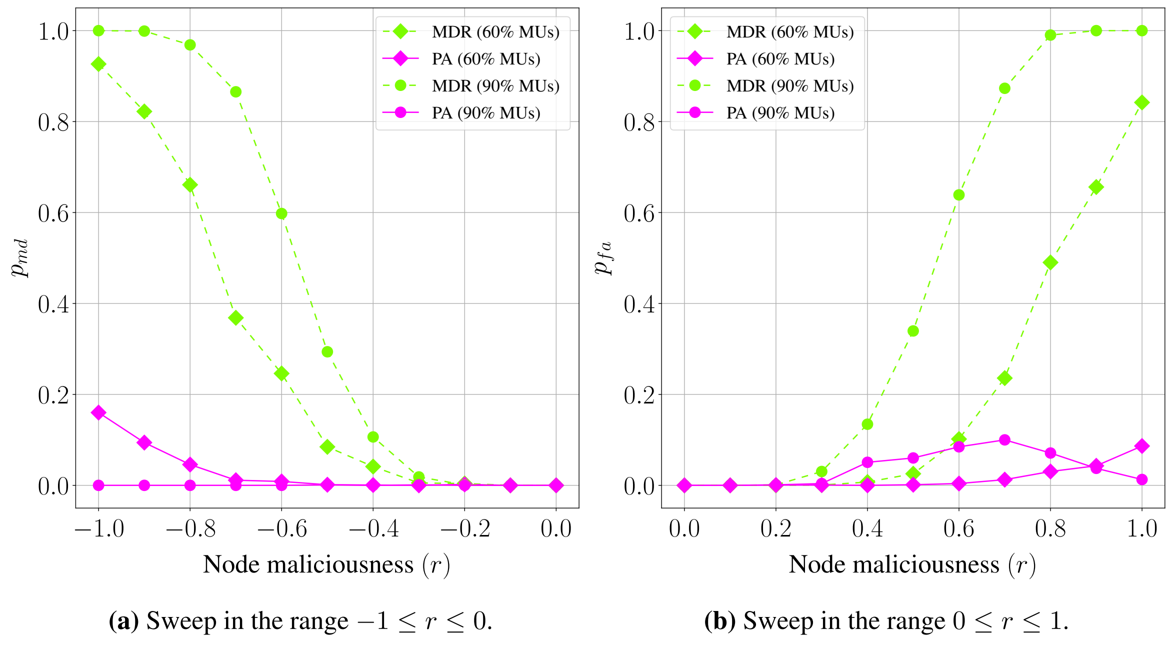
Table 5.5. CRN parameter set 5.

Parameter	**Value**
N_{su}	30
N_k	5
T_l	30
η	3
A_{crn}	$75 \text{ m} \times 75 \text{ m}$

5.3.1 Resistance to MU attacks

5.3.1.1 Varying the node maliciousness of MUs

The effect of varying the node maliciousness (r) in the network is shown in Figures 5.3(a) and 5.3(b); the CRN parameters of Table 5.1 were used for this simulation. Note that "proposed algorithm" is abbreviated as PA in these figures. The sensing performance, as a function of the change in maliciousness, is plotted in these figures. It is clear that for $|r| > 0.6$, the MDR struggled to handle the effect of altered reports, to the point that p_{fa} and p_{md} approached unity. The proposed algorithm was able to use its learned knowledge of the MUs to reduce the incidence of errors significantly, with neither p_{fa} nor p_{md} exceeding an error rate of 0.2.



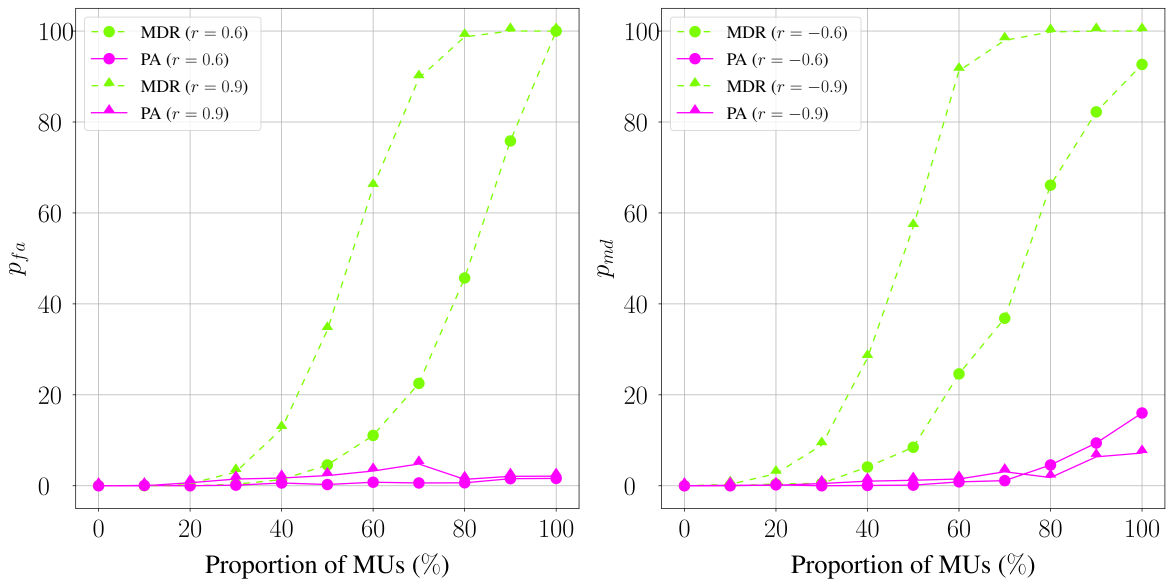
(a) Sweep in the range $-1 \leq r \leq 0$.

(b) Sweep in the range $0 \leq r \leq 1$.

Figure 5.3. Effect of varying the node maliciousness of MUs in the CRN.

5.3.1.2 Increasing the proportion of MUs in the network

The effect of varying the proportion of the MUs, while keeping the node maliciousness of each MU fixed, is shown in Figure 5.4(a) for SUs with $r = \{-0.9, -0.6\}$ and Figure 5.4(b) for $r = \{0.6, 0.9\}$. In both cases, the CRN parameters in Table 5.1 were used. The MDR struggled to make accurate predictions for large MU proportions. In contrast, the proposed algorithm handled the MUs effectively, with only a marginal increase in error rates at higher MU proportions.



(a) Sweep for $r = \{-0.9, -0.6\}$. **(b)** Sweep for $r = \{0.6, 0.9\}$.

Figure 5.4. Effect of increasing the proportion of MUs in the CRN.

5.3.1.3 MU detection performance

A snapshot of a scenario where a set of MUs with varying degrees of maliciousness attacks the CRN was considered; the CRN parameters in Table 5.2 were used. The final beliefs associated with the malicious nodes $(b(r_2), b(r_4), b(r_5), b(r_6))$ are given in Figure 5.5. The vertical lines in Figure 5.5 represent the true maliciousness of the respective nodes. The proposed algorithm overestimated the maliciousness associated with the MUs. This is because the algorithm cannot distinguish between reports that are malicious and reports that are measured incorrectly owing to noise and path loss. The loss in sensing performance due to an incorrect measurement is the same as that of a maliciously altered measurement; discrimination of reports from users making unreliable measurements still resulted in a sensing performance improvement.

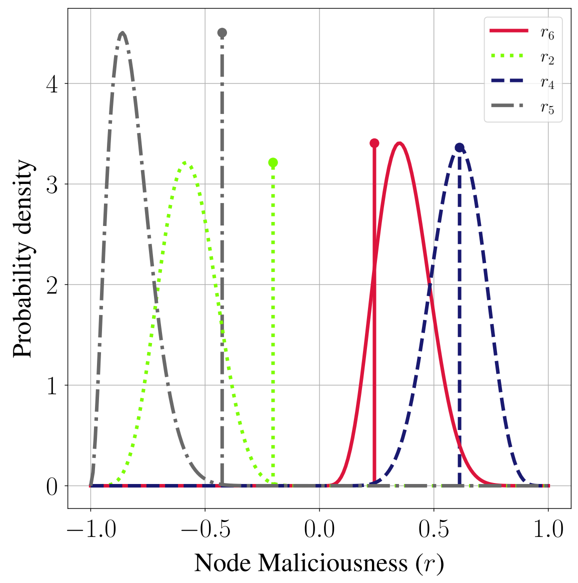
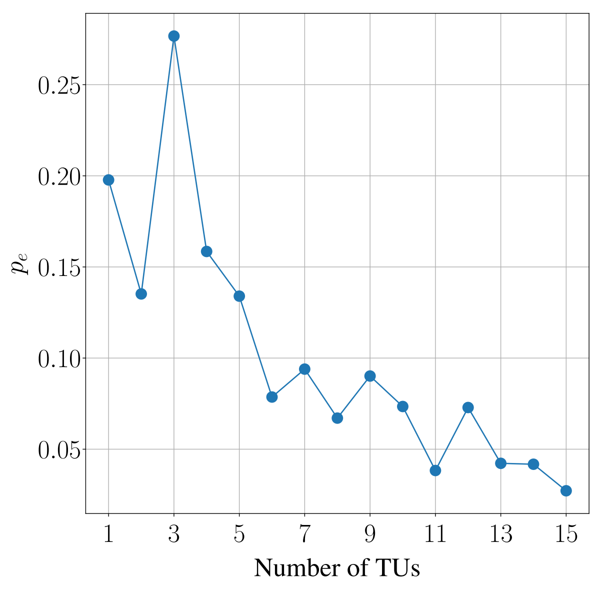


Figure 5.5. Final beliefs $b(r_m)$ for a group of MUs.

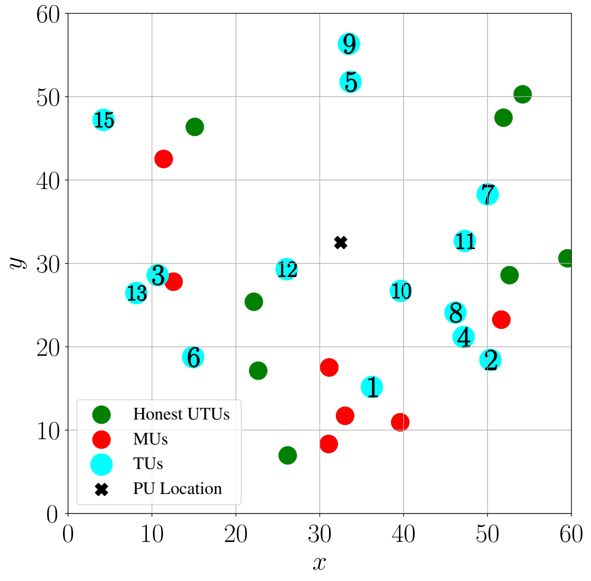
5.3.2 Effect of increasing the number of SUs

5.3.2.1 Increasing the number of TUs in the network

Because of the large amount of variety present in the placement of the TUs, three scenarios – each with a distinct set of SU maliciousness values – were considered. The CRN parameters in Table 5.3 were used for this simulation. A larger area ($65\text{ m} \times 65\text{ m}$) was used to increase the variation of the possible SU placements. The CRN diagrams and sensing results are shown for scenarios 1, 2, and 3 in Figures 5.6, 5.7, and 5.8 respectively. The enumerations of the TUs in Figures 5.6(b), 5.7(b), and 5.8(b) correspond to their order of introduction in the CRN. In general, using more TUs led to an overall improvement in the sensing performance. Adding TUs that were far away from the PU (low measurement quality) led to – in most cases – a small reduction in performance. In some instances, such as the addition of TU 3 in scenario 1, a much larger drop in sensing accuracy occurred. The impact of the performance reduction was minimised when more TUs were used in the network. Adding TUs that were close to the PU (high measurement quality) led to a significant improvement in sensing performance.

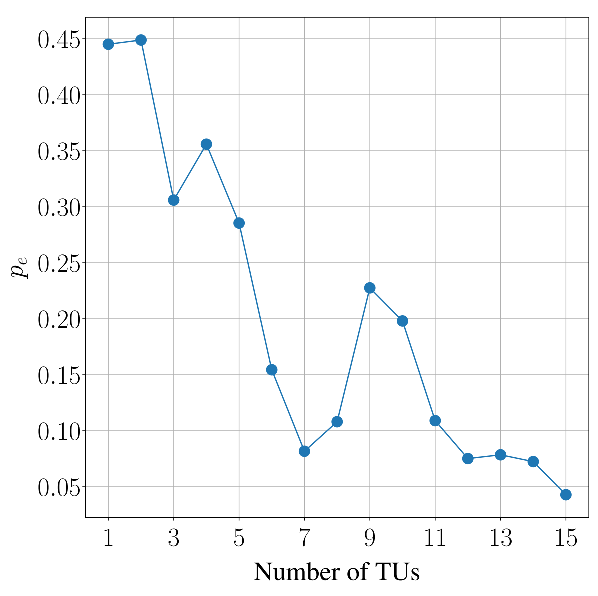


(a) Sensing performance

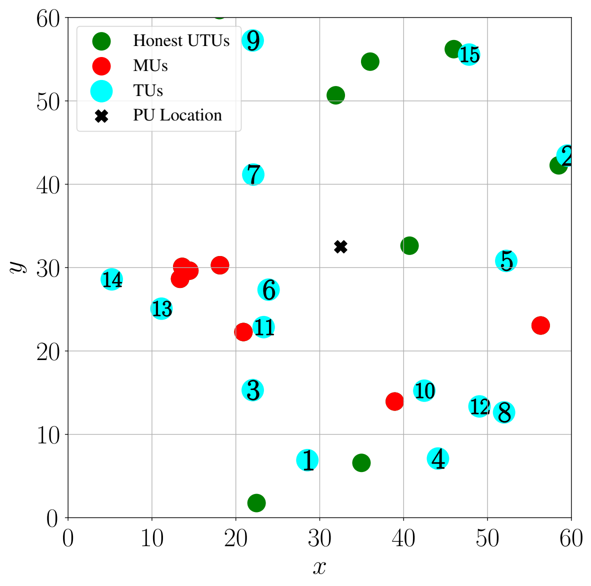


(b) Layout of CRN

Figure 5.6. Scenario 1 – TU sweep with $\mathbf{r} = [-0.369, 0.960, -0.278, 0.288, -0.964, 0.588, -0.865]$

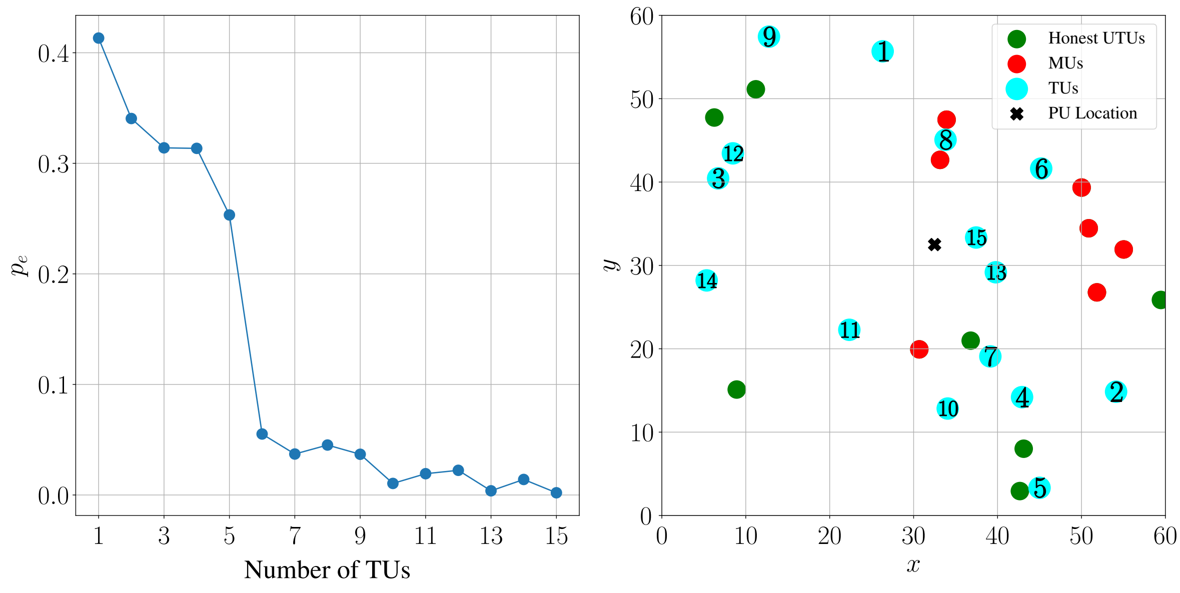


(a) Sensing performance



(b) Layout of CRN

Figure 5.7. Scenario 2 – TU sweep with $\mathbf{r} = [0.242, 0.168, 0.474, -0.249, 0.088, 0.916, -0.356]$

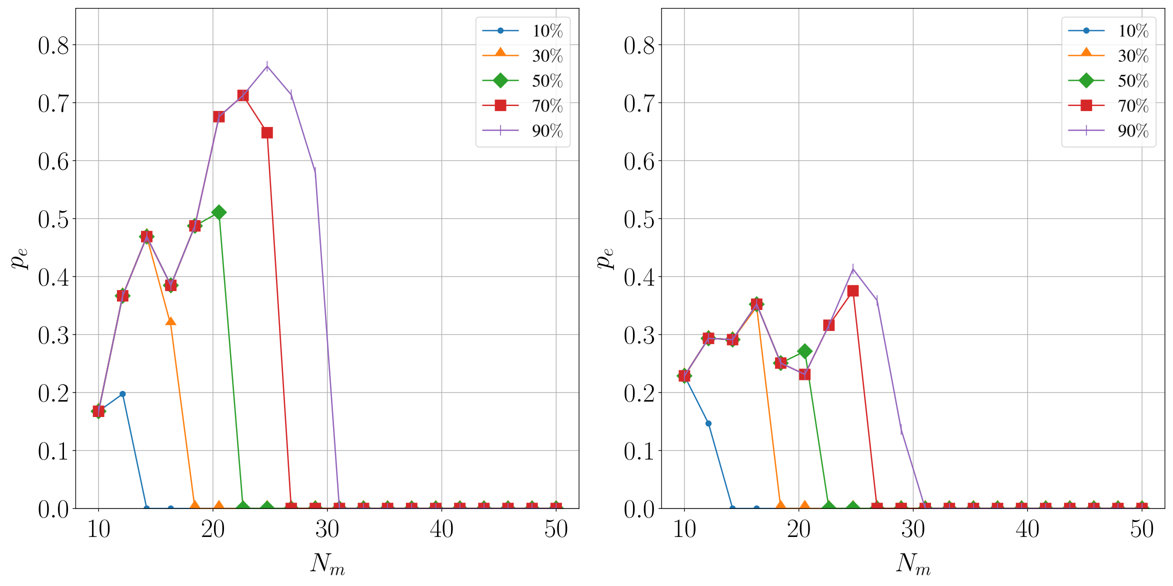


(a) Sensing performance **(b)** Layout of CRN

Figure 5.8. Scenario 3 – TU sweep with $\mathbf{r} = [0.960, 0.465, 0.804, 0.167, 0.481, 0.033, 0.294]$

5.3.2.2 Increasing the number of UTUs in the network

The network was initialised with five TUs and five UTUs. A small number of UTUs were added to each following simulation instance. The sensing results over the many instances, as a function of MU proportion (with maliciousness fixed to $r = -0.8$), is shown in Figure 5.9. The CRN parameters in Table 5.4 were used for the simulation. Increasing the number of UTUs resulted in a compromise, as the improvement in sensing performance due to the addition of more information was traded off against an increase in the overall number of MUs. Figure 5.9 shows that the CRN had a critical point of UTUs past which the sensing error dropped to zero. This point shows where the UTUs had “saturated” the CRN area; the point remained the same regardless of the sensing algorithm used. Figure 5.9(b) shows that the proposed algorithm can mitigate the adverse effects of attacks for networks with a small number of UTUs, even when the proportion of MUs in the UTU population is large.



(a) MDR **(b)** Proposed algorithm

Figure 5.9. Sensing performance as a function of the number of UTUs in the network and an increasing MU proportion.

5.3.3 Dynamics and complexity

5.3.3.1 Learning speed of the proposed algorithm

The CRN parameters in Table 5.5, with an MU proportion of 30% ($r = -0.3$), were used for this simulation. The plot in Figure 5.10 shows the learning process for the proposed algorithm. At time $t = 1$, the algorithm possessed no prior knowledge of the UTUs, with its sensing performance equal to the MDR. The proposed algorithm began with no prior information on the state of the CRN, treating each SU report independently. As measurements and results were stored, the proposed algorithm built up associations between SU reports, eventually leading to the preference of reports the algorithm deemed reliable. This caused an improvement in sensing performance when compared to the MDR. After two time instants, the proposed algorithm accrued enough information to improve the sensing performance significantly, as more information was available to express the distribution in (3.28).

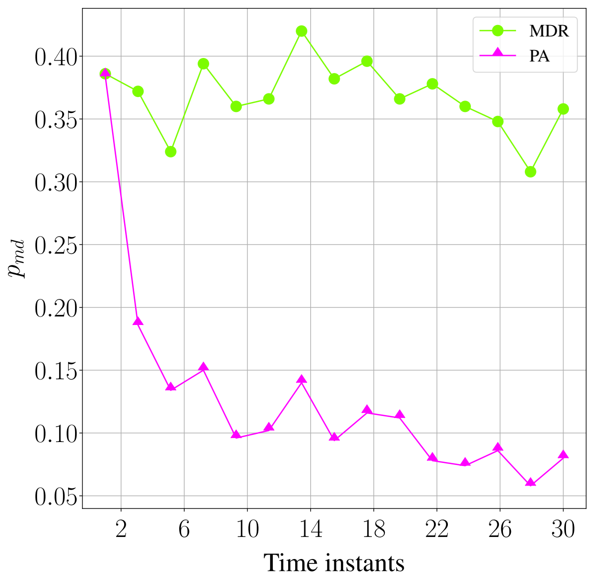


Figure 5.10. Comparison of the sensing performance for the proposed algorithm and the MDR over 30 time instants.

5.3.3.2 Computational complexity of the proposed algorithm

The CRN parameters in Table 5.4 were used for the simulation. The number of UTUs was increased from 10 to 30, and the time taken to run each sensing instant was measured; the results are plotted in Figure 5.11. An Intel Core i5-6600 processor was used to perform the simulation. The form of the plot in Figure 5.11 shows that the algorithm's computational cost grows approximately linearly.

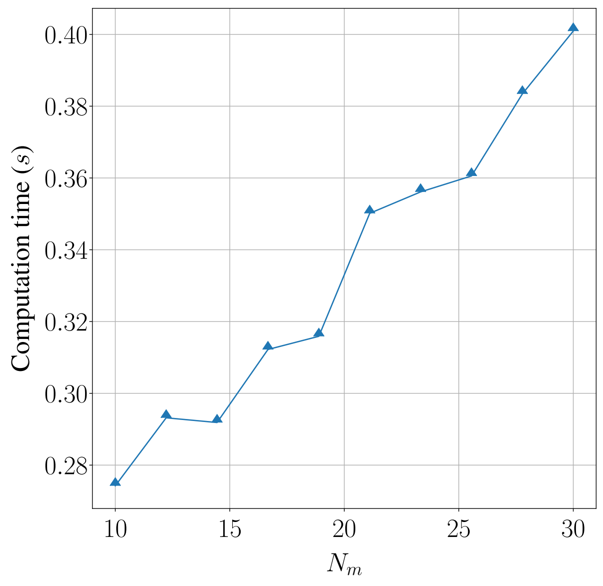


Figure 5.11. Average computation time to run a simulation of the process in Figure 4.8 with an increasing number of UTUs.

5.3.3.3 Iterations and convergence

Figure 5.12 shows the plot of the number of iterations against the estimated (and true) beliefs; the proposed algorithm's belief in $\mathcal{H}$ converges to the true state in the first iteration, resulting in $\eta = 3$ hops across the network to make a single prediction. The CRN parameters in Table 5.1 were used for this simulation. While the rapid convergence of beliefs in the algorithm is useful for a multi-hop implementation, it could imply that the technique is overly aggressive with PU state predictions.

5.3.4 Factor effectiveness analysis

The sensing performance after removing the influence of a single class of factors in $\mathcal{G}_s$, measured over an increasing MU proportion, is shown in Figure 5.13. The maliciousness of the MUs was fixed to $r = -0.9$, and the CRN parameters of Table 5.1 were used for the simulation.

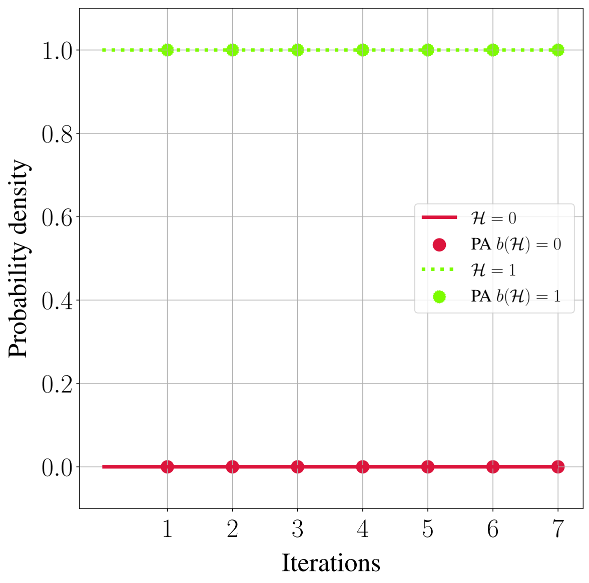


Figure 5.12. Belief $b(\mathcal{H})$ as a function of iterations.

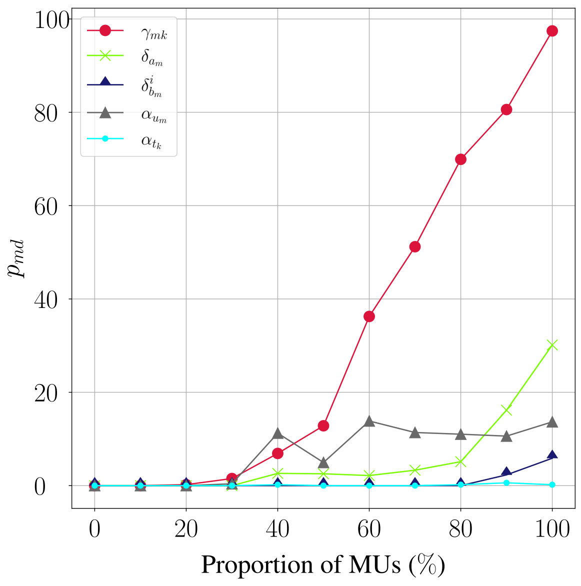


Figure 5.13. The impact of removing the contribution of a class of factors from the factor graph.

5.4 ANALYSIS

This section contains the analysis of the results presented in Section 5.3.

5.4.1 Resistance to MU attacks

The proposed algorithm was highly resistant to MU attacks, but the performance degraded slightly when exposed to very malicious attacks (i.e. large $|r|$) and large attack populations (MU proportion close to 1). The algorithm could not distinguish between altered and incorrect reports; measurements from SUs placed in unfavourable conditions were discriminated as though they were maliciously altered, resulting in the algorithm overestimating the maliciousness of the SUs in a noisy and lossy CRN environment (as seen in Figure 5.5).

It should be noted that the probabilistic attack model assumed by the algorithm is limited in sophistication. Employing a more sophisticated attack model may lead to worse results, but the cause-agnostic nature of the algorithm's formulation ensures that reports deviating from the norm, regardless of attack model, will be discriminated. Efficiently counteracting more complicated attacks requires the sensing model $\mathcal{G}_s$ to be re-formulated.

5.4.2 Increasing the number of SUs

It is clear from the scenarios in Figures 5.6, 5.7, and 5.8 that while a non-monotonic relationship was observed between p_e and N_k, the overall trend was an improved sensing performance for a greater number of TUs. The placement of the TUs relative to the PU location played an important role in the enhancement of sensing performance. Performance was reduced when TUs far away from the PU were introduced to the network; conversely, performance improved significantly when the TU was in a reliable location. Since the PU location was not known prior to sensing, a specific number of TUs was required in the network to ensure an improvement in performance. Thus, the proposed algorithm was found to be highly sensitive to TU diversity and location. The utility of the TUs in decentralised sensing could be improved further if location awareness is added to the algorithm – facilitating this change would require adapting the sensing model $\mathcal{G}_s$ accordingly.

The effect of the number of UTUs on the sensing performance of the CRN was dependent on the proportion of MUs present in the UTU population. The sensing error rate p_e initially increased with the introduction of new UTUs. For a given MU proportion, p_e then dropped after a critical number of UTUs was introduced; this number was identical for both the proposed algorithm and majority decision rule. The proposed algorithm did, however, result in a much lower p_e when N_m was below the critical number, implying that this sensing scheme may be more effective for smaller networks.

5.4.3 Dynamics and complexity

Figure 5.10 shows the adaptation of the proposed algorithm to the network conditions; the buffering of the SU measurements used to estimate $\hat{\zeta}_{mk}$ and $\hat{\tau}_{mk}$ results in a "learning period" for the algorithm. This functionality allows the algorithm to adapt to a sudden change in network conditions. The bulk of the sensing performance improvement occurs within the first few time instants, with the remaining time instants only marginally improving the performance. This, coupled with the algorithm's quick convergence (seen in Figure 5.12), shows that the dynamics are favourable for a real-time implementation. The algorithm's computational cost grows linearly, as seen in Figure 5.11. This is due to linear growth in the number of variable and factor nodes in the sensing graph $\mathcal{G}_s$. Thus, a decrease in complexity would require a fundamental change in the shape of $\mathcal{G}_s$.

5.4.4 Factor effectiveness analysis

The removal of each class of factor adversely affected the sensing performance when the MU proportion was increased.

1. The removal of the γ_{mk} factors, responsible for capturing the mutual occurrence of measurements from a given UTU m and a TU k, resulted in the steepest drop in sensing performance. This drop was especially pronounced for larger proportions of MUs, showing that this factor contributes significantly to MU resistance. The CRN does not leverage cooperative sensing without the use of this factor, as all other factors only consider the effects of an SU in isolation. Furthermore, the generality of this factor allows it to serve as a catch-all for phenomena that were not accurately determined, such as the assumptions made for $p_{md}^{(n)}$ and $p_{fa}^{(n)}$.
2. The α_{u_m} factors contributed moderately to PU state prediction. The reduction in sensing performance here shows the effects of incorrectly assuming $p_{md}^{(n)}$ and $p_{fa}^{(n)}$. Setting $p_{md}^{(n)} = 0.5$ and $p_{fa}^{(n)} = 0.5$ results in a drop in sensing performance when MUs are present. The magnitude of this drop is mitigated by the presence of the γ_{mk} factor.
3. The removal of the α_{t_k} factors did not significantly influence the sensing performance when the CRN was under MU attacks. This was due to the small of number of α_{t_k} factors in comparison to the α_{u_m} factors (since there are typically many more UTUs than TUs in the CRN).
4. The δ_{a_m} factors (responsible for determining $b(r_m)$ using the current report) also contributed significantly to sensing performance, as evidenced by the large increase in p_{md} for an increasing MU proportion.

5. The $\delta_{b_m}^i$ factors (responsible for determining $b(r_m)$ using historical reports) contributed little to sensing performance. These factors serve to fine-tune the MU maliciousness estimation made by the δ_{a_m} factors, and are only somewhat effective against a large MU population.

The sensing performance and computational requirements can be traded off by selectively removing factors from the factor graph (based on their effectiveness).

5.5 CHAPTER SUMMARY

The performance of the proposed algorithm was evaluated using a simulation platform developed in Python. The results were obtained from tests formulated according to a number of criteria, including the algorithm's resistance to the probabilistic attack model, the effect of changing the number of SUs in the network, the dynamics and complexity of the algorithm, and the effectiveness of the factors of which the sensing model $\mathcal{G}_s$ is composed. The MDR was used as a control for the experiment. The results were summarised, with the algorithm deemed to handle the impacts of the probabilistic attack model for which it was formulated effectively.

CHAPTER 6 CONCLUSION

A novel algorithm, used to perform secure sensing in relay-based decentralised CRNs, was developed in this work. The following outputs emanated from the formulation of the proposed algorithm:

- A literature review of CRNs and security was presented in Chapter 2. The cognitive cycle was introduced as a new paradigm in radio communication. The subfields of CR were defined in terms of the nodes in this cycle. Individual and cooperative sensing techniques, physical layer attacks, and defence algorithms were summarised. The formulation of integrated sensing and defence algorithms was presented as a step forward in the realisation of a fully cognitive radio.
- The statistical model forming the backbone of the proposed algorithm was described in Chapter 3. The factor graph was chosen as an appropriate model to capture the intricacies of the sensing process. An analytical expression for the joint distribution encoding all possible outcomes of the sensing process was obtained, concluding the development of the sensing model.
- Belief propagation was applied to the developed statistical model to obtain the proposed algorithm in Chapter 4. A data exchange protocol for a relay-based decentralised CRN was formulated by interpreting the utility and meaning of the messages derived.
- A performance evaluation and analysis of the proposed algorithm was conducted in Chapter 5. The algorithm was found to be resistant to MU attacks. The sensing performance improved when more UTUs and TUs were added to the network, showing that cooperative diversity was exploited effectively; the performance was, however, highly sensitive to the TU placement relative to the PU. The favourable dynamics of the algorithm – including the convergence speed and the buffering of measurements to handle changes in the environment – show that probabilistic inference-based algorithms are suitable candidates for distributed sensing and fusion.

6.1 FUTURE WORK

Further research in Byzantine MU and distributed sensing either involves refining already developed concepts or solving existing problems with the proposed algorithm. The following avenues for further research exist:

- **A framework for graphical modelling in secure distributed sensing** – The sensing model developed for this work can be reused to formulate a similar algorithm to prevent primary user emulation attacks in the same CRNs. Furthermore, the sensing model considered in this work is not exclusive to CRNs; the developed graphical model and data exchange protocol can be reused for other secure estimation problems in the domain of wireless sensor networks.
- **Formulating and understanding advanced Byzantine attack models** – Significantly more advanced attack protocols can be developed using techniques from machine learning. Specifically, deep learning models can be used by malicious agents to determine the best possible falsification strategies for a given defence algorithm. More research into the effects of deep learning-based Byzantine attacks on CRNs must be conducted.
- **Structure learning to prevent Byzantine attacks** – Structure learning can be used to estimate the conditional dependencies between desired variables autonomously, given some form of input data – effectively automating the model development in Chapter 3. This technique can be used to observe the relationships between variables in complicated sensing and attack models; sophisticated defence algorithms can be created by performing inference on the generated graphical models.

REFERENCES

[1] “Cisco visual networking index: Global mobile data traffic forecast update, 2017–2022,” Cisco Systems, Tech. Rep., Feb. 2019.

[2] “Imt traffic estimates for the years 2020 to 2030,” International Telecommunication Union, Tech. Rep., 2015.

[3] S. Haykin, “Cognitive radio: Brain-empowered wireless communications,” *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 2, pp. 201–220, 2005.

[4] J. Mitola and G. Q. Maguire, “Cognitive radio: Making software radios more personal,” *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, Aug. 1999, ISSN: 1070-9916.

[5] C. Sun, W. Zhang, and K. B. Letaief, “Cluster-based cooperative spectrum sensing in cognitive radio systems,” in *2007 IEEE International Conference on Communications*, Jun. 2007, pp. 2511–2515.

[6] T. S. Dhope and D. Simunic, “Cluster based cooperative sensing: A survey,” in *2012 International Conference on Communication, Information Computing Technology (ICCICT)*, Oct. 2012, pp. 1–6.

[7] Z. Shu, Y. Qian, and S. Ci, “On physical layer security for cognitive radio networks,” *IEEE Network*, vol. 27, no. 3, pp. 28–33, May 2013, ISSN: 0890-8044.

[8] L. Zhang, G. Ding, Q. Wu, Y. Zou, Z. Han, and J. Wang, “Byzantine attack and defense in cognitive radio networks: A survey,” *IEEE Communications Surveys Tutorials*, vol. 17, no. 3, pp. 1342–1363, Third quarter 2015, ISSN: 1553-877X.

[9] P. Kaligineedi, M. Khabbazian, and V. K. Bhargava, “Malicious user detection in a cognitive radio cooperative sensing system,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 8, pp. 2488–2497, 2010, ISSN: 15361276.

[10] F. Penna, Y. Sun, L. Dolecek, and D. Cabric, “Detecting and counteracting statistical attacks in cooperative spectrum sensing,” *IEEE Transactions on Signal Processing*, vol. 60, no. 4, pp. 1806–1822, Apr. 2012, ISSN: 1053-587X.

[11] A. S. Rawat, P. Anand, H. Chen, and P. K. Varshney, “Collaborative spectrum sensing in the presence of Byzantine attacks in cognitive radio networks,” *IEEE Transactions on Signal Processing*, vol. 59, no. 2, pp. 774–786, 2011, ISSN: 1053587X.

[12] S. Liu, H. Zhu, S. Li, X. Li, C. Chen, and X. Guan, “An adaptive deviation-tolerant secure scheme for distributed cooperative spectrum sensing,” in *2012 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2012, pp. 603–608.

[13] T. Yucek and H. Arslan, “A survey of spectrum sensing algorithms for cognitive radio applications,” *IEEE Communications Surveys Tutorials*, vol. 11, no. 1, pp. 116–130, First 2009, ISSN: 1553-877X.

[14] B. Wang and K. J. R. Liu, “Advances in cognitive radio networks: A survey,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 1, pp. 5–23, Feb. 2011.

[15] A. Ali and W. Hamouda, “Advances on spectrum sensing for cognitive radio networks: Theory and applications,” *IEEE Communications Surveys Tutorials*, vol. 19, no. 2, pp. 1277–1304, Second quarter 2017.

[16] G. I. Tsiropoulos, O. A. Dobre, M. H. Ahmed, and K. E. Baddour, “Radio resource allocation techniques for efficient spectrum access in cognitive radio networks,” *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 824–847, First quarter 2016.

[17] K. Cichoń, A. Kliks, and H. Bogucka, “Energy-efficient cooperative spectrum sensing: A survey,” *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 1861–1886, Third quarter 2016, ISSN: 1553-877X.

[18] W. Lee and I. F. Akyildiz, “Optimal spectrum sensing framework for cognitive radio networks,” *IEEE Transactions on Wireless Communications*, vol. 7, no. 10, pp. 3845–3857, Oct. 2008.

[19] I. F. Akyildiz, W.-Y. Lee, and K. R. Chowdhury, “Crahns: Cognitive radio ad hoc networks,” *Ad Hoc Networks*, vol. 7, no. 5, pp. 810–836, 2009, ISSN: 1570-8705.

[20] L. Le and E. Hossain, “Cross-layer optimization frameworks for multihop wireless networks using cooperative diversity,” *IEEE Transactions on Wireless Communications*, vol. 7, no. 7, pp. 2592–2602, Jul. 2008.

[21] N. ul Hasan, W. Ejaz, M. K. Atiq, and H. S. Kim, “Distributed connectivity restoration using cognitive relay nodes positioning in damaged wireless sensor network,” in *2013 IEEE Symposium on Wireless Technology Applications (ISWTA)*, Sep. 2013, pp. 159–162.

[22] H. Zeng and Z. Kang, “Relay node placement to restore connectivity in wireless sensor networks,” in *2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN)*, May 2017, pp. 301–305.

[23] N. S. Shankar, C. Cordeiro, and K. Challapali, “Spectrum agile radios: Utilization and sensing architectures,” in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.*, Nov. 2005, pp. 160–169.

[24] P. Wang, L. Xiao, S. Zhou, and J. Wang, “Optimization of detection time for channel efficiency in cognitive radio systems,” in *2007 IEEE Wireless Communications and Networking Conference*, Mar. 2007, pp. 111–115.

[25] Q. Zhao, S. Geirhofer, L. Tong, and B. M. Sadler, “Opportunistic spectrum access via periodic channel sensing,” *IEEE Transactions on Signal Processing*, vol. 56, no. 2, pp. 785–796, Feb. 2008.

[26] H. Tang, “Some physical layer issues of wide-band cognitive radio systems,” in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.*, Nov. 2005, pp. 151–159.

[27] N. Han, S. Shon, J.H.Chung, and J.M.Kim, “Spectral correlation based signal detection method for spectrum sensing in IEEE 802.22 WRAN systems,” in *2006 8th International Conference Advanced Communication Technology*, vol. 3, Feb. 2006, 6 pp.–1770.

[28] J. Proakis, *Digital Communications 5th Edition*. McGraw Hill, 2007.

[29] D. Cabric, S. M. Mishra, and R. W. Brodersen, “Implementation issues in spectrum sensing for cognitive radios,” in *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers, 2004.*, vol. 1, Nov. 2004, 772–776 Vol.1.

[30] J. Wang and I. R. Chen, “Trust-based data fusion mechanism design in cognitive radio networks,” in *2014 IEEE Conference on Communications and Network Security*, Oct. 2014, pp. 53–59.

[31] A. W. Min, K. G. Shin, and X. Hu, “Secure cooperative sensing in IEEE 802.22 wrans using shadow fading correlation,” *IEEE Transactions on Mobile Computing*, vol. 10, no. 10, pp. 1434–1447, Oct. 2011, ISSN: 1536-1233.

[32] X. Huang, L. Chen, Q. Chen, and B. Shen, “Joint malicious user detection and resource allocation in cognitive radio networks,” in *2015 10th International Conference on Communications and Networking in China (ChinaCom)*, Aug. 2015, pp. 278–282.

[33] J. Ma, G. Zhao, and Y. Li, "Soft combination and detection for cooperative spectrum sensing in cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 11, pp. 4502–4507, Nov. 2008.

[34] S. A. Aldosari and J. M. F. Moura, "Topology of sensor networks in distributed detection," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 5, May 2006, pp. V–V.

[35] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, Jul. 2008.

[36] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.

[37] A. Ainomäe, T. Trump, and M. Bengtsson, "Distributed diffusion LMS based energy detection," in *2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, Oct. 2014, pp. 176–183.

[38] A. Attar, H. Tang, A. V. Vasilakos, F. R. Yu, and V. C. M. Leung, "A survey of security challenges in cognitive radio networks: Solutions and future research directions," *Proceedings of the IEEE*, vol. 100, no. 12, pp. 3172–3186, Dec. 2012.

[39] R. Chen, J. Park, and J. H. Reed, "Defense against primary user emulation attacks in cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 1, pp. 25–37, Jan. 2008.

[40] M. Laghate and D. Cabric, "Cooperative spectrum sensing in the presence of correlated and malicious cognitive radios," *IEEE Transactions on Communications*, vol. 63, no. 12, pp. 4666–4681, 2015, ISSN: 00906778.

[41] F. Adelantado and C. Verikoukis, "A non-parametric statistical approach for malicious users detection in cognitive wireless ad-hoc networks," in *2011 IEEE International Conference on Communications (ICC)*, Jun. 2011, pp. 1–5.

[42] S. Sodagari, A. Attar, V. C. M. Leung, and S. G. Bilen, "Denial of service attacks in cognitive radio networks through channel eviction triggering," in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, Dec. 2010, pp. 1–5.

[43] A. Vempaty, K. Agrawal, P. Varshney, and H. Chen, "Adaptive learning of Byzantines' behavior in cooperative spectrum sensing," in *2011 IEEE Wireless Communications and Networking Conference*, Mar. 2011, pp. 1310–1315.

[44] X. He, H. Dai, and P. Ning, “HMM-based malicious user detection for robust collaborative spectrum sensing,” *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 11, pp. 2196–2208, Nov. 2013, ISSN: 0733-8716.

[45] C. Bishop, *Pattern Recognition and Machine Learning*, First. New York: Springer, 2006.

[46] E. Soltanmohammadi and M. Naraghi-Pour, “Fast detection of malicious behavior in cooperative spectrum sensing,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 3, pp. 377–386, Mar. 2014, ISSN: 0733-8716.

[47] F. Penna and R. Garello, “Decentralized Neyman-Pearson test with belief propagation for peer-to-peer collaborative spectrum sensing,” *IEEE Transactions on Wireless Communications*, vol. 11, no. 5, pp. 1881–1891, May 2012, ISSN: 1536-1276.

[48] S. Zarrin and T. J. L. Lim, “Belief propagation on factor graphs for cooperative spectrum sensing in cognitive radio,” *2008 3rd IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks*, pp. 1–9, 2008.

[49] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001, ISSN: 0018-9448.

[50] K. P. Murphy, Y. Weiss, and M. I. Jordan, “Loopy belief propagation for approximate inference: An empirical study,” in *In Proceedings of Uncertainty in AI*, 1999, pp. 467–475.