Bio-Inspired Medium Access Control for Wireless Sensor Networks

Tautvydas Mickus

PhD

University of York

Electronics

March 2017

Abstract

This thesis studies the applications of biologically inspired algorithms and behaviours to the Medium Access Control (MAC) layer of Wireless Sensor Networks (WSNs). By exploring the similarity between a general communications channel and control engineering theory, we propose a simple method to control transmissions that we refer to as transmission delay. We use this concept and create a protocol inspired by Particle Swarm Optimisation (PSO) to optimise the communications. The lessons learned from this protocol inspires us to move closer to behaviours found in nature and the Emergence MAC (E-MAC) protocol is presented. The E-MAC protocol shows emergent behaviours arising from simple interactions and provides great throughput, low end-to-end delay and high fairness. Enhancements to this protocol are later proposed. We empirically evaluate these protocols and provide relevant parameter sweeps to show their performance. We also provide a theoretical approach to proving the settling properties of E-MAC. The presented protocols and methods provide a different approach towards MAC in WSNs.

Contents

Abstract **2**

Acknowledgements **12**

Declaration **13**

1 Introduction **14**

1.1 Motivation 14

1.2 Hypothesis 16

1.3 Outline 16

1.4 Publications 18

2 Literature Review **19**

2.1 Wireless Sensor Networks 20

2.1.1 Applications 20

2.1.2 Devices 21

2.1.3 Scenarios and Topologies 21

2.1.4 Communications Architecture 22

2.1.5 Constraints and Challenges 23

2.2 Medium Access Control 24

2.2.1 Schedule-Based Medium Access Control Protocols 25

2.2.2 Contention-Based Medium Access Control Protocols 25

2.2.3 Hybrid Medium Access Control Protocols 25
2.2.4 Limitations and Challenges 26
2.3 Multi-Agent Systems and Distributed Artificial Intelligence 29
2.3.1 Deliberative Multi-Agent Systems and Machine Learning 30
2.3.2 Reactive Multi-Agent Systems 34
2.3.3 Structure of a Multi-Agent System 35
2.3.4 Properties of Bio-Inspired Systems 36
2.3.5 Wireless Sensor Network as Multi-Agent System 37
2.4 Artificial Intelligence in Wireless Sensor Networks 38
2.5 Conclusions 40

3 Experimental Methodology 41

3.1 Wireless Sensor Network Simulator 41
3.1.1 Classical Single Hop Topology 42
3.1.2 Multi-Hop Linear Chain Topology 42
3.1.3 Multi-Hop Merging Linear Chain Topology 43
3.1.4 Ad-Hoc Network Topology 43
3.1.5 Ad-Hoc Networks with Point Processors 44
3.1.6 Radio Propagation and Link 45
3.1.7 Traffic Model 47
3.2 Empirical Evaluation 48
3.2.1 Performance Metrics 48
3.2.2 Statistical Validation 50
3.2.3 Model Validation 50
3.3 Comparison Scheme 51
3.4 Conclusion 52

4 Control Engineering in Medium Access Control and Particle Swarm Optimi-

sation **53**

4.1 Medium Access Control and Control Engineering 54

4.1.1 Controlled Transmission Delay 55

4.1.2 ALOHA with Transmission Delay 56

4.1.3 Experimentation 57

4.1.4 Conclusions 57

4.2 Particle Swarm Optimisation 58

4.2.1 Flocking in Natural Systems 58

4.2.2 Social Concept 59

4.2.3 Basic Particle Swarm Optimisation Operation 59

4.2.4 Particle Swarm Optimisation Algorithm 60

4.2.5 Multi-dimensional Particle Swarm Optimisation 61

4.3 Optimising Transmission Delay for Multi-Hop Chain Network 61

4.3.1 Offline Particle Swarm Optimisation 62

4.3.2 Experimental Optimisation Results 63

4.4 Conclusions 67

5 Swarming Medium Access Control Protocol **69**

5.1 Motivation 70

5.1.1 Control Engineering in Medium Access Control 71

5.2 Swarm Intelligence-Based Medium Access Control Protocol 71

5.3 Scenario and Assumptions 73

5.3.1 The Basic Scenario 73

5.3.2 Traffic 73

5.3.3 Propagation and Radio 74

5.3.4 Scheme for Comparison 74

5.3.5 Simulation Parameters 74

5.4 Performance Evaluation 75

5.4.1 Metrics for Analysis 75
5.4.2 Results 75
5.5 Conclusions 80

6 Bio-Inspired Emergence Medium Access Control 81

6.1 Motivation 81
6.2 Biological Metaphors 83
6.2.1 Task Allocation and Division of Labour in Social Insects 84
6.2.2 Ant Clustering and Sorting 85
6.2.3 Biological Mechanisms 86
6.3 Emergence Medium Access Control Protocol 87
6.3.1 Protocol Design 87
6.3.2 Simulation Parameters and Assumptions 93
6.3.3 Results 95
6.4 Settling Time and Theoretical Analysis 105
6.4.1 Definition of the Markov Model 105
6.4.2 Tendencies and Convergence 111
6.5 Conclusions 113

7 Enhancing The Emergence Medium Access Control protocol 115

7.1 Motivation 116
7.1.1 E-MAC Lock-Up Description 116
7.2 Biological Inspiration 117
7.2.1 Firefly Behaviour and Synchronisation 117
7.2.2 De-Synchronisation 118
7.2.3 Diffusion 120
7.3 E-MAC+ 121
7.3.1 Protocol Design 121

7.4 Empirical Evaluation on a Merging Chain 123
7.4.1 Simulation Parameters and Assumptions 123
7.4.2 Performance Metrics 124
7.4.3 Performance Evaluation 125
7.4.4 Gradient Formation and Basic Operation 128
7.4.5 Discussion 130
7.5 Empirical Evaluation on an Ad-Hoc Network 132
7.5.1 Simulation Parameters and Assumptions 132
7.5.2 Performance Metrics 133
7.5.3 Results 133
7.6 Conclusions 140

8 Conclusions and Further Work **142**
8.1 Conclusions 142
8.1.1 Original Contributions 143
8.1.2 Hypothesis 145
8.2 Recommendations for Further Work 146

Abbreviations **149**

Bibliography **152**

List of Figures

1.1 Examples of biological swarm systems 15

2.1 Communication layers of WSN 23
2.2 Fuzzy inference system [1] 31
2.3 Comparison of Centralised, Decentralised and Distributed systems . . . 36

3.1 Single-Hop Topology 42
3.2 Multi-hop chain topology 43
3.3 Multi-hop chain topology 44
3.4 Generated Network Architectures 45
3.5 Generated point processed Network Architectures 46
3.6 Communication and Interference on multi-hop chain network 47
3.7 Box plot example 51
3.8 Classic ALOHA simulations 51

4.1 MAC and control engineering mechanism comparison 55
4.2 MAC and control engineering mechanism comparison 55
4.3 Throughput when direct control engineering based feedback for $txDelay$ is applied 58
4.4 Fitness values versus number of iterations - Optimising transmission delays for chain network (Section 3.1.1) using PSO. 65
4.5 Throughput versus number of iterations - Optimising transmission delays for chain network (Section 3.1.1) using PSO. 65
4.6 Fairness versus number of iterations - Optimising transmission delays for chain network (Section 3.1.1) using PSO. 66
4.7 Transmission delay solutions for each node found by PSO. 67
4.8 Average throughput for nodes with $txDelay$ solutions presented in Figure 4.7. 67

5.1 Throughput CDF 76
5.2 Delay CDF 77
5.3 Fairness CDF 78
5.4 Transmissions per packet CDF 79

6.1 Response threshold functions . . . 84
6.2 Simulation of ant clustering . . . 86
6.3 Probability of responding after consecutive events when using Robbins Monro algorithm for approximated $pSuccess$, $\alpha = 0.2$ . . . 90
6.4 The chain scenario . . . 93
6.5 Throughput for system with 2 sources and 2 hop interference range . . . 96
6.6 Throughput for system with 2 sources and 3 hop interference range . . . 97
6.7 End-To-End delay for system with 2 sources and 2 hop interference range 97
6.8 End-To-End delay for system with 2 sources and 3 hop interference range 98
6.9 Fairness for system with 2 sources . . . 99
6.10 Mean throughput comparison for different numbers of sources . . . 99
6.11 Mean fairness comparison for different numbers of sources . . . 100
6.12 Throughput variation when α and $change\_scale$ are varied . . . 101
6.13 Throughput variation when α and $change\_scale$ are varied . . . 102
6.14 End-to-end delay variation from source 5 when α and $change\_scale$ are varied . . . 102
6.15 End-to-end delay variation from source 5 when α and $change\_scale$ are varied . . . 103
6.16 End-to-end delay variation from source 11 when α and $change\_scale$ are varied . . . 103
6.17 End-to-end delay variation from source 11 when α and $change\_scale$ are varied . . . 104
6.18 Multi-hop chain network . . . 106
6.19 Probability of successful transmission given static transmission delay . . 107
6.20 Change in pSucc value based on Robbins Monro algorithm given success or failure ($\alpha = 0.2$) . . . 108
6.21 State Grid with states holding value of transmission delay and $pSucc$ . . 109
6.22 Transitions directions in terms of next state values . . . 109
6.23 State transition tendencies . . . 112
6.24 Theoretical and simulation values for settling time of E-MAC . . . 113
6.25 Theoretical and simulation based results for transmission delay values after 100s . . . 114

7.1 E-MAC lock-up example case . . . 117
7.2 Synchronisation example of two pulsed couple oscillators (fireflies) . . . 119
7.3 Desynchronisation example of two nodes . . . 120
7.4 Merging chain scenario . . . 123
7.5 Average Throughput CDF . . . 125
7.6 Achieved Running Throughput CDF . . . 126
7.7 Fairness CDF . . . 126
7.8 Fairness CDF . . . 127
7.9 Average End-to-End packet delay CDF . . . 128

7.10 Achieved Running End-to-End packet delay CDF 128
7.11 $txDelay$ value gradients accross routes 130
7.12 Change of $txDelay$ value over time 130
7.13 Average Overall Throughput over time 131
7.14 Running Throughput 131
7.15 Throughput per node on ad-hoc network give different number of sources (full-size) 134
7.16 Throughput per node on ad-hoc network given different number of sources (zoomed view) 135
7.17 Throughput per node on ad-hoc network median comparison for different number of sources 136
7.18 Fairness comparison on ad-hoc network for different number of source 137
7.19 Fairness median comparison for different number of sources 138
7.20 End-to-end packet delay comparison for different number of sources . . 138
7.21 Throughput per node for E-MAC+ for different $shiftFrac$ values given different number of sources 139
7.22 Fairness results for E-MAC+ for different $shiftFrac$ values given different number of sources 140

List of Tables

2.1 Some of Schedule-Based MAC Protocols 26
2.2 Some of Contention-Based MAC Protocols 27
2.3 Some of Hybrid-Based MAC Protocols 28
2.4 Some of AI applications in different areas of WSN 39

4.1 PSO Parameters 64
4.2 Simulation Parameters 64

5.1 PSO parameters 73
5.2 Simulation Parameters 75
5.3 Swarming MAC performance metrics 75

6.1 Simulation Parameters 94
6.2 E-MAC performance metrics 95
6.3 Throughput (Erlangs) (mean ± standard deviation) 96
6.4 End-to-end delay (mean ± standard deviation) 98
6.5 Fairness (Jain's Fairness Index) (mean ± standard deviation) 100

7.1 E-MAC+ merging chain scenario performance metrics 124
7.2 E-MAC+ ad-hoc scenario performance metrics 133

Acknowledgements

I would like to express my deepest gratitude to my brilliant supervisors Tim Clarke and Paul Mitchell. They provided me with never ending support, motivation, patience and guided me through this exciting field of research, allowing me to freely explore and grow.

My sincere thanks goes to my family - Žygimantas, Jolanda, Artūras and Evelina. Their support, love and patience made this work possible.

I am very grateful to the colleagues and friends in the Department of Electronics who created both a wonderful research environment and an enjoyable life in York.

Finally, I would like to acknowledge that this work has been funded by the Engineering and Physical Sciences Research Council (EPSRC).

Declaration

All work presented in this thesis is original to the best knowledge of the author. References to other researchers are provided where appropriate. This work has not previously been presented for an award at this, or any other, University.

The material presented in Chapter 5 is published as "Swarming Medium Access Control Protocol for Large-Scale Wireless Sensor Networks" at the 9th International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST), 2015.

The material presented in Chapter 6 is published as "The Emergence MAC (E-MAC) protocol for wireless sensor networks" in the Engineering Applications of Artificial Intelligence (EAAI) journal, volume 62, pages 17-25, 2017.

Chapter 1

Introduction

Contents

1.1	Motivation	14
1.2	Hypothesis	16
1.3	Outline	16
1.4	Publications	18

1.1 Motivation

Imagine millions of ants achieving complex tasks and termites building extraordinary structures (Figure 1.1a) or millions of starlings forming complex moving shapes (Figure 1.1b). These biological systems, comprising large number of simple entities, cooperatively achieve complicated tasks that could never be accomplished by single individuals. Without any central control these swarms and flocks self-organise to achieve robust, scalable and fault-tolerant task-allocation, brood-sorting, flocking and synchronisation behaviours. In addition they show the ability to react to and recover from disasters as well as to adapt to different or changing environments. The process behind these complex behaviours is usually emergence, distributed intelligence and stigmergy[1].

This thesis is motivated by natural collective intelligence. We explore the properties of emergence, Distributed Intelligence (DI) and apply the lessons learnt from biology to a practical communications engineering problem. We incorporate concepts from this so-called Swarm Intelligence (SI), to create emergent behaviours in a Wireless Sensor Network (WSN). We form distributed algorithms for Medium Access Control

[1] A way of indirect coordination/communication between agents

(a) Termite nest [2]

(b) Flock of starlings [3]

Figure 1.1: Examples of biological swarm systems

(MAC) protocols that show fault-tolerant, self-organised and autonomous operation with only simple logic and processing required.

> *"Networked sensors - those that coordinate amongst themselves to achieve a larger sensing task - will revolutionize information gathering and processing both in urban environments and in inhospitable terrain."* - one of the first WSN papers [4].

The WSN is an increasingly more important technology, due to the increased need for various monitoring applications in many different contexts [5]. While WSN can be considered as a standalone system, the development of Internet of Things (IoT) is one of the main applications of WSN [6]. Some WSN applications include: environmental monitoring [7], building and industrial process monitoring [8], habitat monitoring [9], vehicle monitoring [10], target detection and tracking [11]. Technological advances in wireless communications and electronics have reduced the cost of devices and have enabled the production of low-power multi-functional sensor nodes. This, in turn, has enabled WSN to be implemented in many different areas and has increased both network sizes and functionality [12] [13].

Depending on the application there are different challenges to be considered for WSN design. One of the main challenges across the majority of applications is efficient routing and transmission of data through the network. Many WSN applications are also limited by power availability. Therefore energy efficiency and power consumption are other important factors to consider. However, development in renewable energy or energy scavenging is benefitting this area significantly [14]. Scalability is a very important consideration for a larger WSN. Usually a WSN, as an ad-hoc network, presents scenarios where nodes can join and leave the network or fail at random times [15]. Moreover, it is extremely challenging to manage or synchronise large-scale WSNs. There is a need for mechanisms to achieve self-organisation and autonomous operation.

Just like natural swarms - the devices in a WSN are usually very simple individually, yet try to achieve complex and efficient operation at either routing or MAC layers. We believe that, inspired by natural collective intelligence, it is possible to solve the challenges of self-organisation, scalability and low-complexity in a WSN.

1.2 Hypothesis

The work in this thesis is guided by the following hypothesis:

"By combining some concepts taken from control engineering theory and Distributed Artificial Intelligence (DAI), it is possible to achieve performance levels comparable with conventional MAC schemes whilst exhibiting self-organisation with greater robustness, scalability and lower complexity."

MAC in communications and WSNs enable efficient operation to achieve good throughput, latency and fairness. Many schemes are proposed to deal with challenges presented by MAC layer. The majority of these are highly tailored towards specific scenarios and usually require tuning to achieve optimum operation. Inspired by natural, biological processes and the algorithms derived from these, we propose a variety of methods to approach MAC. Guided by the hypothesis, we focus on designing MAC protocols that are not only simple, but are able to adapt and adjust autonomously to the environment without any central control or significant processing power. The simulation experiments described in the thesis illustrate the performance and the benefits of these protocols.

1.3 Outline

The thesis is structured as follows:

- Chapter 2 is a research literature review of relevant fields. The concepts behind a WSN are firstly reviewed. This includes WSN structures, topologies and applications. An introduction to MAC then follows with a short survey of protocols used for WSNs. The fields of AI and DAI are then presented, summarising a number of different techniques. Biologically inspired intelligence is then introduced, including some concepts behind it. Finally, a survey of various applications of AI to MAC is provided.

- Chapter 3 presents the experimental methodology used to empirically evaluate the proposed MAC protocols. Simulation models for topologies, radio connectivity and propagation are presented. Performance metrics are detailed and explained as well as details of the comparison scheme employed. This chapter mainly acts as a supplementary chapter that we refer to when presenting empirical evaluation results.

- Chapter 4 presents the main concepts used throughout the protocol design in later chapters. We have noticed a striking similarity between the control engineering theory fundamentals and the operation of MAC. This inspired a slightly different approach towards MAC that is presented in this chapter. Results and discussions of empirical validation of such an approach are detailed. Simple (off-line) Particle Swarm Optimisation (PSO) is then applied to the proposed control method. The results and discussions that reinforce the benefits of the approach are detailed. The proposed approach forms a simple platform which can be used to apply a variety of AI and DAI techniques.

- Chapter 5 presents a Swarming MAC protocol, which is inspired by the prior work in the field of PSO. Using the control engineering theory approach presented in Chapter 4, we create and apply an online variant of a PSO to control the transmission rate of a WSN. An empirical evaluation is then carried out and results presented. The protocol provides a stepping stone towards even simpler and better performing protocols presented in Chapters 6 and 7.

- Chapter 6 presents our Emergence MAC (E-MAC) protocol. After applying and analysing the Swarming MAC protocol, the decision was made to take a step closer to Biology. Instead of using algorithms inspired by Nature, we look at the properties, behaviours, and actions presented in biological swarm systems, such as ant or bee colonies. This provides inspiration, motivation and methods for creating truly emergent behaviour in the MAC layer of a WSN. The design of E-MAC is presented in this chapter and its emergent properties are discussed. Empirical evaluation, analysis, results and parameter sweeps are then detailed and discussed.

- Chapter 7 presents the E-MAC+ protocol. After exploring the E-MAC protocol under a variety of conditions we enhance it by using some additional processes found in Nature. Exploitation of these additional simple rules significantly improves cooperation between the nodes that employ E-MAC+. Therefore better overall performance in more complex scenarios is achieved. Extensive simulation results and discussions are presented.

- Chapter 8 concludes the thesis. A summary of the contributions is presented and the hypothesis is revisited. Recommendations for further work are then provided.

1.4 Publications

The material presented in Chapter 5 is published as "Swarming Medium Access Control Protocol for Large-Scale Wireless Sensor Networks" at the 9th International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST), 2015.

The material presented in Chapter 6 is published as "The Emergence MAC (E-MAC) protocol for wireless sensor networks" in the Engineering Applications of Artificial Intelligence (EAAI) journal, volume 62, pages 17-25, 2017.

Chapter 2

Literature Review

Contents

2.1 Wireless Sensor Networks 20
2.1.1 Applications 20
2.1.2 Devices 21
2.1.3 Scenarios and Topologies 21
2.1.4 Communications Architecture 22
2.1.5 Constraints and Challenges 23
2.2 Medium Access Control 24
2.2.1 Schedule-Based Medium Access Control Protocols 25
2.2.2 Contention-Based Medium Access Control Protocols 25
2.2.3 Hybrid Medium Access Control Protocols 25
2.2.4 Limitations and Challenges 26
2.3 Multi-Agent Systems and Distributed Artificial Intelligence 29
2.3.1 Deliberative Multi-Agent Systems and Machine Learning . . 30
2.3.2 Reactive Multi-Agent Systems 34
2.3.3 Structure of a Multi-Agent System 35
2.3.4 Properties of Bio-Inspired Systems 36
2.3.5 Wireless Sensor Network as Multi-Agent System 37
2.4 Artificial Intelligence in Wireless Sensor Networks 38
2.5 Conclusions 40

This chapter is a background review that inspired the hypothesis and also informed the work carried out to prove it. Firstly we look into simple concepts behind Wireless Sensor Networks (WSNs), their applications and scenarios. In addition, we summarise some of the popular MAC protocols used in such networks. The chapter then continues with a review of different Artificial Intelligence (AI) and Distributed Artificial Intelligence (DAI) models and techniques. We present our opinion on the suitability of these for particular aspects of WSNs. A summary of available AI and DAI applications to WSNs is then provided. We also present some concepts behind biologically inspired systems as Nature is one of the greatest examples of Distributed Natural Intelligence (DNI). These examples from Biology drive our main ideas and contributions in this thesis in the following chapters. This chapter is simply a broad overview of the related fields and the further chapters that present the design of protocols will provide an in-depth review of the relevant concepts used.

2.1 Wireless Sensor Networks

Addressing the simplicity, cost, distributed nature, rapid-deployment and fault-tolerance of WSNs opens up a wide range of applications [5]. The increased need for monitoring and control of large and small scale environments further increases the interest and development of WSNs [16]. This section gives an overview of WSNs. Applications of WSNs, their structure, scenarios and constraints are summarised.

2.1.1 Applications

Military

WSNs can be used for a variety of military applications. Battlefield monitoring and tracking using WSNs allows for better strategic decisions to be made: soldier monitoring and tracking [17], sniper detection and localisation [18]. Other applications can provide defensive benefits: early attack reaction sensors [19], remote chemical sensors [20]. WSNs can also be applied to surveillance [21] and public safety [22].

Environmental

Large and small scale environmental monitoring can be achieved by using WSNs. Deployed sensors allow collection of various data about the environment. Existing systems use WSNs for agricultural monitoring to increase growth and quality of plants and cattle [23, 24]. Scientific monitoring can also be achieved by deploying sensors in remote hostile environments [25]. Meteorological WSNs can be implemented to monitor weather conditions [26].

Industrial

WSNs provide a lot of benefits for industrial automation applications as well. Elimination of cabling can be achieved with wireless FieldBus [27]. WSNs can also be used for monitoring and control of industrial process [28, 29]. These provide cheaper and easier to implement industrial automation and management.

Disaster Management

By using hydrological, meteorological and landslide nodes, an effective disaster management system can be created. This allows for early disaster detection systems to be implemented [30, 31, 32, 33]. Moreover, WSNs can be applied in a variety of search and rescue operations [34].

2.1.2 Devices

Existing WSN nodes usually consist of several main hardware components: a processor and memory, transceiver, power supply and sensors [35]. Communication usually uses the most power in these devices. Therefore efficient control of transmission is required. Power is usually supplied by a battery [36] or a super-capacitor [37]. Nevertheless, examples exist where the wireless sensors are connected to a power grid [38]. In the majority of environmental and disaster management applications, wireless nodes are deployed on a large scale [39, 40, 41]. There may be hundreds or thousands of these nodes deployed in an area which can be hostile and cause nodes to fail [39]. This calls for low-cost, low-simplicity nodes. Lower simplicity and cost, as well as large numbers of devices calls for simpler protocols. These require advanced techniques to self-organise and maintain performance without the need of additional supervision or significant hardware capabilities.

2.1.3 Scenarios and Topologies

WSNs, depending on their application, can be deployed both randomly or in a very organised way. Precision applications such as farmland monitoring or industrial monitoring will tend to have well planned deployments to achieve the precision and efficiency required [42]. Environmental monitoring is usually less restricted and is not impacted by less organised deployment [40]. It is not always possible to deploy nodes precisely due to terrain limitations or the ability to reach the area easily. In some cases these nodes can be simply scattered over an area from an aerial vehicle [43].

At first it may seem that the WSN is inherently an ad-hoc network of nodes spread over an area. While many industrial, agricultural and military applications satisfy this argument, there are equally as many WSN applications on chain-like ad-hoc networks as well. Applications of environmental monitoring of coastlines, riversides, roadsides, power lines and even smart cities may require short range simple nodes to be deployed in a chain like fashion. This may result in very large chain networks which become more and more complex to organise, maintain and efficiently pass data through. Nevertheless the data provided by such networks can be extremely useful for scientific, planning and disaster management purposes [5].

2.1.4 Communications Architecture

Communication system architectures for WSNs usually contain five architectural levels and follow the OSI Model. The elements are: application layer, transport layer, network layer, data link layer and physical layer [5]. The physical layer provides functionality to transmit data stream over the channel (the physical medium). This layer controls the modulation, coding, frequency selection, transmission and reception [44]. The data link layer ensures reliability of a link, manages the channel sharing between the nodes (MAC) and error control. The network layer deals with the routing in the WSN. It defines and manages paths that data should follow through the network to the sink node. The main function of the transport layer is to provide reliable operation where a lot of different protocols (access to other networks) are being used. However the transport layer is rarely fully exploited. [5]. The application layer obtains the data and passes it to lower layers for transmission. It also receives data from lower levels and processes (or acts on) it.

Each of these layers has a rigid structure and data flow between them (Figure 2.1). Protocols are optimised to work efficiently for each particular layer. In some cases, a protocol in one layer can bottleneck protocols in the other layers. As a consequence there is some research for cross-layered protocols for WSNs to enhance performance [45].

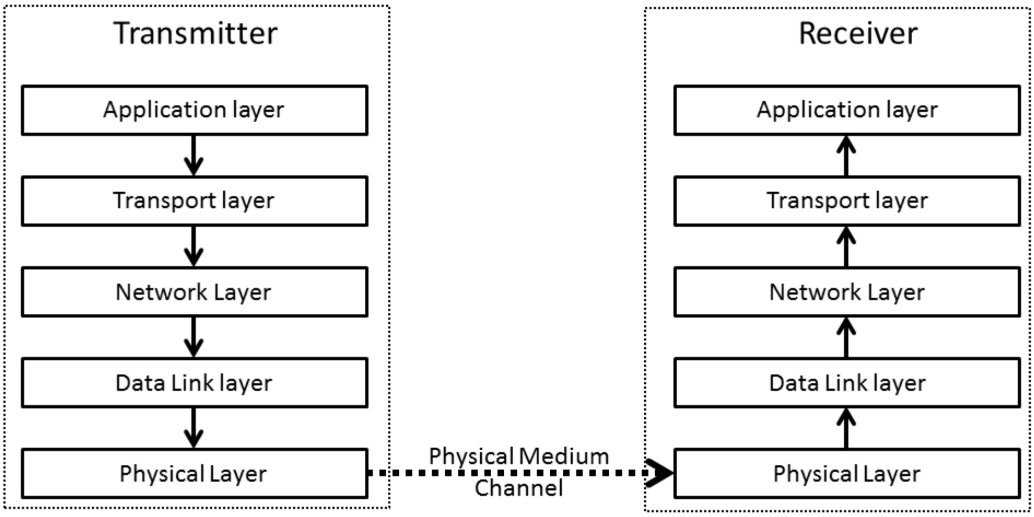


Figure 2.1: Communication layers of WSN

2.1.5 Constraints and Challenges

All types of communication networks and devices exhibit similar constraints such as: interference, noise, transmission power limitations or security. WSNs, however, can present some additional specific requirements and constraints:

- Financial and simplicity - most WSN applications involve many nodes with mobile power sources, randomly deployed over large areas. Usually the nodes will not be reusable and will be deployed once until they fail. Some applications require great spatial and temporal resolutions. Therefore the cost and simplicity of the nodes must be taken into account.

- Operation under varying conditions and deployments - the initial deployment of the nodes may differ with environmental conditions. Given a large WSN, maintaining and tuning nodes becomes a costly and challenging task. To keep the costs low and the performance high, the protocols have to be able to adjust themselves to the environment (self-organise). Once deployed, the operation has to be maintained and so re-organisation and fault-tolerance is preferable. As the number of nodes in the network can be continuously varying (additionally deployed or replaced nodes and failures), they must be able to adapt [46]. Some applications of WSNs require nodes to be capable of dealing with dynamic environments. Nodes might be moving around all the time. MAC and routing protocols have to adjust and adapt to these changes.

- Fairness - given the large number of nodes operating in a single network, maintaining fairness between the nodes can become an issue. Given critical and urgent information, it is important that each node has fair access to pass its data to the sink. Otherwise the integrity and usability of the WSN can be affected.

- End-to-end delay - in the case of disaster monitoring, it may be crucial that data is received at the sink in a timely manner. Protocols have to be able not only to deliver fair operation but also to offer good Quality of Service (QoS) and quick transfer of data through the network.

While each of these components may seem reasonable and simple to achieve individually, it becomes significantly more challenging when they are combined. Large numbers of simple and cheap nodes have to be able to form a WSN and maintain it. They are required to not only pass data but also guarantee fairness, low end-to-end delay and high throughput. Given the size of the network, central control becomes cumbersome. Also, synchronisation is hard to achieve due to the simplicity of hardware and network size. This is where AI, DAI and biologically inspired algorithms can contribute. Many proposed AI algorithms and natural processes exhibit simplistic implementation, yet very complex outcomes. Systems with simplistic rules are able to somehow reason and make decisions. Therefore they offer potentially significant benefits when applied to communications. In Section 2.3 we present a review of a number of AI techniques and express our opinions regarding their suitability in this context.

2.2 Medium Access Control

This section focuses on introducing Medium Access Control (MAC) and reviewing some classic and state of the art protocols. It provides an overview of appropriate available techniques and the background to the field.

Some of the major challenges and constraints for WSNs are caused by the physical medium. Therefore efficient Medium Access Control (MAC) is required [47] for enhancing throughput, latency, fairness and energy usage.

Some classic multiple access techniques are: Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA), Code Division Multiple Access (CDMA), Orthogonal Frequency Division Multiple Access (OFDMA) and Space Division Multiple Access (SDMA). MAC protocols use these multiple access techniques to achieve their goals for different applications. The most common multiple access technique for WSN is TDMA [48]. Due to the cost, size and simplicity of WSN nodes, most of them are only able to use one frequency and one of the simplest means to enable and control multiple access is separating packet transmissions in time. MAC protocols can also be categorised as contention-based and schedule-based [49]. However there is ongoing research into hybrid MAC protocols [50] as well.

2.2.1 Schedule-Based Medium Access Control Protocols

Schedule-based MAC protocols establish a schedule between the nodes so that each node knows when it should be listening, transmitting or staying inactive to enhance performance. There are a variety of approaches for schedule based protocols, some of which are described in Table 2.1. We chose to include these particular protocols here due to their popularity and to show a variety of different approaches.

2.2.2 Contention-Based Medium Access Control Protocols

Due to the distributed nature of WSN, contention-based MAC protocols seems to be a more intuitive approach to achieve scalability, self-organisation and fault-tolerance. ALOHA and slotted ALOHA are classic contention based protocols [54]. These are a very simple to implement and allow users to transmit data whenever they have it available. This causes the system to operate well under very small traffic loads but as the traffic level increases performance drops significantly. One of the most widely used protocols is Carrier Sense Multiple Access protocol (CSMA). Popular variants also provide Collision Detection (CSMA/CD) in wired networks and Collision Avoidance (CSMA/CA) [47] in radio systems. Its simplistic nature and benefits of avoiding and detecting collisions caused a lot of new MAC protocols to arise. Some of the contention-based MAC protocols are provided in Table 2.2.

2.2.3 Hybrid Medium Access Control Protocols

Hybrid MAC protocols can achieve even better performance [60]. These protocols bring together ideas from both contention- and schedule-based protocols. The benefits of both methods can be utilised to address bottlenecks. However, the drawbacks can be present as well. Some examples of such protocols are presented in Table 2.3.

Protocol Name	**Employed techniques, schemes**	**Operation and Properties**
Low-Energy Adaptive Clustering Hierarchy (LEACH) [51]	TDMA, CDMA, clustering	Cluster-based protocol that achieves longer network lifetime by distributing energy consumption between nodes. Elected cluster heads manage clusters and create schedule. This requires nodes to be of higher complexity and can hinder performance at large scale as each cluster head has to be able to directly reach Access Point (AP).
Priority-Based MAC Protocol for Wireless Sensor Networks (PRIMA) [52]	LEACH	PRIMA is an improvement on LEACH to provide better QoS by introducing priority to packets. However same benefits and drawbacks are carried over from leach.
Power-Efficient and Delay-Aware MAC Protocol for sensor networks (PEDAMACS) [53].	CSMA, TDMA	A complex protocol that operates by collecting data on topology, learning it and distributing schedules across all nodes. This is a very centralised approach which can provide very efficient scheduling. However it requires a lot of overheads to set-up and is not easily adaptable which can hinder its performance on large-scale networks.
Identity-Based MAC Protocol (ID-MAC)	TDMA, pseudo random functions	Each node uses a pseudo random function to decide when to transmit or receive based on unique identifier. Nodes share this counter during set-up phase, therefore transmission and reception of neighbours can be predicted. Synchronisation is required for this to work and given varying environment the pre-set pseudo random functions may not utilise MAC to its potential. This renders it less effective on large-scale WSN.
Traffic-Adaptive MAC protocol (TRAMA)	TDMA, Adaptive Election Algorithm (AEA)	Each node exchanges neighbourhood information with its neighbours. AEA is used to switch nodes to low-power based on gathered information. Therefore schedules are created based on traffic levels which increases utilisation of the medium and energy. Significant overheads and running of AEA is required which can hinder deployments and operation on large-scale networks.

Table 2.1: Some of Schedule-Based MAC Protocols

2.2.4 Limitations and Challenges

WSNs can present very different challenges for MAC depending on the application and deployment. MAC protocols are usually designed to fulfil the needs of particular

Protocol Name	**Employed techniques, schemes**	**Operation and Properties**
Sensor MAC protocol (S-MAC) [55]	TDMA, duty-cycling, RTS/CTS	S-MAC employs fixed duration duty-cycle (sleep and listen/transmit) at each node. Using the neighbourhood information, each node can make sure no information will be missed by adjusting their duty-cycles. This scheme aims to minimise energy consumption instead of increasing throughput or delay. It is a complex protocol that also requires synchronisation to achieve its full potential.
Timeout MAC protocol (T-MAC) [56]	S-MAC, TDMA, duty-cycling, RTS/CTS	T-MAC is an enhancement of S-MAC. It introduces adaptive duty cycle. If there is no activity on the channel, nodes will go to sleep for a pre-set timeout, hence saving even more energy. It shows improved performance in dynamic environments.
Dynamic Sensor MAC protocol (DS-MAC) [57]	S-MAC, TDMA, duty-cycling, RTS/CTS	This protocol employs dynamic duty cycle which is varied with traffic load. By observing one-hop latence nodes interpret traffic levels and increase or decrease duty-cycles by factor of 2. This protocol is able to better adapt to changing traffic levels.
Versatile Low Power MAC protocol (B-MAC) [58]	TDMA, CSMA, preamble	B-MAC is a MAC protocol based on CSMA that provides low-power network operation but still ensures collision avoidance. B-MAC adds a preamble to packets that is long enough for neighbours to sense it. Each node then wakes up only for a short amount of time to check the Received Signal Strength Indication (RSSI) on radio. If the RSSI indicates there is a transmission, the device powers up to receive the packet. Otherwise the device goes back to sleep. It provides adaptable and low-power operation.
Short Preamble MAC Protocol(X-MAC) [59]	TDMA, preambles	X-MAC uses a similar approach to B-MAC by the addition of preambles to packets to allow nodes to identify when they should stay listening. However, it modifies the long preamble to many short ones. Each short preamble contains a target node ID. When nodes listen to a preamble, they can determine whether a packet is destined for them. If it is not, a node can simply go to sleep until the next wake up. There are also small pauses between preamble packets to allow a receiver to acknowledge them and start the data transmission. Significant energy savings can be observed.

Table 2.2: Some of Contention-Based MAC Protocols

Protocol Name	**Employed techniques, schemes**	**Operation and Properties**
Zebra-MAC protocol (Z-MAC) [61]	TDMA, CDMA, DRAND	Z-MAC is a hybrid protocol which combines features of TDMA and CDMA. During a setup phase, this protocol deals with neighbour discovery. Each node finds their two hop neighbours and then uses Distributed Randomised TDMA scheduling (DRAND). The schedule is created such that the hidden terminal problem is avoided. No slot in the two hop neighbourhood is assigned to more than one node. To improve utilisation in the case of uneven traffic load, Z-MAC also allows nodes to borrow slots from neighbours if they do not have anything to transmit.
Adaptive CSMA/TDMA hybrid MAC protocol [50]	CSMA, TDMA	This protocol focuses on improving both energy consumption and throughput of WSNs. The basis of this protocol is the IEEE 802.15.4 standard. It applies modifications to the standard to allow the algorithm to decide and switch periodically between CSMA and TDMA. The algorithm looks at network load state and average channel utilisation and calculates what the proportion between CSMA and TDMA time periods should be. For different traffic loads, different advantages of CSMA or TDMA can be utilised

Table 2.3: Some of Hybrid-Based MAC Protocols

scenarios but usually fail to deliver good performance in different ones. While the classic and popular protocols such as 802.11 IEEE standard (discussed in more detail in Section 3.3) deliver fairly steady performance in a variety of situations, many of the state-of-the art protocols are tailored towards particular topologies or goals. Schedule-based protocols can utilise the medium extremely efficiently if appropriate synchronisation and information sharing is available. However, they fail if these needs are not met. Contention-based protocols usually tend to provide predictable throughput and adapt to the environment, but they are not able to utilise the medium as efficiently. Hybrid MAC protocols combine the strengths of both contention and schedule based methods to alleviate some of these problems. However, they often take some of the drawbacks from both. Often hybrid MAC protocols use smart techniques of switching between the two approaches to reduce negative effects. Some protocols, however, require more complex hardware to combine these different techniques. In fact, many hybrid MAC protocols use the IEEE 802.11 standard as a fall-back if the required conditions are not met. Similar MAC layer issues and challenges for WSN can be found in different applications as well [62].

The application of AI in such situations for MAC can be very beneficial. It can enable the learning/reasoning process and allow a protocol to adapt to different environments and tailor itself for better performance. This can potentially allow deployment of such a MAC protocol to different scenarios without needing adjustment. In addition it can enable re-organisation and fault-tolerance throughout operation, while maintaining high performance levels.

2.3 Multi-Agent Systems and Distributed Artificial Intelligence

The Multi-Agent System (MAS) is a concept for a system comprised of entities (biological or computational), referred to as agents, and an environment in which they are placed. These agents interact trying to achieve some goal. Many biological systems such as insect colonies or flocks of birds can be considered as a MAS. Each entity can be considered to exhibit Natural Intelligence (NI) - an intelligent acting biological entity. When combined together these entities exhibit larger Distributed Natural Intelligence (DNI) and present a MAS which achieves goals that are beyond the capabilities of the single entity. Artificial Intelligence (AI) defines an intelligent artificial node that exhibits intelligent behaviour, often inspired by NI. A combination of such nodes then exhibits a Distributed Artificial Intelligence (DAI), which is often inspired by DNI.

"Distributed Artificial Intelligence (DAI) is the study, construction, and application of multi-agent systems, that is, systems in which several interacting, intelligent agents pursue some set of goals or perform some set of tasks." [63].

The structure and behaviour of a WSN closely matches the MAS. Many identical (or similar) wireless sensor nodes intelligently interact with each other to achieve a higher level goal. This, therefore, draws our attention towards a review of MAS and its components.

MAS can be classified into two major classes: reactive and deliberative (or cognitive). We review these two areas and look at examples of AI and DAI expressed in such systems.

2.3.1 Deliberative Multi-Agent Systems and Machine Learning

A deliberative MAS is considered to be a system comprised of agents that are able to perceive and reason about their environment in some way. This usually implies some aspects of learning, building up an understanding of the environment or the problem domain and utilising this knowledge. It is possible to apply Machine Learning (ML) to a MAS.

ML can be defined as a general inductive process that builds up knowledge by learning. Learning is directed with pre-set rules or data or through observation. ML algorithms usually operate by building explicit or implicit models of the system they are trying to learn. Models may be built based on inputs and some fitness/reward function. This allows programs or devices to learn what actions are more rewarding or better to be used in the environment. It allows for adaptive and efficient operation.

Reinforcement Learning

One of the more popular ML approaches is Reinforcement Learning (RL) [64]. RL algorithms are based on a trial and error approach. This was inspired by biological systems and their learning processes. When an agent takes any action, there is some reward (positive or negative) associated with the outcomes. This allows the agent to develop a knowledge base about actions - some of which are more rewarding than others. This policy allows for the selection of best rewarding actions and consequent optimisation of performance. It is a very powerful process as it tries to build up an indirect model of the environment and use it to improve performance (assuming Markov property holds). RL has two major operational phases associated with it: exploration and exploitation. Exploration is the process of building policies and learning about the environment. Exploitation is the process of using these policies to act on the environment. This also presents major challenges for RL: the balance between exploration and exploitation. [65]

Fuzzy Logic

Fuzzy logic is an AI model inspired by human reasoning [66][67]. Human reasoning tends to include uncertainty about objects. This uncertainty can be represented by the commonly used words such as *large, small, many, few, most, least*. Fuzzy logic introduces approaches for dealing with fuzzy sets (linquistic information). Fuzzy sets are simply classes with unsharp boundaries [67]. In fuzzy sets, objects can be a partial member of a set (for example person can be 80% tall and 20% short). This allows characterisation of a system in a linguistic way that connects to human knowledge. Therefore, it enables the possibility for implementing approximate reasoning. Fuzzy rules are usually defined as: *if* antecedent *then* consequent [1]. However, it is different from general classical threshold values, as these are defined in linguistic terms. To apply these methods to real systems a so called Fuzzy Inference System (FIS) is used.

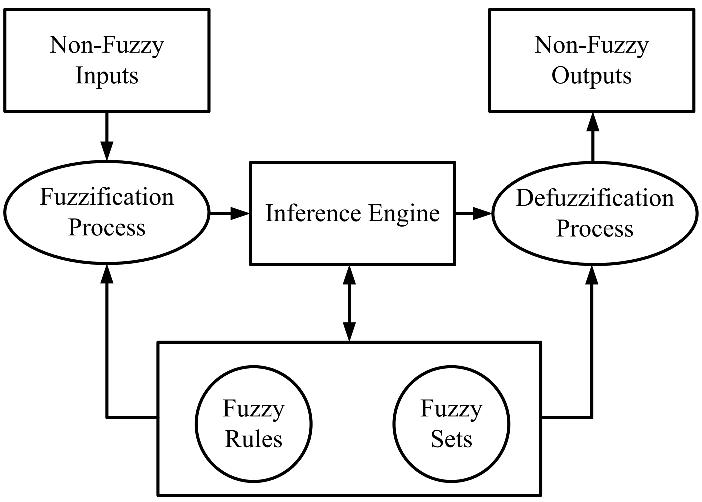


Figure 2.2: Fuzzy inference system [1]

FIS uses fuzzy rules and fuzzy sets with inference engine to map the system inputs to outputs. In order to produce fuzzy inputs from real sensor inputs a process called fuzzification is used. An inference engine then uses the knowledge-base built of fuzzy rules and sets to reason and produces fuzzy outputs. To convert fuzzy outputs to non-fuzzy (crisp) outputs a similar backward process of fuzzification is being used - defuzzification. These processes use information from fuzzy rules and sets to determine the outputs. Figure 2.2 shows the basic block diagram of a FIS [1]. Basically the FIS tries to determine a non-fuzzy value that represents action to be taken if certain inputs are given [68].

Fuzzy logic has been successfully used in pattern recognition, control systems and signal processing. Nevertheless, application of such methods to WSNs could be challenging. The complexity of such methods is high. Also the whole idea behind fuzzy logic is more applicable to classification whereas MAC requires a precise control of transmissions.

Evolutionary Algorithms

Evolutionary Algorithms (EA) are inspired by the natural evolution of biological systems. Biological systems over many years have evolved to achieve better survivability (survival of the fittest [69]). Through many years the processes of mutation, reproduction, recombination and selection have affected biological systems. This caused species to change significantly to adapt to the environment and new species to appear. EA seeks to imitate the process of altering genes in the chromosomes. Each chromosome contains many genes which define different properties of the specie. EA represents solutions as chromosomes comprising many genes. At the initialisation of the algorithm, a random population of chromosomes is produced. The solutions encoded through the chromosomes are tested against a cost or fitness function. The genes in the chromosomes can mutate and recombine during reproduction. After each algorithm iteration, the fittest chromosomes are chosen to continue the reproduction process. On completion, solutions are presented as those corresponding to the fittest chromosomes. Evolutionary algorithms can be divided into several major classes:

Genetic Algorithms (GA) - models genetic evolution. It works in an iterative manner by generating new populations from old ones. Stochastic operations of mutation, selection, recombination are applied as well as random initial population. This class of algorithms is mostly derived from nature. [70]

Genetic Programming (GP) - is a type of GA where each individual in a population is actually a program. This means that each GP comprises of a set of instructions that are evaluated via the fitness function to measure their performance. It can be used to optimise a population of computer programs. [71]

Evolutionary Programming - is similar to GP. While GP allows changing the set of instructions, Evolutionary Programming only allows changing numerical parameters during evolution. The process of mutation is most significant here. [72][73]

Evolutionary Strategies - model strategy (or parameters) to steer the evolution. Using the ideas of adaptation and biological evolution it can optimise the parameters to achieve certain system goals or performance.

EAs can provide significant benefits in optimisation tasks. However the need for many iterations renders them difficult to be used for WSNs in an online way. Nodes could be built to evolve their schedules over-time but it would need some central control to define which nodes are fittest and the process would take too long to be feasible. Evaluating such network online would be extremely challenging.

Neural Networks

Neural Networks (NN) are inspired by the complex network of neurons in a brain which is able to learn and memorize. Each neuron is a very simple mechanism which simply receives and sends signals. A combination of many neurons editing and passing signals create complex learning behaviours. NN algorithms build a network of neuron weights that are then summed and passed through non-linear functions (to form an artificial neuron network). Acquired data sets are used in the process to train neurons (obtain weights). After a NN is trained, it can start predicting values, recognising patterns or approximating functions. Trained neurons are able to distinguish features in the input data [74]. There are many applications where this can be exploited, such as predicting future data from available data and training systems to recognise properties of a complex environment. While being extremely popular in number of areas NN usually requires the availability of large sets of data to achieve required outcomes and decisions. Some good results could be achieved by collecting data from number of different WSN deployments and teaching the NN. It is possible that such a NN could learn trends and how to respond to different behaviours on

MAC to achieve good results. However, collecting and classifying data to be used for the learning process of NN would be a serious challenge by itself.

2.3.2 Reactive Multi-Agent Systems

Where many simple agents react to their environmental conditions or interactions directly, following simple behavioural rules, we have a reactive MAS. Many biological systems do this. For example, swarm intelligence or DNI, such as ants or bees, can be considered as reactive agents that respond directly to stimuli.

Artificial Immune Systems

Artificial Immune Systems (AIS) are inspired by biological immune systems that protect higher organisms from threats (pathogens). Biological immune systems can be categorised as innate or adaptive. An adaptive immune system aims to eliminate or prevent pathogen growth. They learn from previous encounters with pathogens. Dealing with same pathogens then becomes more efficient over time. Once a threat is identified, specialised cells and processes are used to destroy it. In contrast, the innate immune system does not have any long-term memory of known pathogens. It provides an immediate response to a pathogen if an adaptive immune system fails to destroy it. It uses a complex processes of recruiting cells to deal with infections or activating certain chemical processes in body to clean or remove unwanted pathogens. Biological immune systems in general are complex and distributed systems that can adapt to a variety of situations. AIS tries to model this behaviour. [75]

Simplicity of the entities in AIS and their simple interaction method present a viable option for application towards MAC in WSNs. A messages mimicking the immune network could be passed around to initiate reporting processes from relevant parts of networks or destroying inefficient routes. However scheduling and MAC control with such methods could be extremely challenging. AIS relies on emergence and it makes it hard to design such system. Nevertheless, by mimicking the behaviours observed in biology the end result can be predictable.

Cellular Automata

Cellular Automata (CA) are mathematical entities that aim to model natural cell systems. They take inspiration from the complex behaviour of many simple locally-interacting agents [76]. A popular example of such a system is Conway's Game of Life [77]. Here, the environment is a simple two-dimensional orthogonal grid made of square cells. Cells have only two states - dead or alive (1 or 0). These cells only interact with their 8 neighbour cells. The state of a cell is defined by state of the neighbours according to very simplistic rules. This causes extremely complex behaviours to emerge which are not encoded into any one agent (pulsars, gliders). The system even shows self-replication and migration. This gives high motivation for developing systems that exploit emergence and distributed artificial intelligence to achieve complex behaviours. Such methods have been mainly applied in cryptography [78] but applications to wireless networks are also present [79].

Swarm Intelligence

Swarm Intelligence (SI) is inspired by the social behaviour of biological swarms such as ant, bee or termite colonies. These insects together show self-organised, fault tolerant, decentralised and complex behaviour. If we take an example of the ant colony, each ant is a very simple agent that could not survive on its own. Each agent follows very simple behavioural rules such as following highest pheromone (scent) level paths to a food source and bringing back the food by leaving pheromone on the path. The stigmergy behind pheromones allows ants to find optimum paths to the food source without any centralised coordination. Moreover, if the path is disturbed they quickly adapt to it. Similar rules for moving dead ants, taking care of eggs and the queen takes place. This leads to highly complex swarm behaviour where many thousands of simple ants cooperate to achieve tasks such as the building of complex nests [80]. This complex behaviour from a distributed system has inspired many optimization algorithms such as Particle Swarm Optimisation (PSO) [81], ant colony optimization [82], the bees algorithm [83].

2.3.3 Structure of a Multi-Agent System

The MAS can also be distinguished by different interaction structures in the system. Decentralised systems can still contain some centralisation - for example clusters of agents with one agent controlling each cluster and reporting to one central node. Distributed approaches have no central control units and no clusters with cluster heads. Figure 2.3 shows comparison of these systems where each white circle represents an agent and dark circle represents the central agent.

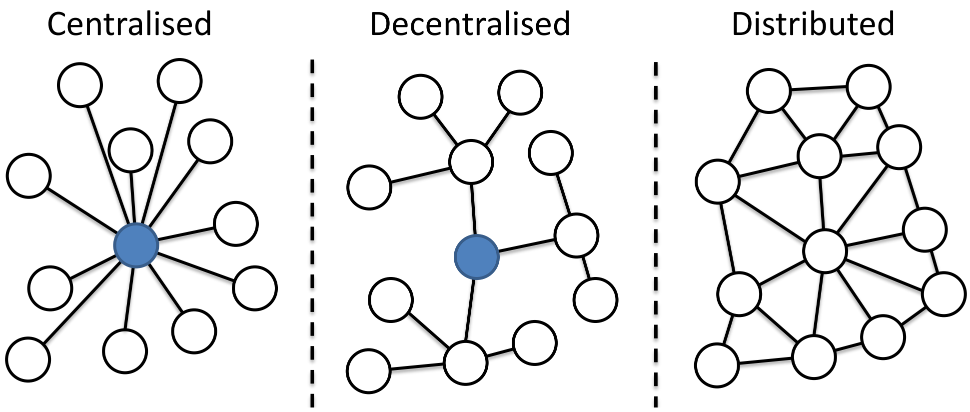


Figure 2.3: Comparison of Centralised, Decentralised and Distributed systems

Centralised systems have benefits of global control, building up models based on information collected centrally. However, the tradeoff lies in fault-tolerance. The centralised system presents a single point of failure, whereas a distributed system does not. In addition centralised systems can suffer from larger bandwidth requirements, overheads, latency and potential high complexity. Decentralised systems are hybrid and can be tailored to exhibit benefits of both structures in some applications.

2.3.4 Properties of Bio-Inspired Systems

As the majority of the outlined AI and DAI methods are inspired by biological phenomena, we now review some attractive properties they exhibit. The same properties will motivate the MAC protocols presented later in this thesis.

Nature is by no doubt the most proficient engineer. Solutions presented by biological systems are usually simple, extremely robust, fault-tolerant and adaptable. This drives the motivation for many scientists to design algorithms and protocols using the principles found in Nature [84].

Self-Organisation

Self-organisation is a concept that is very desirable in many different engineering systems. It presents as the ability of a system to initialise and configure itself under different operating conditions without supervision.

Emergence

In biological systems simple rules and interactions of many agents, acting together, form large complex entities, that exhibit complex behaviours. This is emergence. No single entity has either the goal or an explicit behaviour that points directly to the achieved collective system. The solution is usually flexible and robust. It also shows self-organisation and fault-tolerant behaviour while adapting to different conditions.

Emergent behaviours are impossible to design or sometimes predict, but emergence can be created.

Robustness

If we take the same example of an ant colony and disrupt it (removing food sources, destroying part of the nest), we will see how quickly and seamlessly it adapts. An ant colony is usually comprised of highly specialised workers - some are more efficient with nest duties, others with foraging or fighting. However, once a disruption occurs, for example death of large number of foraging ants, the whole colony reorganises without any central coordination. Otherwise specialised workers start taking up foraging. Each ant makes a decision individually as well, therefore there is no one point of failure that could kill the colony. In fact, killing the queen can hinder the nest as no reproductive organ is left. Nevertheless the rest of the nest keeps operating as normal and worker ants replace the queen in some species [85]. This is a very desirable property for a WSN. A WSN is comprised of many simple nodes. Many of these might fail over time, but we still want the network to continue operating.

2.3.5 Wireless Sensor Network as Multi-Agent System

WSN topologies and communication very much resemble the distributed MAS principles. Actually, the sensor nodes in WSN resemble the agents themselves as they have some means of sensing environment or communication channel, some processing power and can act. Therefore a WSN can be considered as a MAS. Theory behind MAS and DAI can be brought to WSNs not only for data collection but to improve the communications performance as well. The properties of self-organisation and robustness is also very desirable for WSN. Cognitive MAS could be very beneficial to WSN due to its ability to learn and defined design methods. ML is one of very promising areas regarding application to scheduling and resource allocation. Extremely well performing protocols have been proposed for MAC control in WSN and cellular networks. One such example is Q-ALOHA where ML is applied to make nodes learn which slots in a framed and slotted ALOHA should be used [86]. Another example is the resource allocation problem where nodes have to learn which frequencies to use in order to get data through [87]. These show near optimal performance once settled, but usually required synchronisation and some setting up. On larger-scale networks, however, such methods might become harder to achieve. The slot sizes would need to be varied to achieve optimum operation. Synchronisation might not be an option as well. Cognitive MAS can present both complex and simple implementations and is usually pre-defined for specific tasks that it tries to optimise.

Reactive MAS, however, offers ability to emerge into self-organised, robust and well performing system. However, the design is not intuitive and the more sensible method is to try and use the biological phenomena that was already researched in other areas or mimic behaviour of real biological systems. The concepts of SI is in fact what drives our thesis and proposals. The inspiring results achieved by swarms in nature (insect colonies, flocks of birds) are extremely complex, fault-tolerant and self-organising. In addition the topologies usually resemble the WSN. Many simple nodes, communicating with each other in some way, sensing and observing the environment very much resemble wireless nodes deployed in WSN. SI methods also show a high level of collective behaviour which is desirable for WSNs. According to distributed nature of WSN it should be possible to achieve wanted behaviour from simple interactions. However, emergence is not easily achieved as there is no representation of it. Therefore the design process is not intuitive.

2.4 Artificial Intelligence in Wireless Sensor Networks

There are many examples of AI application to WSN in several different areas. The majority of these are usually offline algorithms. They tend to solve deployment, routing or sometimes even scheduling problems before a WSN is deployed and switched on. Routing is probably the largest field that uses AI, DAI methods extensively. The principles behind routing tends to be similar to many different biological processes (path-finding, ant foraging). Scheduling and MAC layer on the other hand benefits from learning algorithms that are inspired from learning processes observed in nature. Nevertheless, there seems to be lack of applications of swarm intelligence in the scheduling fields and MAC. The application of swarm and collective intelligence is not intuitive as most solutions arise through emergence. It is a difficult task to obtain a desired behaviour through emergence but it is one of our ambitions to employ it. While emergence is hard to predict, the behaviour often shows significant organisation, adaptation and robustness. An example of such outcomes can be observed in nature: simple individual ant behaviours emerge into complex colony survival, firefly reactions emerge into synchronisation of great number of entities, simple interactions between bees emerge into a complex and efficient nest maintenance. A comprehensive survey of AI methods used in WSN is given in [1] (Table 2.4 is based on [1] and additional collected information).

Area	**Class of AI**	**Methods and references**
Deployment	Fuzzy logic	Fuzzy-Deployment [88]
	Swarm Intelligence	Sequential PSO for deployment [89]
		PSO for Urban Traffic Surveillance System [90]
	Reinforcement Learning	RL for service directory placements [91]
Localisation	Evolutionary Algorithms	Estimated localization using GA [92]
		Two-Phase localisation [93]
	Swarm Intelligence	BFA for localisation [94]
Routing	Reinforcement Learning	Q-Routing [95]
		Q-PR [96]
		SAMPLE [97]
		FROMS [98]
		DRQ-Routing [99]
		RLGR [100]
	Swarm Intelligence	Ant-based routing [101] [102]
		Various PSO based-routing algorithms [103] [104]
	Neural Networks	Neural network based routing [105]
Scheduling	Reinforcement Learning	Q-Aloha [106]
		RL-MAC [107]
	Evolutionary Algorithms	Active interval scheduling [108]
	Artificial Immune Systems	AIS based energy efficient algorithm [109]
Security	Fuzzy Logic	FS-MAC [110]
	Neural Networks	Neural Network based MAC [111]

Table 2.4: Some of AI applications in different areas of WSN

2.5 Conclusions

This chapter has presented a background review into Wireless Sensor Networks (WSN), Medium Access Control (MAC) and Artificial Intelligence (AI) with a bias towards biologically inspired systems. The field of biologically inspired intelligence is very well established as is the field of WSN and MAC protocols. Nevertheless the combination of the two fields is still a new and exciting concept that has a lot of areas of application. The inspiration from Nature and Biology drives our motivation towards applying these biologically inspired methods to the MAC layer of WSNs. To the best of our knowledge, there has been little research on true online swarm intelligence application to MAC. We borrow concepts from Biology and its collective intelligence and apply it to the MAC problem. Inspired by Nature our goal is to create a high performing MAC protocol that shows lower complexity and great adaptability while exploiting DAI.

Chapter 3

Experimental Methodology

Contents

3.1 Wireless Sensor Network Simulator **41**
3.1.1 Classical Single Hop Topology 42
3.1.2 Multi-Hop Linear Chain Topology 42
3.1.3 Multi-Hop Merging Linear Chain Topology 43
3.1.4 Ad-Hoc Network Topology 43
3.1.5 Ad-Hoc Networks with Point Processors 44
3.1.6 Radio Propagation and Link 45
3.1.7 Traffic Model 47
3.2 Empirical Evaluation **48**
3.2.1 Performance Metrics 48
3.2.2 Statistical Validation 50
3.2.3 Model Validation 50
3.3 Comparison Scheme **51**
3.4 Conclusion **52**

In this thesis, we propose a number of biologically inspired Medium Access Control (MAC) protocols. In order to demonstrate their operation and empirically evaluate their performance a simulation model is required. This chapter presents the methods and models used for such purpose. This chapter mainly acts as a supplementary chapter to which we refer from various sections throughout the thesis.

3.1 Wireless Sensor Network Simulator

We propose several different protocols for WSN MAC layer in this thesis. In order to empirically evaluate and assess their performance, we use some classical topologies recognised by a wide range of researchers. These classical methods offer a clear and concise description of operation and realistic equivalents of real systems. In addition, it provides rapid understanding of the behaviour of tested protocols and their limitations.

3.1.1 Classical Single Hop Topology

One of the most popular and well known scenarios is a single-hop topology. This topology is depicted in Figure 3.1. Nodes are all within range of each other and all act as sources except for node 0, which is a sink to which all packets are sent.

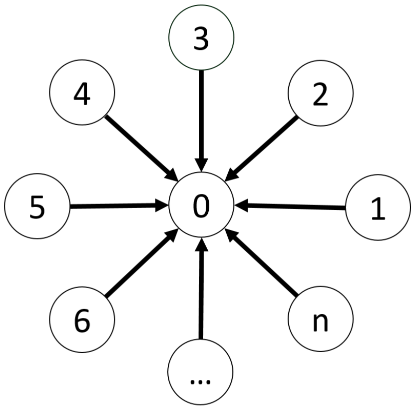


Figure 3.1: Single-Hop Topology

There are many examples of systems that employ such a topology, ranging from widely-used wireless access points and mobile provider networks, to small area sensor networks such as smart-homes [112] where nodes can relay data to single node directly.

3.1.2 Multi-Hop Linear Chain Topology

A multi-hop chain network is an imitation of a multi-hop route through a distributed network. It is significantly more complex than the classical single-hop case. Due to the spacing and data travelling down the route, a significant amount of interference is present. It offers a closer representation of a real randomly distributed. As it is less complicated than an ad-hoc network, it presents significant benefits for eased analysis. It enables a clearer understanding of the protocol operation. Such a scenario is presented in Figure 3.2. The network consists of nine nodes equally spaced in a chain. Data packets are usually sent to the sink on one end of the chain. Radio connectivity and interference in such cases are usually specified based on hops. Nodes usually can communicate with their nearest neighbours (1 hop away), but also interfere with reception over 2 hops. It is very difficult to define a clear boundary of interference range in real applications. Therefore the model employing a 2 hop interference range is commonly used for evaluation [86].

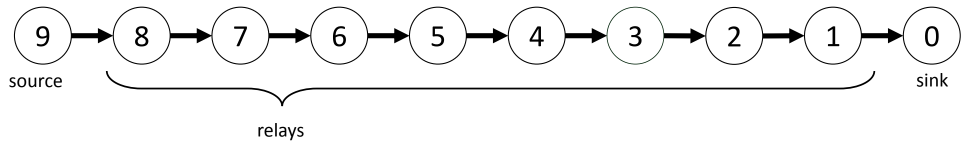


Figure 3.2: Multi-hop chain topology

Examples of such topology applications can be found in a variety of environmental or disaster monitoring situations. Such examples include coast-line monitoring, power-line monitoring, volcano monitoring and smart city applications [30, 32, 38].

3.1.3 Multi-Hop Merging Linear Chain Topology

A merging linear chain is an extension of a simple multi-hop linear chain that imitates two routes joining. The merging of two routes presents a challenge as it is hard to organise transmission timings correctly. Nodes nearest to the merging point are usually unaware of other routes and are not able to communicate to them directly. This poses difficulties when trying to reduce collisions and interference. It provides an easy framework for analysis of a more complex system that approaches the complexity of an ad-hoc network. Such a scenario is represented in Figure 3.3.

Sensor networks used for power line junctions, river-side monitoring, or simply a route from two sources towards the sink in an ad-hoc network are typical examples of such a topology.

3.1.4 Ad-Hoc Network Topology

Many real systems have randomly distributed nodes over the target sensed area. One application in particular that we are interested in is moorland fire monitoring. In such a case nodes would be deployed with an appropriate density of distribution. They would probably be deployed using an aerial platform (either an unmanned or a manned aerial vehicle) which would introduce an irregularly spaced deployment. Nodes would still need to have an appropriate communication distance to send/relay

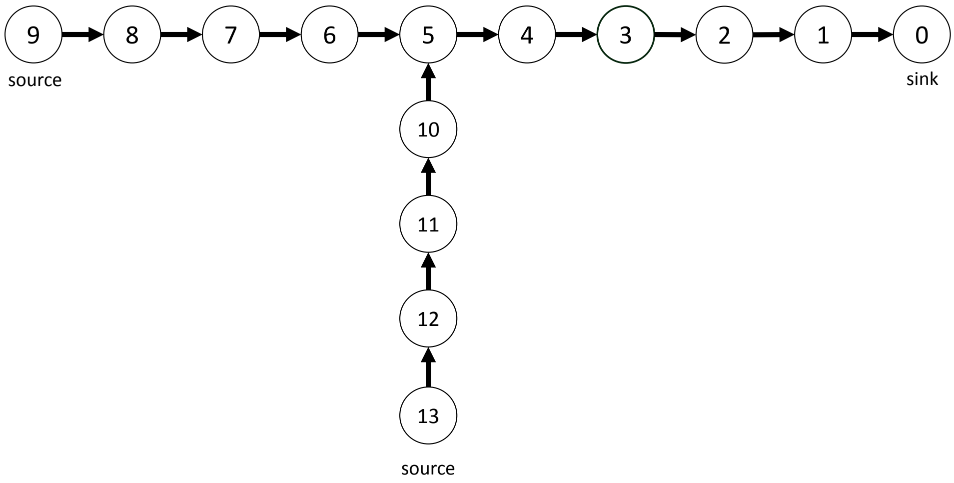


Figure 3.3: Multi-hop chain topology

data back to the main base station (sink). The information carried in such scenario could be for example humidity, temperature and smoke density measurements.

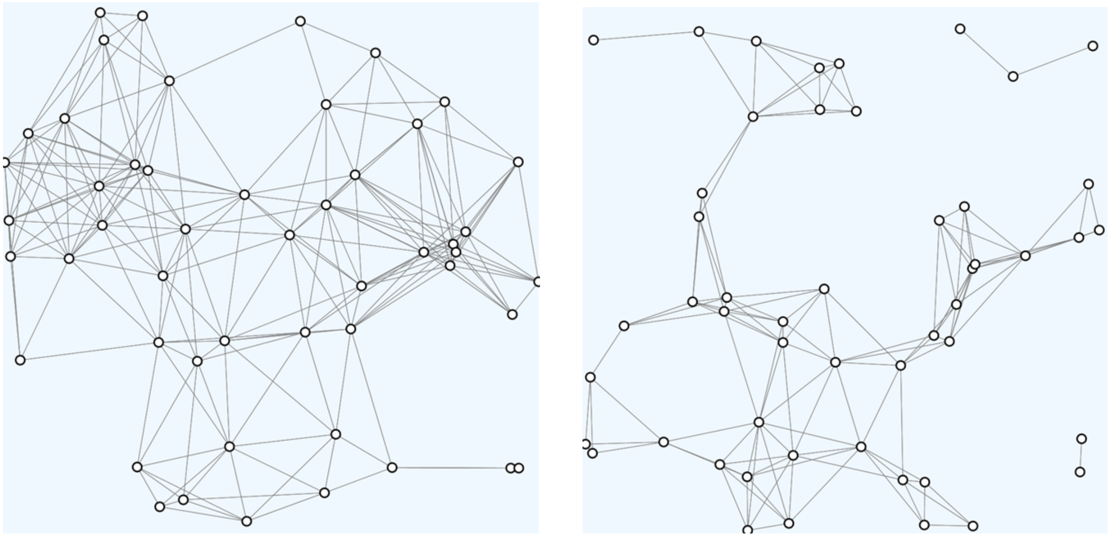
To create such a topology, we firstly take a specified area and deploy nodes according to a uniform probability distribution. A random coordinate within the area is selected for each node. Communications range is then considered to enable the formation of routes.

This type of topology can also represent a variety of other monitoring applications or ad-hoc networks. Networks for agricultural monitoring are usually deployed in a grid, but random ad-hoc deployments are common [42]. Disaster management scenarios can often use this topology as well to provide the coverage and spatial distributions required [113] .

Figure 3.4 shows two examples of a generated network architecture in same physical area. Lines between nodes in this figure indicate the reachable neighbours (i.e. within communication range). The decrease of density of connections with smaller communication ranges can be observed in these examples.

3.1.5 Ad-Hoc Networks with Point Processors

Point processing a network map allows the creation of more controlled realistic deployments. Many monitoring situations deploy devices with a pre-set pattern to cover particular areas. Nodes would not be completely randomly distributed. Instead aerial vehicles would try to deploy nodes in approximate positions.



(a) Network architecture in a square $1km^2$ area where communications distance is 200 meters

(b) Network architecture in a square $1km^2$ area where communications distance is 160 meters

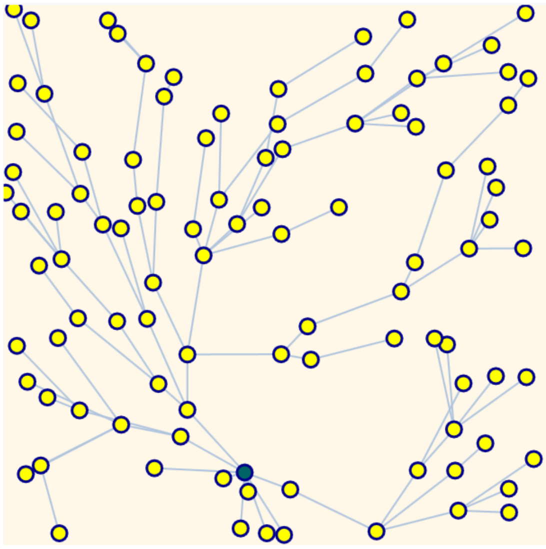
Figure 3.4: Generated Network Architectures

We can use the concept of a Matern Hard-core point process [114]. We firstly generate a network using a Poisson process. We then check the distance of every node to every other node in the network. If they do not match the criteria (if there are other nodes within a specified area), we generate new random positions for these nodes or discard them. On repeating the process multiple times we can obtain a more organised network with a specified minimum distance between nodes that represents controlled deployments more closely. Some networks with different required minimum spacings between nodes are shown in Figure 3.5. Lines between nodes actually represent the routes formed to the sink. Routes were formed using Djikstra's shortest path algorithm [115]. A single sink is present and indicated by a darker color.

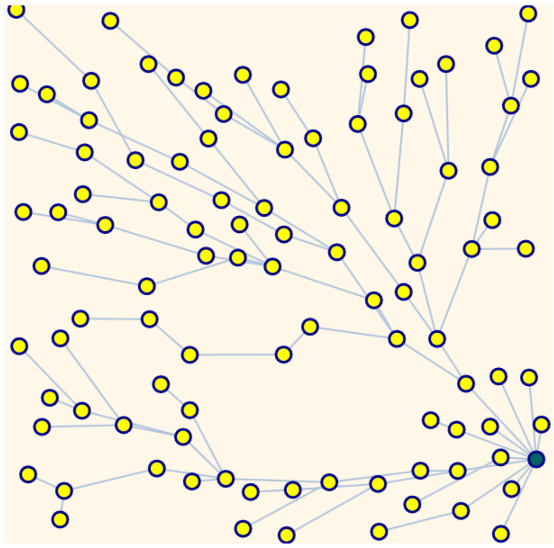
3.1.6 Radio Propagation and Link

A traditional hop-based model is mostly used to represent the communication and interference properties where nodes are able to transmit their data over 1 hop (nearest neighbours), but interference is experienced over 2 hops (as shown in Figure 6.18). Given an equally spaced chain network this implies a maximum throughput of 0.25 Erlangs through it. For simulation parameters, for clarity, we specify the range of 1 hop and interference range in meters where appropriate.

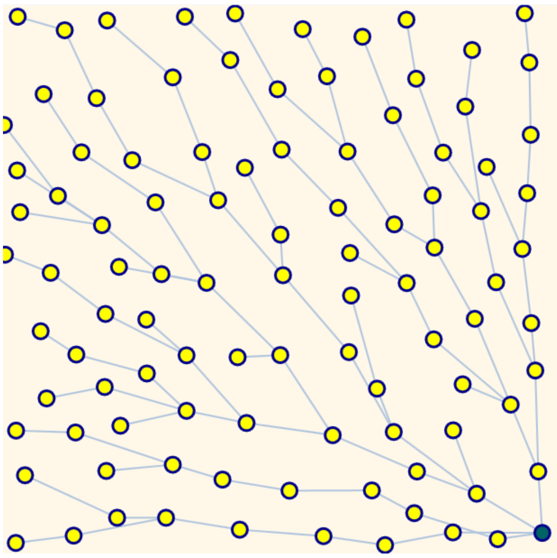
We increase the interference range to observe the adaptability and performance of the protocols under differing conditions. Packets are only received correctly if no in-



(a) Network architecture where nodes are at least 50 meters apart



(b) Network architecture where nodes are at least 100 meters apart



(c) Network architecture where nodes are at least 150 meters apart

Figure 3.5: Generated point processed Network Architectures

terference or other transmissions are present. Propagation delay is calculated based on distance between nodes in meters. Speed of light is assumed for propagation. Given that real simple device hardware can only perform one action (transmit or receive) in simulation, nodes are not permitted to transmit if they are receiving a packet. Acknowledgements are sent right after successful reception of a packet. Minimum acknowledgement timeout is considered which is required for ACK to successfully reach the sender.

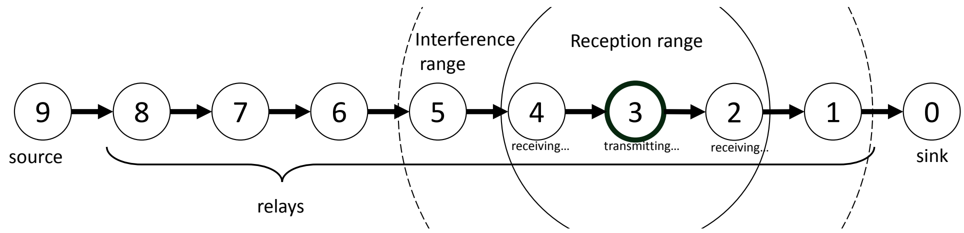


Figure 3.6: Communication and Interference on multi-hop chain network

3.1.7 Traffic Model

Poisson

Different intensity traffic (new packet arrivals) can be modelled using a Poisson distribution. This is widely used to test the protocol or system behaviour at different traffic levels and is known as a Poisson arrival process. The Poisson distribution describes the arrival process for calls or packets and characterises the inter-arrival times. Such traffic models are already used for evaluation of a variety of MAC protocols for WSN [107][86].

We can calculate the next packet arrival time by using the following:

$$\begin{aligned} Average\ number\ of\ events\ per\ interval = \\ (packet\ size/datarate) \cdot (number\ of\ transmitters/load) \end{aligned} \tag{3.1}$$

$$\lambda = 1/Average\ number\ of\ events\ per\ interval \tag{3.2}$$

$$Time\ until\ next\ packet\ arrival = -log(rand)/\lambda \tag{3.3}$$

where $packet\ size$ and $datarate$ are shown in $bits$ and $bits/second$. $load$ is measured in Erlangs and $rand$ is a uniformly distributed random number in the range from 0 to 1. In single-hop scenarios where every node can detect the rest, a maximum $load$ represents the system capacity and is 1 Erlang.

Saturated Traffic

We use a saturated traffic model extensively as it presents a significant challenge to the system. Such a model simply generates a new packet as soon as the previous-generated packet is successfully passed on (to the sink or the next hop). We consider saturated traffic here at a node level rather than network level as only some nodes may be active.

This seems most viable in situations such as fire monitoring. The more information shared in the event of fire means the better the situational picture becomes. In addition, such traffic can test the system/protocol to the critical limit and provides the worst case load performance. It also shows if the system/protocol becomes unstable under high load conditions.

3.2 Empirical Evaluation

3.2.1 Performance Metrics

To evaluate system performance we use several different metrics. These are summarised in this section.

Throughput

Throughput is measured in Erlangs and is calculated based on the number of packets received successfully at the sink throughout each simulation. It can be expressed as:

$$Throughput = \frac{number\ of\ packets\ received\ at\ the\ sink\ *\ packet\ size/bitrate}{simulation\ time} \tag{3.4}$$

which essentially represents the ratio of time that is spent in successful reception over a given simulation time. Usually a maximum of 1 Erlang can be achieved (unless the system uses multiple channels) in the case of a single-hop topology. For multi-hop chain network, the theoretical maximum value becomes more complex to calculate and depends on the interference model chosen. Usually overheads are not included and environments are not perfectly represented in the calculation. Therefore only a value close to the theoretical maximum throughput is achievable.

Offered Traffic

In some of our simulations we also consider offered traffic. This is a measure of the amount of transmitted traffic. It is measured in Erlangs and can be expressed as:

$$Offered\,Traffic = \frac{number\ of\ transmitted\,packets\ *\ packet\ size/bitrate}{simulation\ time} \tag{3.5}$$

End-to-End delay

To measure the latency of the packets we also measure end-to-end packet delays. This provides us with information about how long packet takes to get through the network. In addition, depending on scenario, latency can be critical. Having a system that has lower throughput but better latency could offer significant advantages for some scenarios (moorland fire monitoring). Information arriving late might not be of any use.

End-to-end delay is measured in seconds from the instant of packet generation until the end of successful reception at the sink.

Fairness

In a WSN, fairness is an important aspect of the system. Some nodes in the system can be overwhelmed by interference from other nodes. This can cause them to lose packets (and therefore never present their data to the sink), while other nodes pass packets continuously. In critical environmental monitoring situations, information from every node counts. Therefore fairness of data throughput from each node has to be maintained.

We establish the throughput fairness for different sources using Jain's Fairness Index [116] which is expressed as:

$$J = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \cdot \sum_{i=1}^{n} x_i^2} \tag{3.6}$$

where, in this case, n is the number of source nodes and x_i is the throughput from the i^{th} source node. The results range from $1/n$ (worst case) to 1 (best case).

Settling Time

For a variety of learning and Artificial Intelligence (AI) algorithms, settling time (or convergence time) is an important metric to consider. Settling time shows when the algorithm has finished adapting and stopped searching/exploring for new solutions. However, some of the algorithms by nature never settle and always have some continuous movement, adaptation. In many cases it is considered as initialisation time of the network. In this thesis, the results do not separate initialisation and settled operation. Results are displayed as the average over the whole simulation time. This also includes the non-settled operation time. During this time, performance is usually lower than when settled.

3.2.2 Statistical Validation

To ensure the results presented in this thesis are statistically valid we use different approaches to displaying them. Firstly, we extensively use Cumulative Distribution Functions (CDF) to display the majority of obtained results. This provides a useful and informative way to see all the statistical features in the data - median values, percentiles and deviations. In addition, we include tables that summarise mean values and deviations numerically. This is useful when comparing different schemes.

The results are displayed following a large number of simulations (1000 runs) and over sufficiently long runtime (up to 10000 seconds) to give accurate plots and statistically significant results. A long runtime also ensures that any bottlenecks or unexpected drops in performance are detected.

In the case of other types of plots, we use multiple simulations (at least 100) to average each data point. Where necessary, we also use box plots to represent the data more clearly.

Box plots

Later we use box plots extensively. These are extremely useful when a large amount of statistical data needs to be presented. A box plot visually summarises the main statistical features of data in a simple and easy to understand way. An example of a box plot is shown in Figure 3.7. Sometimes box plots show min and max values using whiskers. However in long tail distributions this presents inaccuracies, as outlier values are not included in min and max definitions. Here we redefine whiskers to represent the 5th and 95th percentiles for clarity.

3.2.3 Model Validation

As we built our simulations from a theoretical baseline, we needed to validate our models. We use some basic schemes such as Pure-ALOHA and Slotted-ALOHA to demonstrate correct operation of the physical layer in simulation. These schemes are very well established and the results are well known. Simple validation test results

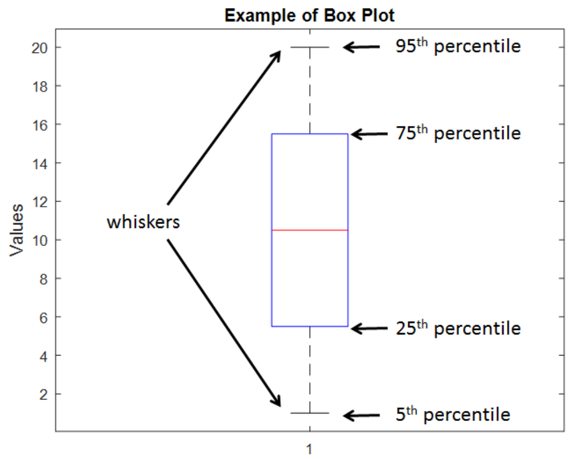


Figure 3.7: Box plot example

for these classic schemes are shown in Figure 3.8.

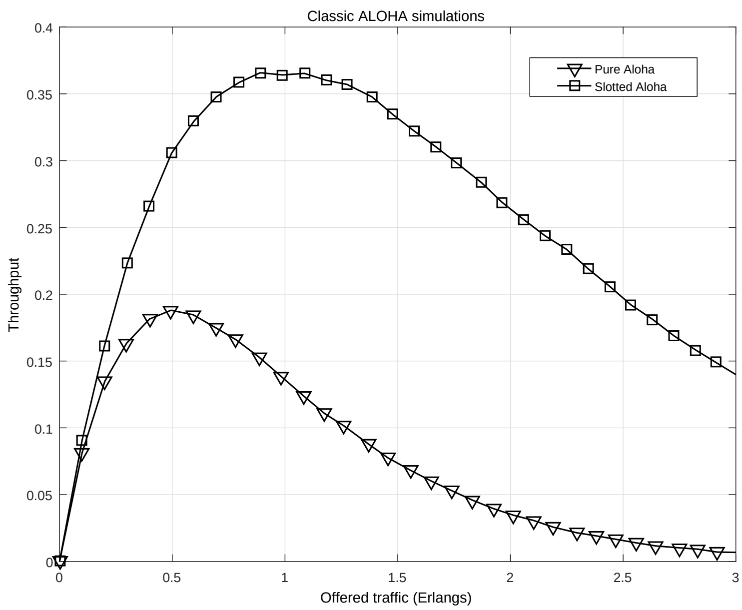


Figure 3.8: Classic ALOHA simulations

3.3 Comparison Scheme

We chose as our comparison scheme the IEEE 802.11-1999 standard [117] - CSMA/CA with RTS/CTS and Binary Exponential Backoff (BEB). It is widely used and provides simplicity and good performance without requirements for synchronisation. It effects collision avoidance through carrier sensing and uses RTS/CTS messages to inform surrounding nodes of transmissions by way of dealing with the hidden node problem on multi-hops. BEB aims to avoid further collisions or interference. Compared to many other, much newer, WSN protocols, CSMA is very low in complexity but offers good performance without synchronisation even for substantial networks. State-of-art MAC protocols that address particular aspects of WSNs, in fact, use raw CSMA or 802.11 standard as a fall-back mechanism to maintain good performance when synchronisation is not available [61]. Due to its popularity and clearly defined implementation, many other researchers also use this scheme for comparison too [118, 56, 55]. While IEEE 802.11 is not an energy efficient protocol, it still provides comparable or even better performance under varying conditions when compared to state-of-art protocols [119]. There are many alternative protocols for WSN. Some of the more well known and established ones are S-MAC, Z-MAC and LEACH. There is a considerable range of approaches, many of which are complex. We now make some general comments on their suitability. Contention schemes are appropriate for distributed networks but suffer from energy waste through collisions. Distributed scheduling is potentially energy efficient but requires a lot of signalling and therefore scalability suffers. With the increased complexity of the state-of-art schemes, their appropriateness for comparison becomes questionable, whereas the classical IEEE 802.11 scheme is well known and established which aids in the understanding of performance.

This is why we chose it for comparison. Our aim is to propose very low complexity protocols with a good performance. For clarity and comparison, we also show and discuss maximum theoretical bounds when evaluating the performance.

3.4 Conclusion

This chapter has presented a summary of simulation methods, evaluation and validation techniques employed to generate the results in this thesis. The key metrics used are throughput, end-to-end delay and fairness. For baseline comparison we use IEEE 802.11 standard which offers simple yet well performing solutions. One of the key elements of our proposed protocols is simplicity of their operation. Therefore the IEEE 802.11 standard fits well for comparison. In addition the results are well known and gives good comparative insight of the performance evaluation.

Chapter 4

Control Engineering in Medium Access Control and Particle Swarm Optimisation

Contents

4.1 Medium Access Control and Control Engineering 54
4.1.1 Controlled Transmission Delay 55
4.1.2 ALOHA with Transmission Delay 56
4.1.3 Experimentation 57
4.1.4 Conclusions 57
4.2 Particle Swarm Optimisation 58
4.2.1 Flocking in Natural Systems 58
4.2.2 Social Concept 59
4.2.3 Basic Particle Swarm Optimisation Operation 59
4.2.4 Particle Swarm Optimisation Algorithm 60
4.2.5 Multi-dimensional Particle Swarm Optimisation 61
4.3 Optimising Transmission Delay for Multi-Hop Chain Network . . 61
4.3.1 Offline Particle Swarm Optimisation 62
4.3.2 Experimental Optimisation Results 63
4.4 Conclusions 67

In this chapter, we discuss our work and findings in searching for a simpler way to define Medium Access Control (MAC). In particular significant similarities between control engineering fundamental theory and the MAC layer behaviour exist. Control engineering in its essence is about designing systems to exhibit desired behaviour. At its core, the approach simplifies the problem via feedback where specific behaviours are controlled. We wished to explore a similar approach to MAC and further improve it by applying some more sophisticated control by means of Distributed Artificial Intelligence (DAI). We define the transmission delay concept. After applying this concept to a simple ALOHA problem we apply an off-line Particle Swarm Optimisation (PSO) to prove the possibility of optimising multi-hop networks via use of only the defined transmission delay parameter. The discussed concept of transmission delay is then carried on and further explored in Chapter 5 and 6 (on Swarming MAC and E-MAC).

4.1 Medium Access Control and Control Engineering

Control engineering conventionally defines a system in a general form to comprise a controller, a plant and some form of feedback from the sensed outputs of the plant back to the input of the controller. To explain these terms, let us take a simple example of controlling boiler pressure. In such a system, we have a valve that measures the boiler pressure and tries to maintain required pressure inside by either opening more or closing down. The boiler, forms a plant, which we try to control. Measured pressure is a form of feedback back to a controller (the valve). This valve compares required pressure with measured pressure and produces the error signal (the difference between measured pressure and maximum allowed pressure). Based on the error signal it either closes down or opens up more (acts as a policy).

Hence, the input signal (required pressure) is coupled with the feedback (measured pressure) to form an error signal to which a policy is applied, amplifying or reducing it (valve opening). The output (or a fraction of it) is fed back to the controller, which then acts on the new input signal and adjusts the signal controlling the plant with the objective of improving the output (maintaining required pressure).

Similarly we can represent MAC protocol behaviour. Generated traffic can be considered as an input to the system. The controller accepts feedback and generated traffic and regulates offered traffic. This offered traffic passes through the plant (environment) and the output is a function of throughput. Feedback in this case is in the form of acknowledgements. The controller can act to increase or decrease offered traffic based on these acknowledgements. A comparison is shown in Figure 4.1.

Much research that models communications using control engineering theory already exist [120, 121, 122]. However, in this thesis we use the similarity between control engineering theory and communications as an inspiration to define a different way of controlling transmissions.

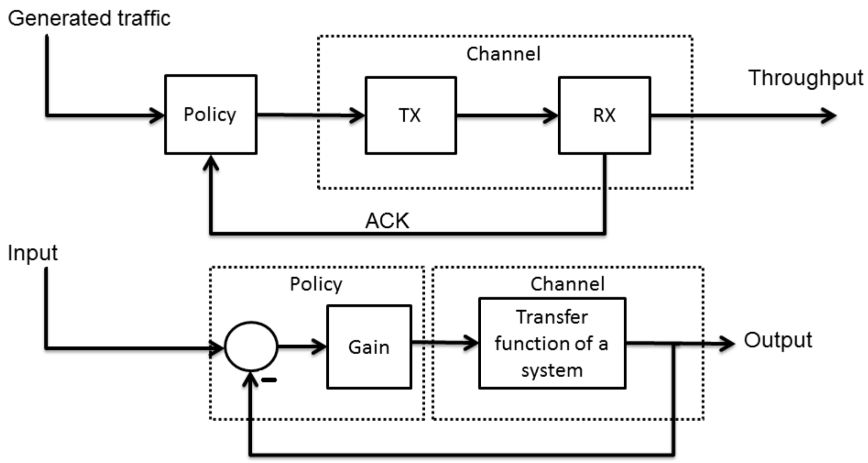


Figure 4.1: MAC and control engineering mechanism comparison

4.1.1 Controlled Transmission Delay

To mimic a simple control engineering system we can define a simple transmission delay ($txDelay$). The $txDelay$ value represents a time period after each transmission during which no new packet transmission would be allowed. A larger $txDelay$ corresponds to lower offered traffic (less frequent transmissions). Just as in a control engineering problem, changing $txDelay$ would control the output (almost like boiler pressure). Feedback in this case corresponds to transmission outcome (acknowledgement or time-out).

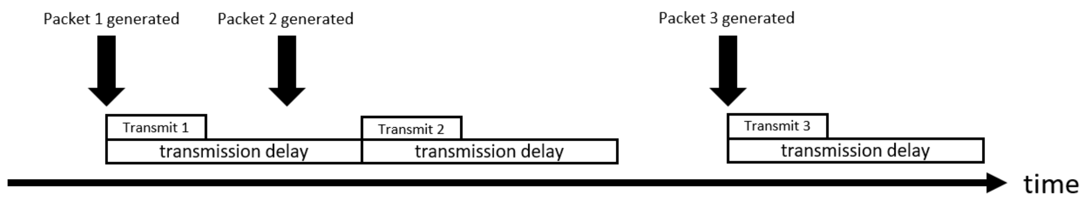


Figure 4.2: MAC and control engineering mechanism comparison

Transmission delay is different to back-off. It is constantly applied directly after transmission no matter what the outcome. Back-off usually is only applied after a transmission failure and is reset after it succeeds. The $txDelay$ value can be incremented or decremented based on each outcome and a new, updated value can be applied for the next coming transmission. Example of this transmission delay is shown in Figure 4.2.

We normalise the value of $txDelay$ to packet durations for clarity. Packet duration can be expressed as ($packetsize/bitrate$). Hence if packet duration is 100ms, transmission delay of x would imply a waiting before next transmission is allowed of $x \times$100ms. The $txDelay$ is defined from the start of transmission. A value of 1 would imply no transmission delay as it would expire at the same time as the transmission is finished. This also sets the minimum value of $txDelay$ as 1 packet duration.

4.1.2 ALOHA with Transmission Delay

We can take this simple concept and apply to a Pure ALOHA scenario. We treat acknowledgements as the form of feedback and, based on outcome, increment or decrement the $txDelay$ value at each node. We need to consider the scale of incrementation and, in this particular case, the dispersion during the incrementation to avoid repetitive collisions. To make sure this is satisfied the incrementation can be randomised within a small range.

Algorithm 1: Pseudo-code of txDelay variation (direct control engineering feedback for txDelay)

```
while network running do
    // Update txDelay
    if ACK is received then
        txDelay ← txDelay − rand(0, 5)
    end
    if ACK expired then
        txDelay ← txDelay + rand(0, 5)
    end
end
```

Algorithm 1 shows the general pseudo code of the protocol and how $txDelay$ will be varied. If acknowledgement fails - this means there is contention on the channel, hence incrementing $txDelay$ (reducing output). Successful transmission implies the channel is either fully- or under-utilised so the output could be increased (decrement $txDelay$).

This simple implementation is not expected to offer high throughput, but should be able to limit the output and stabilise the throughput even at high generated traffic levels. Pure ALOHA in this case tends to suffer in performance significantly.

4.1.3 Experimentation

To quickly evaluate the possibility of using transmission delay, we set-up a classical scenario. We define a one-hop scenario where 50 nodes attempt to transmit a packet to a single sink (discussed in Section 3.1.1). The traffic is modelled as a Poisson process in this case (see in Section 3.1.7).

We define a simple protocol for comparison and a protocol that we want to test:

Classical Pure Aloha - If a node has a packet to transmit, transmit it immediately. If a transmission is in progress, transmit directly after this is finished. If a packet does not reach the sink, do not retransmit.

Aloha with transmission delay - If a node has a packet to transmit and transmission delay is not engaged, transmit immediately. At the start of transmission, engage transmission delay with the $txDelay$ value. $txDelay$ initially starts as 1 (no delay). Change $txDelay$ value based on Algorithm 1.

Figure 4.3 shows throughput versus generated traffic of each protocol. It is clear that both perform well at low traffic load and not many collisions occur. ALOHA with transmission delay is simply working like Pure ALOHA as $txDelay$ value stays at 1 due to no collisions. However when the traffic load becomes too high, Pure ALOHA tends to suffer in performance with throughput tending to zero. Our approach with transmission delay, as expected, stabilises. Therefore it avoids decline in throughput. Unlike usual back off schemes the $txDelay$ value also retains memory of the channel which can represent the condition of the environment.

4.1.4 Conclusions

This method, inspired directly by Control Engineering, does not show any significant improvement on Pure ALOHA and does not outperform any state of the art schemes. However it is the basis of a concept that can be utilised for applying simple DAI techniques. While this particular approach would not achieve good performance in more complex multi-hop scenarios (as is the case for Pure ALOHA), a more sophisticated controller of transmission delay that would imply distributed intelligence could be feasible. Transmission delay by itself forms a very simple parameter to adjust with the objective of controlling the MAC layer, and it can be applied to a variety of algo-

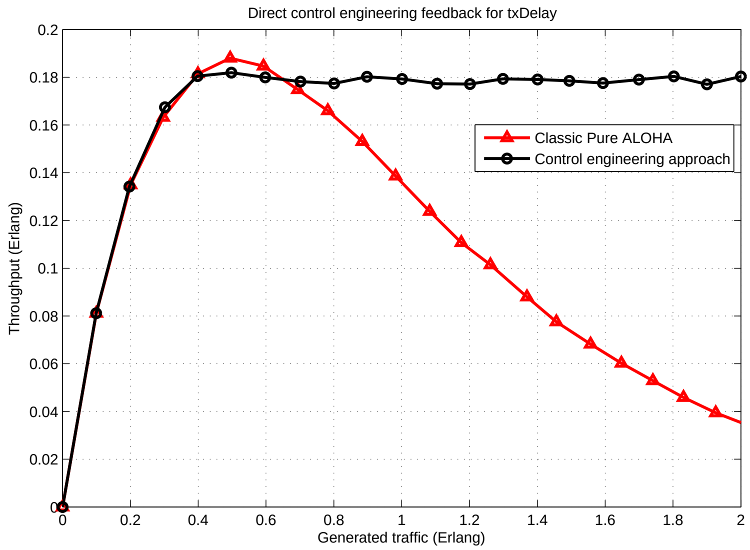


Figure 4.3: Throughput when direct control engineering based feedback for $txDelay$ is applied

rithms. In addition it does not incur additional processing/storage requirements and, by itself, can provide a measurable indication of node performance.

4.2 Particle Swarm Optimisation

Algorithms inspired by nature attract our attention due to their emergent problem solving abilities in a distributed manner. The PSO is one such algorithm. It has proven its optimisation capabilities in a variety of engineering problems. We could apply such an algorithm to decide on the transmission delay value necessary to optimise the network. This section will introduce the PSO and the main concepts behind it.

4.2.1 Flocking in Natural Systems

Many emergence examples can be observed in nature, particularly in swarms. Reynolds [123] looked at bird flocking behaviour and treated the whole flock as an interaction between simple individuals. Each individual was only affected by the behaviour of near neighbour individuals. Based on this he modelled the behaviour of each individual using three simple rules:

- Separation - avoid colliding with neighbouring individuals.
- Alignment - try to match heading and velocity of neighbours.

- Cohesion - try and stay at the centroid of the nearest neighbours.

These three simple rules successfully recreated bird flocking and show the power of emergence, and the ability of multi-agent systems to cooperatively perform complicated tasks [124]. The simulated flocking agents were originally called boids [123]. This led to the inspiration for an optimisation algorithm referred to as the PSO [125].

4.2.2 Social Concept

Kennedy [126] identifies three main principles that formed patterns in social behaviours. These are:

- Evaluation - defines how each entity evaluates itself and finds features.
- Comparison - allows entities to compare themselves to others and establish better evaluated features.
- Imitation - the entity tries to imitate what it has learned during comparison.

PSO takes the concept of boids [123] and combines it with these social principles observed naturally. This combination creates a powerful optimisation tool that enables intelligent flocking through a solution space. This behaviour shows rapid convergence to find optimal solutions.

4.2.3 Basic Particle Swarm Optimisation Operation

The entities in PSO are called particles. Each particle represents a position in the solution space. A position can represent a parameter value of a particular algorithm. Performance of the algorithm can be evaluated using this parameter. This forms the fitness value that is a function of the position of a particle.

To enable particle movement, each particle also has a velocity associated with it. Inspired by social and flocking concepts, the velocity of each particle is influenced by the best positions found (based on fitness values). Both personal and neighbour/global best fitness positions are used. Given a number of such particles, flocking towards and around good solutions emerges from these interactions.

Flocking enables exploring around a potential solution. This either refines the solution obtained at each step or aids in preventing convergence to local optimal solutions.

4.2.4 Particle Swarm Optimisation Algorithm

Each particle has information associated with it - fitness, position, velocity and personal best position. In a general PSO there is also the global best position which is established from within the whole particle population. If we assume a single dimension solution space ($f(x)$) and a particle swarm that consists of i entities, we can express the information about each as:

x_i^t – position of the i^{th} particle at iteration t

v_i^t – velocity of the i^{th} particle at iteration t

l_i^t – the best found i^{th} particle position at iteration t (local best)

In a classical PSO [125] we also define current g which is the global best position. Three main steps are iterated, in sequence, in this algorithm:

Compare - the position of each particle is compared and the position with the best fitness is recorded as g

Update (imitate) - the velocity and position of each particle are updated based on l and g according to:

$$v_i^t = w \cdot v_i^{t-1} + c_1 \cdot \phi_1 \cdot (l_i^{t-1} - x_i^{t-1}) + c_2 \cdot \phi_2 \cdot (g - x_i^{t-1}) \tag{4.1}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{4.2}$$

where ϕ_1 and ϕ_2 are random weights in the range $[0, 1]$, c_1, c_2 are constants (in the original algorithm simply set to 2) and w defines the inertia weight of particles.

Evaluate - the fitness of each particle position (solution) is evaluated (using a specified fitness function or empirical methods) and l_i is updated:

$$l_i^t = \begin{cases} l_i^{t-1} & \text{if } f(l_i^{t-1}) > f(x_i^t) \\ x_i^t & \text{if } f(l_i^{t-1}) < f(x_i^t) \end{cases} \tag{4.3}$$

where $f(\cdot)$ evaluates fitness

Iterating over these three simple steps allows particles to migrate towards the best global solutions. At the same time particles are being attracted to the best personal solutions as well. These are also called cognitive ($l_i^t - x_i^{t-1}$) and social ($g - x_i^{t-1}$) components of particles. However, the nature of this classical algorithm is unstable as velocities can increase with no bound. Hence, Kennedy and Eberhart originally proposed the limiting parameter v_{max} to restrict particle movement [81].

These simple rules form the basis for PSO that achieves a complex movement of particles through the solution space. The task of searching for a solution is not explicitly defined in the algorithm but instead it emerges from the interactions between particles. Particles then flock towards the best solution. This ability to concentrate in the area around the known best solutions opens quick discovery of even better solutions. Nevertheless, this can also cause particles to stay in local optima and not explore sufficiently. This can be avoided by considering different particle deployments and adjusting the acceleration of particles (using different w).

4.2.5 Multi-dimensional Particle Swarm Optimisation

While the simple concepts of PSO are more easily explained as a one dimensional problem, the real power of the PSO lies in multi-dimensional (or multi-objective) optimisation. All the concepts remain the same. In addition, position and velocity of each dimension is updated separately using the same formulas as in Equations 4.3-4.2. Both l and g in this case store values of all dimensions and present the most fit combination of the positions.

4.3 Optimising Transmission Delay for Multi-Hop Chain Network

As a proof of concept, we apply PSO to a multi-hop chain network to optimise the static transmission delay of each node to improve performance. This allows us to judge whether it is possible to use only transmission delay as a means of MAC on more complex problems.

If we take a large communications network, we can split it into smaller segments, each forming a simple multi-hop route back to the sink. We can apply a transmission delay to every node of this route and control it such that better performance is achieved.

We also discuss this type of scenario in Section 3.1.1. Here we use a scenario with 10 nodes, one of which is a sink (node 0).

4.3.1 Offline Particle Swarm Optimisation

In order to test the feasibility of transmission delay usage on this simple chain network we will consider the static case. We assume that each node starts with a pre-set transmission delay value. This allows us to run the system and obtain the achieved throughput as well as fairness. We will use a PSO algorithm to optimise the pre-set transmission delay values for each node in the chain. Fitness will be defined as an equal combination of both throughput and fairness.

Implementation

We implement the optimisation in this case as a 10 dimensional problem. Each dimension represents the transmission delay value for each node. Therefore during the optimisation, each particle will move in a 10 dimensional solution space. We then have a number of particles moving around this space and storing their own best found position (l) for the cognitive component. The global best position (g) would be evaluated at every iteration to provide a social component for the swarm. Essentially we would have a specified number of particles swarming in a 10 dimensional space.

Throughput can take a value between 0 and 0.25 Erlangs. Fairness is evaluated using Jains Fairness Index (explained in Section 3.2.1) which takes the maximum value of 1. To equalise the weight of throughput and fairness contributing to fitness we can define fitness as:

$$fitness = 4 \times throughput + fairness \tag{4.4}$$

Throughput and fairness therefore contribute equal amounts towards the evaluation of fitness for this optimisation.

Algorithm 2 shows the pseudo code for this optimisation. The general structure of this optimisation follows the 3 main steps for a PSO:

Evaluate - Run a simulation based on each particle position (10 dimensions - 10 transmission delay values for 10 nodes). Obtain results and evaluate fitness based on throughput and fairness (using function defined on lines 4-6). Update l of each particle. Lines 8-10 in Algorithm 2.

Compare - the l fitness of each particle is compared and the position with best fitness is recorded as g. Lines 11-12 in Algorithm 2.

Update (Imitate) - The velocity and position in every dimension of every particle is updated based on l and g values. Lines 13-19 in Algorithm 2.

Algorithm 2: Pseudo-code of Particle Swarm Optimisation applied to find static transmission delays

1 Create n particles with random position and velocities for d dimensions:
2 $x^t_{n,d}$ and $v^t_{n,d}$ for $d \in 0..9$ and $t = 0$
3 Set c_1, c_2, w Set $t = 1$;
4 **Function** *f(x)*
  // **Fitness function**
5   Simulate: $Node_d$ transmission delay is set to $x^{t-1}_{n,d}$ where $d \in 0..9$
6   Return value $4 \times throughput + fairness$
7 **for** *t iterations* **do**
  // **Evaluation step**
8   **for** *n particles* **do**
9   If $f(x^{t-1}_{n,d})$ is higher than $f(l_{n,d})$ set $l_{n,d}$ to $x^{t-1}_{n,d}$ for $d \in 0..9$
10   **end**
  // **Comparison step**
11   Find the particle n with highest $f(l_{n,d})$ value
12   Set the g_d to $l_{n,d}$ for $d \in 0..9$
  // **Imitation step**
13   **for** *n particles* **do**
14   **for** *d dimensions* **do**
15   Generate two random numbers ϕ_1, ϕ_2 between 0 and 1 (uniform distribution)
16   Set $v^t_{n,d} = v^{t-1}_{n,d} + c_1 \cdot \phi_1 \cdot (l_{n,d} - x^{t-1}_{n,d}) + c_2 \cdot \phi_2 \cdot (g^{t-1}_d - x^{t-1}_{n,d})$
17   Set $x^t_{n,d} = x^{t-1}_{n,d} + v^t_{n,d}$
18   **end**
19   **end**
20 **end**

4.3.2 Experimental Optimisation Results

We set up an experiment that runs Algorithm 2 for the network shown in Figure 3.1. In addition we simulated cases where nodes 8, 7 and 6 are also sources, forming a 1 to 4 source cluster at the end of chain. This then enables evaluation of fairness as well. Saturated traffic is assumed in order to find the performance limits (see Section 3.1.7).

Parameters

Based on the classical PSO algorithm, we set the operating parameters to those values shown in Table 4.1. w defines the acceleration of particles. $w < 1$ slows down particles and makes the search finer (decelerates). $w > 1$ makes particles more energetic (accelerating quickly), therefore exploring a wider solution space. We chose a value of 0.7 as it generally provides faster convergence towards better solutions [127].

Communications related parameters are shown in Figure 4.2. The bit rate here is chosen based on a popular ZigBee platform for WSN [128].

Table 4.1: PSO Parameters

Parameters	**Values**
w	0.7
$c1$	2
$c2$	2
n	30

Table 4.2: Simulation Parameters

Parameters	**Values**
Channel bit rate	250 Kbits/s
Data packet length	1000 bits
ACK packet length	20 bits
RTS/CTS packet length	20 bits
Transmit range	200 m (1 hop)
Interference range	400 m (2 hops)

Results

Results for fitness values, throughput and fairness versus number of iterations are shown in Figures 4.4, 4.5 and 4.6 respectively. The results clearly indicate that even such a simple parameter as transmission delay can provide solutions which are very close to the theoretical best performance values. As anticipated, the cases with more sources need more iterations to optimise the values due to the more complex environment. However all cases offer higher than 0.23 Erlangs throughput within only 500 iterations. All cases find near perfect fairness as well.

Figure 4.7 also shows a representation of possible solutions found by the PSO. It

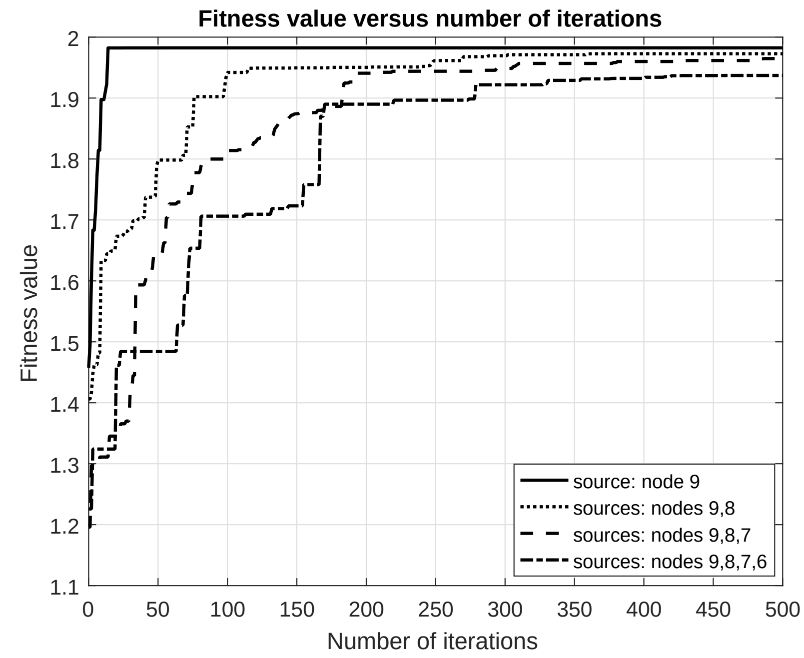


Figure 4.4: Fitness values versus number of iterations - Optimising transmission delays for chain network (Section 3.1.1) using PSO.

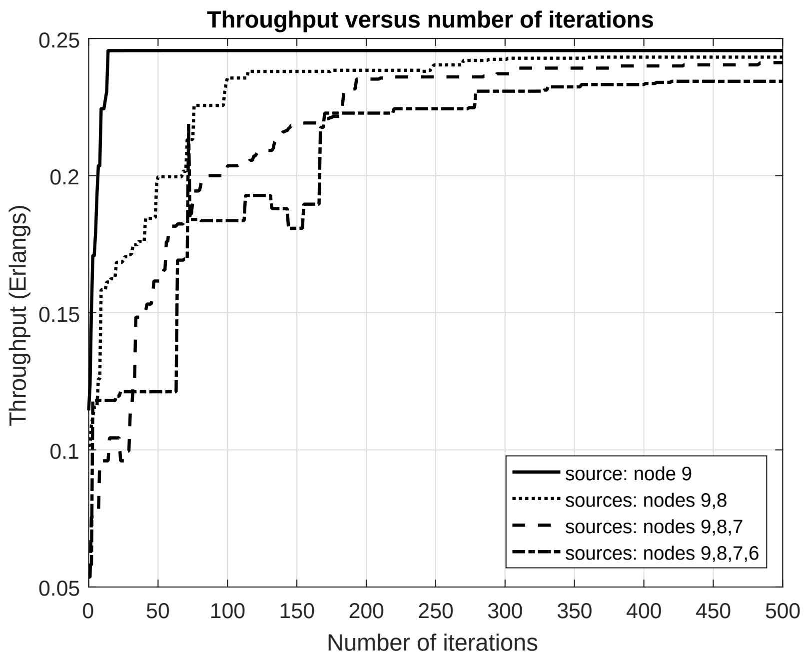


Figure 4.5: Throughput versus number of iterations - Optimising transmission delays for chain network (Section 3.1.1) using PSO.

presents transmission delay values chosen for each node for cases with different numbers of sources. In all the cases (except the 4 source case) we can see a clear pattern of transmission delay chosen for the source nodes. It makes the sources further along the chain maintain lower transmission delay to provide fairness. All the other nodes are

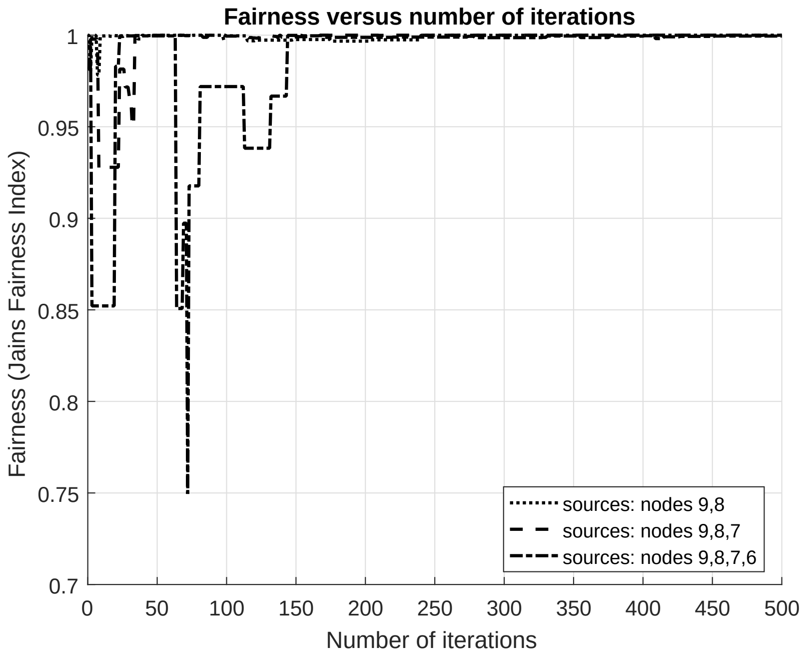


Figure 4.6: Fairness versus number of iterations - Optimising transmission delays for chain network (Section 3.1.1) using PSO.

randomly chosen below the value of 4. These nodes essentially do not affect performance as long as they do not delay the packet transmissions too long. Interestingly, the 4 source case provides a solution where node 8 maintains $txDelay$ of 8.128 and node 9 maintains 5.439. While this might look unusual, separate individual tests have shown high throughput and fairness under this configuration. We did not specify any other fitness influencing parameters in the optimisation other than throughput and fairness, so the system does not optimise for the number of collisions or packet losses. Sometimes a packet goes through but an acknowledgement is lost. This does not reduce the throughput at the sink (node 0) as the packet is still received but the node that sent the packet will not be aware of this.

Figure 4.8 shows the average throughput over time for given solutions in Figure 4.7. All nodes take some time to settle into consistent operation and then perform very well, quickly reaching good average throughput. Interestingly, the case with 2 sources takes the longest. At the beginning the system exhibits more repeated collisions due to $txDelay$ at node 8 and 9 being very close to multiples of each other. Please note that nodes 1-7 may appear to have random $txDelay$ values. As long as these nodes have a value lower than 4.1 (less than 0.25 Erlang throughput), it means a packet can pass through uninterrupted (no collisions). The source node $txDelay$ value regulates how often packets are passed through the network.

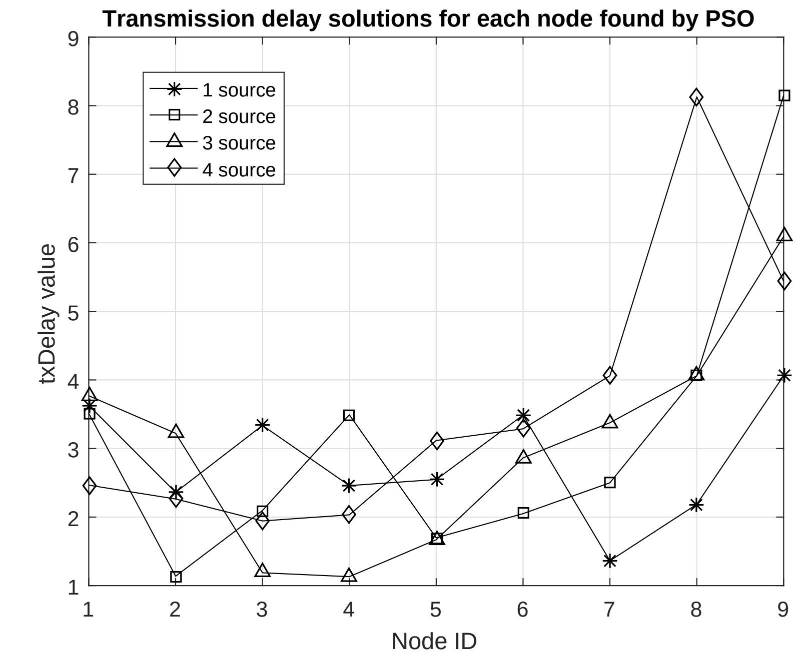


Figure 4.7: Transmission delay solutions for each node found by PSO.

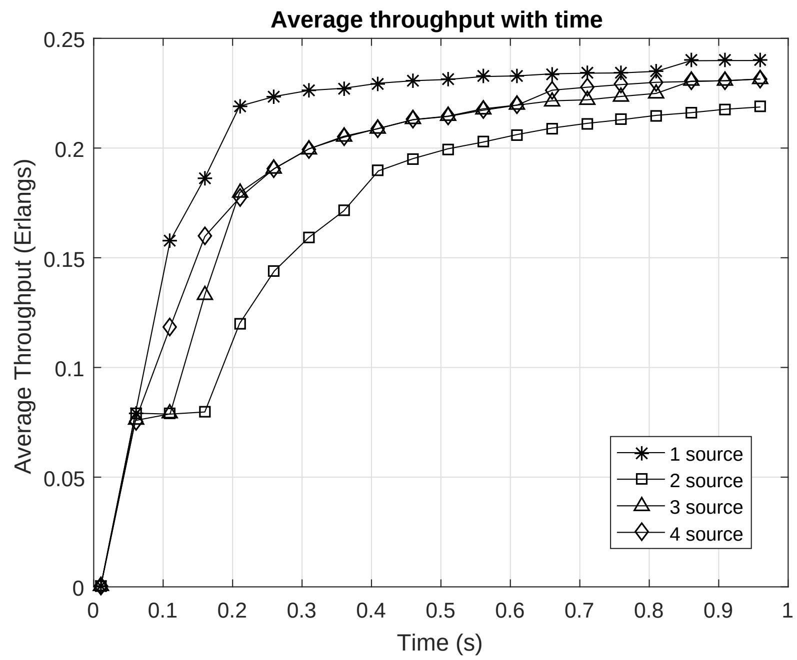


Figure 4.8: Average throughput for nodes with $txDelay$ solutions presented in Figure 4.7.

4.4 Conclusions

The implementation of a static transmission delay policy for MAC control on multi-hop chain networks offers an extremely simple and well behaved solution. This can be easily applied to situations where the environment is precisely known before deployment. This knowledge allows pre-optimised transmission delays and therefore reduces the processing requirements. However, this particular implementation would not work efficiently under a changing environment or where precise conditions are not known before deployment.

Nevertheless we use these results as a proof of concept for using transmission delay to control medium access. In addition the application of an offline PSO allows for pre-optimisation of the system for known conditions. The topology of a PSO closely relates to that of a WSN. In Chapter 5, inspired by the PSO algorithm, we explore the possibility of using its concepts in an online manner to enable cooperative intelligent control of a MAC layer that uses transmission delay.

Chapter 5

Swarming Medium Access Control Protocol

Contents

5.1 Motivation . . . 70
5.1.1 Control Engineering in Medium Access Control . . . 71
5.2 Swarm Intelligence-Based Medium Access Control Protocol . . . 71
5.3 Scenario and Assumptions . . . 73
5.3.1 The Basic Scenario . . . 73
5.3.2 Traffic . . . 73
5.3.3 Propagation and Radio . . . 74
5.3.4 Scheme for Comparison . . . 74
5.3.5 Simulation Parameters . . . 74
5.4 Performance Evaluation . . . 75
5.4.1 Metrics for Analysis . . . 75
5.4.2 Results . . . 75
5.5 Conclusions . . . 80

This chapter introduces a novel Medium Access Control (MAC) scheme for ad-hoc Wireless Sensor Network (WSN). It is based upon the emergent properties of complex systems, exploiting the concept of swarm intelligence. The benefits brought about by this approach are node simplicity given very low implementation overheads, which also contributes to network scalability. Through the co-operative behaviour of nodes, a network can exhibit emergent self-initialisation and organisation and is then able to adapt to environmental and structural changes. The protocol was developed using social concepts drawn from the field of Particle Swarm Optimisation (PSO), alongside negative feedback concepts from control system engineering. It offers comparable or better performance (in throughput and delay) with lower complexity and less overheads for larger scale networks when compared to the widely used and simple but still very effective IEEE 802.11 CSMA/CA standard.

5.1 Motivation

A Wireless Sensor Network (WSN) can provide a number of benefits to a variety of applications ranging from monitoring to industrial control. Technological advances now enable device deployments in large quantities. To keep costs low and operational times long, the need to simplify the hardware and software of the devices becomes a challenging task. Given the key benefits of these devices lie in communications, efficient and reliable MAC becomes increasingly more important. Inspired by biology, we believe the application of algorithms derived from natural phenomena can be beneficial when taking up such a challenge.

Swarm Intelligence (SI) is inspired by the social behaviour of biological swarms such as ant, bee and termite colonies. While each entity in a colony is simple, complex behaviour and patterns emerge when they interact with each other and cooperate [80]. This emergence phenomenon has inspired many optimisation algorithms such as PSO [129], Ant Colony Optimisation (ACO) [130] and the bees algorithm [83]. They employ emergence and distributed intelligence to solve problems with considerable effectiveness. Whilst each entity in a swarm is simple and easy to create, achieving emergence and useful behaviour from social interactions is not an intuitive and easy task.

The purpose of this chapter is to present a new online SI based MAC scheme. We design the protocol to operate online with PSO concepts (offline PSO optimisation is discussed in Chapter 4). It enables self-organisation and adaptivity, where a swarm of nodes is simply and independently able to determine the maximum possible rate to generate/send packets. The scheme demonstrates better performance for larger networks when compared to a more complex, sensing-based Carrier Sense Multiple Access (CSMA) scheme. We compare our scheme to CSMA/CA because it is a well established standard, widely used and understood not only in many WLANs, but also as part of the IEEE 802.15.4 WPAN [131] and IEEE 802.11 standards.

5.1.1 Control Engineering in Medium Access Control

In Chapter 4, we presented a method of controlling transmission at the MAC layer using transmission delay. This method was inspired by approaches used in control engineering where some form of feedback is used by a controller to reduce or increase the input to the system to achieve a desired output.

Transmission delay is a different method of back-off which is applied after every single transmission and either incremented or decremented but never reset. In Chapter 4 we presented a simple scheme where the Transmission Delay length was incremented and decremented given failure or success during transmission respectively. We measure Transmission Delay in packet durations (the time it takes to transmit one packet). We denote it $txDelay$.

In Chapter 4 we presented the application of an Offline PSO to optimise $txDelay$ values in a simple multi-hop chain network that performs well. However the Offline PSO can only be used as a pre-deployment optimisation where precise environment conditions are known. In this chapter, we present a protocol which we have developed using the same PSO principles but for online operation and optimisation. Each node using this protocol aims to optimise its $txDelay$ based on observations of its neighbours and its own failures and successes, in real time.

5.2 Swarm Intelligence-Based Medium Access Control Protocol

We propose that each node in the WSN acts as a particle and 'moves' around the solution space based on transmission delay). Hence, each node searches for the best transmission delay to increase the fitness. Fitness in this case is based on data we know about the environment that a node is operating in. For simplicity, we choose to use an average probability of success taken over a set of packet transmission outcomes. The average is calculated using the Robbins Monro (RM) [132] algorithm.

PSO information is shared between the closest nodes by adding it to every packet and acknowledgement. Before each transmission, each node updates its own fitness and finds the global best fitness based on information it has about itself and its neighbours (directly reachable nodes). Then it performs the update step and recalculates the velocity and hence the new position in the solution space. This position is the transmission delay. We expect this implementation to make nodes change transmission delays as a swarm. In this case nodes will tend to be both competitive and altruistic - they will try to achieve the best probability of success tempered by information about their neighbours. We define the update step as:

$$v = w \cdot v^{t-1} + \phi_1 \cdot (l - x^{t-1}) + \phi_2 \cdot (n - x^{t-1}) + \tau \tag{5.1}$$

$$x = x^{t-1} + v \tag{5.2}$$

where ϕ_1 and ϕ_2 are random weights in the range $[0, 1]$, w is the inertia weight, x is simply a transmission delay, l represents the delay associated with best fitness from own experience and n represents the delay associated with the best performing neighbour. τ plays an important role in this equation. It provides direct feedback about node performance based on experience. We define it as:

$$\tau = \begin{cases} +v_c \cdot (1 - p_s) & \text{, if an ACK failed} \\ -v_c \cdot (1 - p_s) \cdot r_c & \text{, if an ACK succeeded} \end{cases} \tag{5.3}$$

where v_c is the control velocity, p_s is the historical probability of success based on previous values, r_c is the control ratio. τ provides a push for particles in either direction based on recent acknowledgement status. v_c is a constant that defines the speed at which particles react to the feedback. The control ratio allows nodes to slow down once they are successful which increases the resolution of the search. The $(1-p_s)$ term also provides a resolution increase after a node becomes more successful. It reduces the τ speed and improves accuracy, thereby enabling nodes to settle down. It also allows nodes to wake up again and start moving if the environment changes and p_s drops.

All the parameters in the Swarming MAC algorithm (Table 5.1) were set based on nominal values for PSOs [133]. We also ran some parameter sweeps which showed that performance is generally insensitive to the variations in nominal parameter values in the PSO part of the algorithm. We ran parameter sweeps for the different τ parameters. We also found these to be insensitive around the nominals and only changed significantly at parameter extremes.

The benefits of the algorithm appear when a swarm of identical elements is executing it. The outcome is due to the social interaction between the nodes. Nodes share

Table 5.1: PSO parameters

Parameters	Values
w (inertia weight)	0.5
v_c (control velocity)	0.3
r_c (control ratio)	0.02
α (RM averaging)	0.05 (over 20 values)

their transmission delay by adding a tiny amount of information to every data and acknowledgement packet. Shared information and the PSO model for cognitive and social components allow the intelligent movement of particles through the solution space. When nodes change the transmission delay cooperatively under this algorithm, patterns are formed which enable simple collision avoidance and good delay and throughput performance. Most applications of PSO are performed off-line (optimisation before deployment) whereas our proposed algorithm operates on-line (in real-time).

5.3 Scenario and Assumptions

5.3.1 The Basic Scenario

300 nodes are randomly deployed (under a uniform distribution) in a $2km^2$ area. All nodes use the same channel for transmission and reception. There is only one sink node in the network, randomly chosen from the existing nodes. All nodes act as sensors. All the data packets are of the same length. To sort out routes, Djikstra's Shortest Path routing algorithm is used. There is no synchronisation between nodes. The scenario is based on large-scale wireless sensor networks with high node density. Such a scenario is discussed in Section 3.1.4.

5.3.2 Traffic

In many real-time monitoring WSNs, the end-to-end delay of data is very important. Hence, instead of simulating a Poisson arrival model we use a saturated model where source nodes generate packets as soon as the last data packet has been successfully transmitted (as discussed in Section 3.1.7). Hence the transmitted data is always the newest. In the case of a Poisson model this is not always true if the queue grows. In an example case of moorland fire monitoring, the data delay is very important and late data is of little use.

5.3.3 Propagation and Radio

We use a well established model, with communication and interference based on defined hop ranges. Interference range is always twice the communication range. Only the packets within 1 hop (communication range) can be received. Packets are considered to be received successfully if there is no overlap between other inbound reception or interference. Even though distances are very small, we do not ignore propagation delay. The radio and propagation model used here is discussed in Section 3.1.6. Based on realistic devices we also assume that nodes cannot start a new transmission if they are currently in successful reception since the hardware would prevent it.

5.3.4 Scheme for Comparison

We simulated our swarming protocol and as a comparison have implemented and simulated CSMA. We implemented the protocol that is used in the well known IEEE 802.11 standard - CSMA/CA with RTS/CTS and Binary Exponential Backoff (BEB) (discussed in Section 3.3). This also shows the best performance when compared to other variations without RTS/CTS or BEB. As we are considering a hop-based model, we simulated two different detection/sensing ranges for CSMA. One case had a sensing range of two hops (hence sensing all the interference and reception). Another case used one hop sensing range. We felt this was necessary as, in reality, the sensing range can vary. A large sensing range reduces the collisions but can also exacerbate the exposed node problem. If the sensing range is too short nodes are susceptible to a hidden node problem. We also simulated Pure ALOHA with BEB just to show the simplest form of MAC protocol for comparison.

5.3.5 Simulation Parameters

Table 6.1 shows the simulation parameters.

Table 5.2: Simulation Parameters

Parameters	Values
Channel bit rate	250 Kbits/s
Data packet length	1000 bits
ACK packet length	20 bits
RTS/CTS packet length	20 bits
Transmit range	200 m
Interference range	400 m

5.4 Performance Evaluation

5.4.1 Metrics for Analysis

We assume that nodes are pre-initialised with routing information (through simple Pure ALOHA schemes). To obtain statistically valid results, we run simulations on 100 different randomly generated maps. Each map is run for 100 times. The results for each map are averaged over these runs. This then produces average results for 100 different maps. We plot results as a Cumulative Distribution Functions (CDF). In this way statistical features of the data are preserved.

The metrics used in this section are summarised in Table 5.3. More details on each can be found in Section 3.2.1. In addition we also show transmissions per packet. This provides data on the number of packet retransmissions before reaching the sink node and is a simple indication of likely energy consumption.

Table 5.3: Swarming MAC performance metrics

Metric	Unit	Reason
Throughput	Erlang	Evaluates the data transfer capabilities
Fairness	Jain's fairness index	Evaluates an ability for sharing capacity between different nodes.
End-to-End delay	seconds	Evaluates the time it takes for a packet to reach sink.
Transmissions per packet	The number of retransmissions	number of packet retransmissions before reaching the sink node

5.4.2 Results

Figure 5.1 shows the throughput performance of the Swarming MAC protocol and two comparison schemes (CSMA and ALOHA). We also show CSMA results for two different sensing ranges. 50% of the values for CSMA 1-Hop sensing and the Swarming MAC protocol exhibits a throughput of 0.085 Erlangs or better, while CSMA 2-Hop sensing achieves 0.013 Erlangs or better. The results show that, in terms of throughput, CSMA with the full sensing range (2 hops) outperforms all the protocols. However CSMA with a smaller range does no better than our proposed Swarming MAC protocol. We have taken two extreme examples of CSMA here - one with maximum sensing range and another with the minimum. In reality CSMA hardware should be tuned for signal detection thresholds to optimise the throughput. Moreover, CSMA uses Physical Carrier Sensing (PCS) and RTS/CTS to prevent collisions. However the Swarming MAC protocol, just by using distributed intelligence can achieve

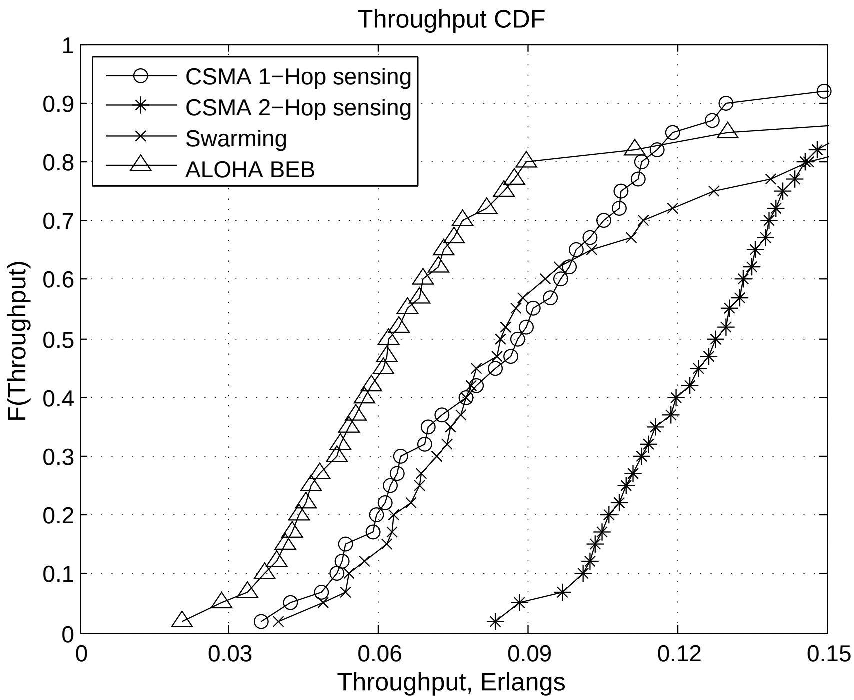


Figure 5.1: Throughput CDF

comparable throughput with much greater simplicity.

Figure 5.2 shows the end-to-end delay performance. A clear trade-off between CSMA detection ranges can be seen here. While at the full sensing range CSMA performs better in terms of throughput, it under-performs in terms of end-to-end delay when compared with the 1 hop detection range CSMA. Nevertheless, the Swarming Protocol here shows similar/better performance in terms of delay compared to both CSMA cases. In this case, over 55% of values for CSMA 2-Hop sensing show delays of 10 seconds or more. Only 20% and 25% of values for the Swarming MAC protocol and CSMA 1-Hop respectively exhibit the same performance.

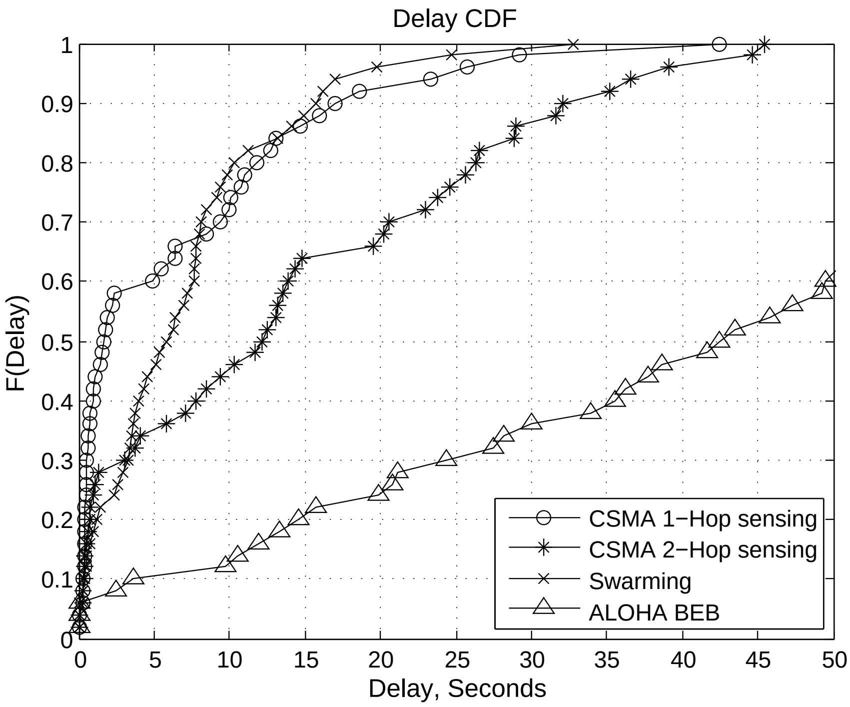


Figure 5.2: Delay CDF

Figure 5.3 provides information about the fairness of the throughput between the nodes. In this case the performance of all schemes is very close with a slight edge for CSMA. For about 50% of the cases, all protocols perform with a fairness index of 0.7 or more. However in the other 50% of the cases, the Swarming Protocol shows worse performance.

Figure 5.4 shows the number of transmissions per packet through the route for successfully-received packets. This also represents an indication of energy efficiency [62]. In this case, the Swarming MAC protocol clearly outperforms both CSMA cases. CSMA with the full sensing range also shows better results than CSMA 1 hop sensing range. The better the sensing capabilities of CSMA, the more collisions it is able to avoid, thus reducing the number of retransmissions but increasing the delay. The Swarming Protocol, however, manages to maintain a good throughput while reducing the number of transmissions per packet and delay. In 50% of cases the Swarming Protocol provides 10 or fewer transmissions per packet or less. However CSMA 2-Hop and CSMA 1-Hop show around 12 and 16 transmissions per packet respectively. Another factor to consider is protocol simplicity. In these simulations CSMA was using both hardware sensing capabilities and RTS/CTS packet transmissions to reduce the chances of collision. However, it also introduces overheads. The Swarming MAC protocol does not use any sensing or any additional messages to achieve its performance. All it

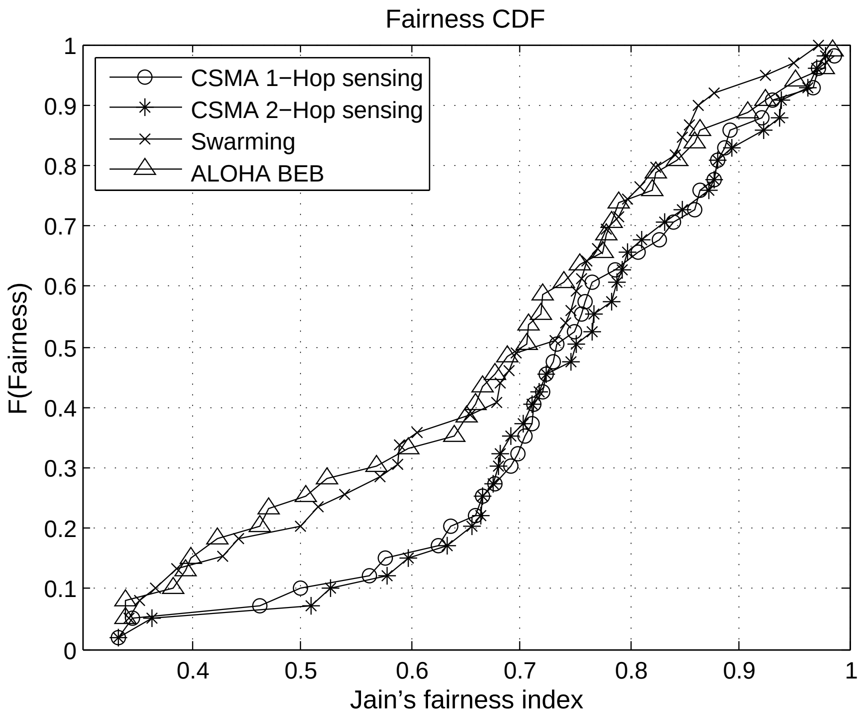


Figure 5.3: Fairness CDF

does is add small amounts of information (small overheads) to each packet and share its current fitness and delay. This in conjunction with the PSO concepts, allows the nodes (like a swarm) to create an emergent solution. The scheme builds up a route with transmission delays to prevent many collisions and reduce delay. Sometimes, however, because it is cautious, it reduces the frequency of transmission by increasing delay, therefore reducing throughput slightly. We have noticed that with larger networks there is tendency for the Swarming MAC protocol to outperform CSMA even more. The longer the routes become, the harder it becomes for CSMA to deal with the hidden node problem. However for small scale networks CSMA shows very good results as RTS/CTS and sensing can prevent most of the collisions. The performance of the Swarming MAC protocol has also been compared to ALOHA with BEB, for the purpose of demonstrating how it offers significantly improved throughput and delay performance given a similar level of complexity.

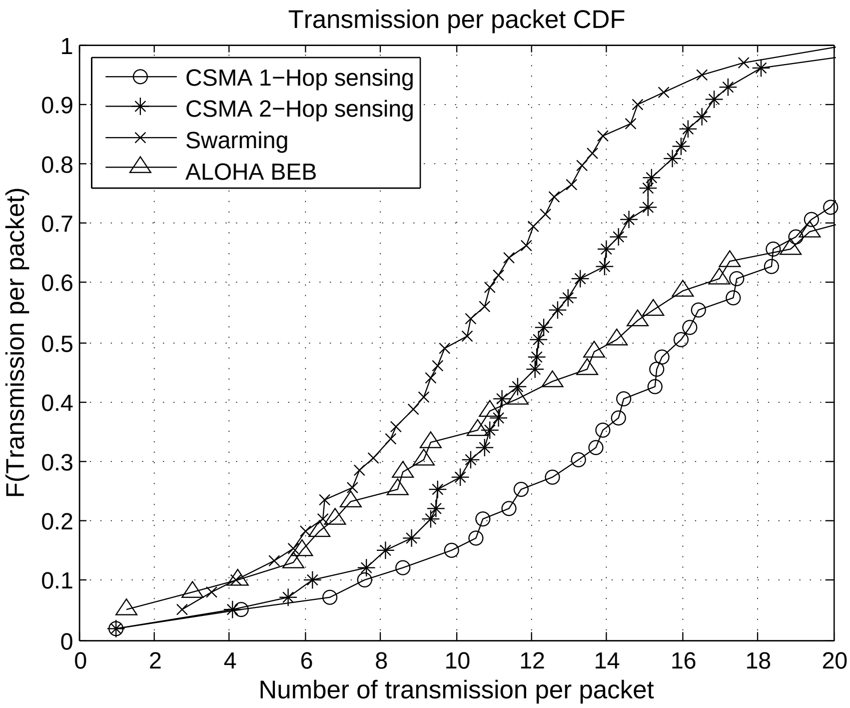


Figure 5.4: Transmissions per packet CDF

5.5 Conclusions

This chapter has presented a Swarming MAC protocol as a novel approach to MAC in a WSN. The protocol exhibits good throughput and delay performance with simplistic implementation and no sensing or synchronisation. The scheme provides a new foundation for MAC protocol design from a different DAI perspective. The comparative results with IEEE 802.11 CSMA/CA RTS/CTS show that our proposed protocol provides comparable throughput performance, better delay performance and fewer retransmissions per packet. The Swarming MAC Protocol also exhibits lower complexity and demonstrates self-organisation. However, the defined formulas are complex and difficult to analyse or predict. We also feel a more significant improvement in performance could be achieved. Instead of applying and adjusting algorithms inspired by biology and nature, in the next chapter we decide to look directly at biological systems. By using mechanisms found nature, we try to create emergent behaviour in a communication system. Chapter 6 discusses such an approach.

Chapter 6

Bio-Inspired Emergence Medium Access Control

Contents

6.1	**Motivation**	**81**
6.2	**Biological Metaphors**	**83**
	6.2.1 Task Allocation and Division of Labour in Social Insects	84
	6.2.2 Ant Clustering and Sorting	85
	6.2.3 Biological Mechanisms	86
6.3	**Emergence Medium Access Control Protocol**	**87**
	6.3.1 Protocol Design	87
	6.3.2 Simulation Parameters and Assumptions	93
	6.3.3 Results	95
6.4	**Settling Time and Theoretical Analysis**	**105**
	6.4.1 Definition of the Markov Model	105
	6.4.2 Tendencies and Convergence	111
6.5	**Conclusions**	**113**

6.1 Motivation

Imagine the scenario where an emergency service, such as fire and rescue, is required to monitor a large area of moorland for spontaneous outbreaks of brush fire [134]. Any such monitoring would be required to report on temperature and humidity levels that indicate high risk conditions and, subsequently, the movement of fire fronts. The movement of fire fronts can be highly unpredictable and poses a serious danger to personnel and equipment. This is an ideal opportunity to deploy a Wireless Sensor

Network (WSN) over a wide area from a suitable aerial platform. It should also be possible to deploy more of these low-cost nodes should there be an operational need.

This scenario presents a set of significant challenges [135]. Long-term remote operation necessitates low power usage and a very simple MAC protocol in each inexpensive node. In contrast, nodes are required to minimise end-to-end delay with no sensor node being dominant (high fairness levels). In the case of these simple nodes, only one communications channel may be available, necessitating an efficient MAC protocol to control the transmissions, ensure correct operation and achieve high throughput. Nodes will be required, at different times, to act purely as relay nodes whilst at other times, they may be additionally required to generate and place data on the network. The protocol must therefore facilitate adaptability.

Many protocols have been proposed for WSNs which offer different benefits [136, 56, 49]. Schemes that employ sophisticated synchronisation or significant information exchange to achieve organisation and performance are inappropriate in the context presented here. Yet, as the scale of networks increases, the need for some form of synchronisation and information exchange becomes overwhelming even if only at a local level.

Routing becomes a challenging task in large-scale networks as well. Dissemination of routing information and discovery of routes becomes difficult process. There are, however, many examples and proposals for good routing practices in the scientific community [137][138]. In this chapter we focus on the MAC layer.

In this chapter we present Emergence Medium Access Control (E-MAC), our biologically inspired solution based on very simple rules. We have already explored the possibility of using biologically inspired computing optimisation algorithms in organising and optimising the performance of wireless sensor network (cf Chapter 5 on Swarming MAC). However, given the limitations of the presented protocol, we decided to step down closer to biology itself rather than researching algorithms inspired by biology. This way we can better understand the emergence of complex systems in nature and their parameter-free behaviour as well as use nature as inspiration directly. "Parameter-free behaviour" refers to a system which does not require setting parameters in advance or is not sensitive to the parameters.

6.2 Biological Metaphors

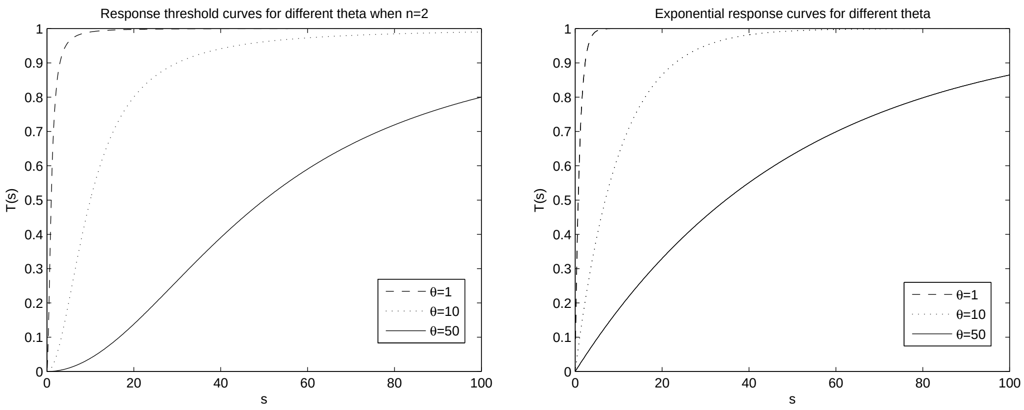
The ability of natural systems to self-organise, reorganise and provide fault-tolerant operation has inspired a huge diversity of mathematical and engineering solutions [139, 140, 141]. For example, the evolutionary metaphor (e.g. genetic algorithms and genetic programming) has enabled otherwise intractable optimisations and facilitated the discovery of novel processes, algorithms and systems [142]. Similarly, the social metaphor (e.g. particle and robotic swarms and multi-agent systems) has done the same, and contributed to the understanding of the emergent properties of complex systems [81].

E-MAC was inspired by the social metaphor.

In this case, very simple entities, generally referred to as *agents*, can offer significant benefits and highly complex behaviours when operating in groups and interacting with each other using simple rules. This *swarming* offers emergent behaviour on a higher social level [143]. Examples from nature include:

- Ant colonies which exhibit complex foraging and task allocation behaviour without central coordination [144]
- Termite colonies that can build complex structures without a global blueprint [145]
- Locust swarms which can fly in perfect synchrony in their billions, efficiently exploiting localised air streams [146]

All of these are achieved without central control, and only through very simple rules, interactions and reaction to the local agent environment, and without explicit encoding of the emergent behaviours. In each case there are up to millions of very simple entities that are continuously changing (adapting, exhibiting different behaviours) without affecting the overall, global performance. The complex behaviours arise from the interactions between individuals affecting their *local* environment. Self-organisation, adaptation and fault-tolerance are frequently the emergent properties of these systems. This simplicity and the same emergent properties correspond to what could be defined as ideal for distributed WSNs.



(a) Response threshold curves based on Equation 6.1

(b) Exponential response curves based on Equation 6.2

Figure 6.1: Response threshold functions

6.2.1 Task Allocation and Division of Labour in Social Insects

It has been observed that many species of social insects exhibit emergent task allocation and division of labour. Without the need for a leader, colonies of huge number of entities are able to organise the tasks. The process usually arises through emergence from simple actions taken by nodes. In addition, such processes are highly robust and adapt to the different needs of the colony. Bonabeau [80] proposed a model based on a response threshold that models the behaviour of ants and bees and shows emergence behaviour at the colony level for the task allocation. The response threshold defines how individuals react to their environment (stimulus). It provides a way to define a probability of taking up an action, given certain stimuli from the environment and the threshold of that stimulus. A threshold can be varied between individuals - therefore creating specialised workers. For example, in an ant colony we can consider forager and fighter ants. Foragers will have a lower threshold for collecting food and a higher threshold for fighting. Therefore they will more likely take up foraging. Fighter ants with a reversed threshold would show a higher tendency towards fighting. Nevertheless given the lack of foragers, the stimuli for foraging increases, therefore fighter ants would start to get involved into foraging tasks as well. The process also involves a learning process. If an agent is performing a task, the threshold for that task will decrease (increasing the likelihood of performing that task again). This also provides a natural process of specialisation.

For example, the probability to take up a task given a certain threshold and stimuli can be expressed as:

$$T_\theta(s) = \frac{s^n}{s^n + \theta} \tag{6.1}$$

where s is the environmental stimuli, θ - the response threshold and n defines the steepness of the curve (see Figure 6.1a).

θ essentially defines the tendency to take up an action given the environmental stimuli, so differently specialised insects would have different threshold towards certain tasks. For example, when θ is 1 in Figure 6.1a the stimuli has to be very high to increase the probability of taking up a task defined by this threshold. However, when $theta$ is 50, even a small stimuli will have a high probability of causing a response.

Another similar example (Figure 6.1b) of a response curve function is given by Plowright [147] [80]:

$$T_\theta(s) = 1 - e^{-s/\theta} \tag{6.2}$$

Similar trends arise in both functions where the probability of engaging is small for $s << \theta$ and is close to 1 for $s >> \theta$ [80].

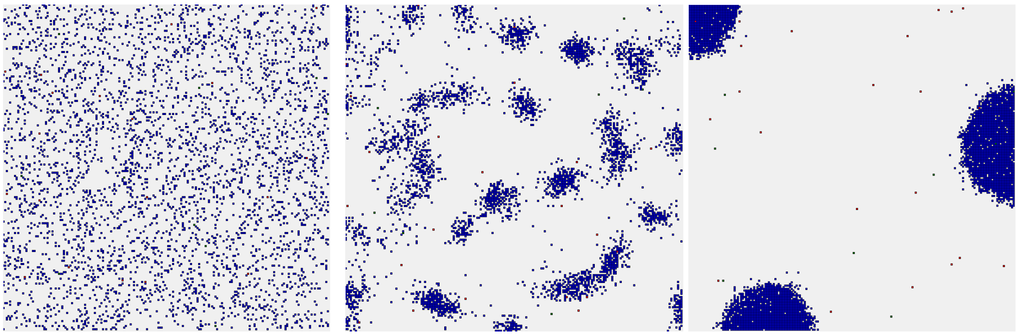
The very simple model presented here can provide very powerful and complex behaviour. Without explicitly specifying behaviour, it emerges due to social interactions between insects and the stigmergy (phenomena of indirect communication by altering environment). In addition, very robust, self-organising, scalable and adaptive behaviour is achieved.

This type of behaviour can be applied to variety of optimisation problems and resource allocation algorithms as well.

6.2.2 Ant Clustering and Sorting

Another very simple and yet very powerful example of emergent behaviour in social insects is corpse clustering or brood sorting. Ants tend to form cemeteries from dead ants, clustering them in one or more areas outside the nest. This is achieved in a distributed fashion, without any central control and can give powerful insights for a variety of optimisation strategies. In a very similar fashion, a brood is sorted in the nest to improve efficiency when taking care of it.

Daneubourg [148] proposed a model for this behaviour. At a high level this model provides an emergent positive feedback where the larger the pile of items, the more it attracts other items to be dropped in the vicinity. However the model represents



(a) t=0 (b) t=250,000 (c) t=100,000,000

Figure 6.2: Simulation of ant clustering

the behaviour of mobile agents (ie ants or other insects) and whether they pick-up an encountered item or drop it. The probability of an agent picking up an item can be given as [80]:

$$p_p = \left(\frac{k_1}{k_1 + f} \right)^2 \tag{6.3}$$

where f is the perceived fraction of items in the neighbourhood (or encountered items in specified time), k_1 is a threshold. Hence the probability of picking up an item is high when f is small compared to k_1 (not many items around) and vice versa. The probability of an agent dropping an item can be provided in a similar way [80]:

$$p_d = \left(\frac{f}{k_2 + f} \right)^2 \tag{6.4}$$

The probability of dropping items is high when there are many items around and low otherwise. If we spread items randomly in the environment and let a number of agents following these few rules randomly walk in the environment some emergent results can be seen. Figure 6.2 shows the simulation using these rules where items form clusters over time. This is one of the examples of emergence - where complex behaviour arises by itself, due to stigmergy in this particular case.

6.2.3 Biological Mechanisms

A lot of similar trends can be observed in different processes in biology/nature. The above sorting and task allocation mechanisms both follow very similar rules yet, applied to different problems, yield different complex results. In both cases we can distinguish a defined random process which is triggered via a threshold - either reinforcing the behaviour or diminishing it (positive and negative feedback). The same can be observed in variety of other biological processes such as the response of immune systems [149], oscillations caused by interacting genes and proteins [150] or even plant communities [151].

What if we mimic this random threshold based response in communication systems, such as a wireless sensor network, where we wish to control the transmission of packets and use the observed success of transmission to obtain the environmental stimuli? Would that show any emergent properties and provide a feasible solution? This is the true inspiration behind E-MAC.

6.3 Emergence Medium Access Control Protocol

When monitoring harsh environments over large areas of undulating terrain, we require cheap, simple nodes that can adapt to different communication requirements and environmental conditions without the need to tune specific system parameters. Also, network fault-tolerance is needed where nodes are likely to progressively fail at, for example, the onset of a fire front. Furthermore, adding nodes should not trigger wholesale network reconfiguration to accommodate them; only locally-affected regions should adapt without affecting global emergent behaviour.

All of this *can* be otherwise achieved with precise deployment planning and complex algorithms. Such approaches tend to introduce many tunable parameters which require more operational maintenance. Also, it is not usually possible to anticipate every scenario and its conditions. We assert that it is better exploit biological metaphors that offer appropriate emergent properties through simple rules of interaction.

The E-MAC protocol employs the notion of reaction to the intensity of stimulus from neighbouring agents. We use a stochastic approximation of the probability of successful message packet transmission as that stimulus.

6.3.1 Protocol Design

The goal of E-MAC is to provide good performance with very low complexity. The protocol is based on a simple implementation inspired by the biological social metaphor of swarm reactions to an environment. The bare minimum amount of data is shared during each data packet transmission. No *additional* transmissions are made and there is no need for carrier sensing.

This section will start with a basic overview of components in E-MAC and present an overall view of what E-MAC does. Then it will continue with detailed information on the algorithm.

Transmission Delay

Many MAC schemes or protocols employ the concept of back-off to reduce congestion and offered traffic, allowing other transmissions to compete for channel access. Once a packet is either received or dropped, back-off is usually reset. Any information on previous actions and outcomes in the environment is then lost. E-MAC employs a back-off strategy that does not subsequently reset, but either increases or decreases incrementally. We more appropriately use the term *transmission delay* which is changed after each (un)successful packet transmission. Increasing or decreasing transmission delay controls the overall transmission rate and, in the manner of conventional back-off, allows other nodes to transmit on the channel. However, unlike traditional back-off schemes, it maintains a transmission rate that becomes periodic and predictable. In effect, the transmission delay retains historical information about the environment which helps to prevent nodes from experiencing repeated congestion. We also discuss and show feasibility of transmission delay in Chapter 4.

Basic Operation

E-MAC performs a simple update action which is called when an acknowledgement is received or a time-out occurs. During updates, E-MAC simply changes the transmission delay duration depending on the acknowledgement outcomes. The adjustable transmission delay is engaged every time the MAC layer passes a packet to the physical layer for transmission. Therefore once the transmission delay is engaged, the node is not allowed to transmit packets, in the manner of conventional back-off. When the transmission delay expires, the node is again allowed to send a packet. Controlling the transmission delay can effectively allow control of the time period between packet transmissions. The way transmission delay is varied is based on a biological social metaphor. Both the averaged and most recent acknowledgement outcomes are used to define an appropriate stimulus to modify the transmission delay. Note that no explicit distinction is made between origination and back-off modes as in usual communication systems. Transmission delay is applied after every single transmission.

Robbins Monro and Probability of Success

The stimulus used to increase/decrease transmission delay is the average observed probability of successful packet transmission. Here we employ a stochastic approximation, the Robbins Monro algorithm [132]. It offers approximate averaging without the need for significant storage of past values. Additionally, it approaches an average value in a non-linear way, which provides a more realistic stimulus representation (similar to biological models in Section 6.2.1) and offers the possibility of continuous reaction. The Robbins Monro algorithm is given by Equation (6.5):

$$X_i = (1 - \alpha)X_{i-1} + \alpha X_{new} \tag{6.5}$$

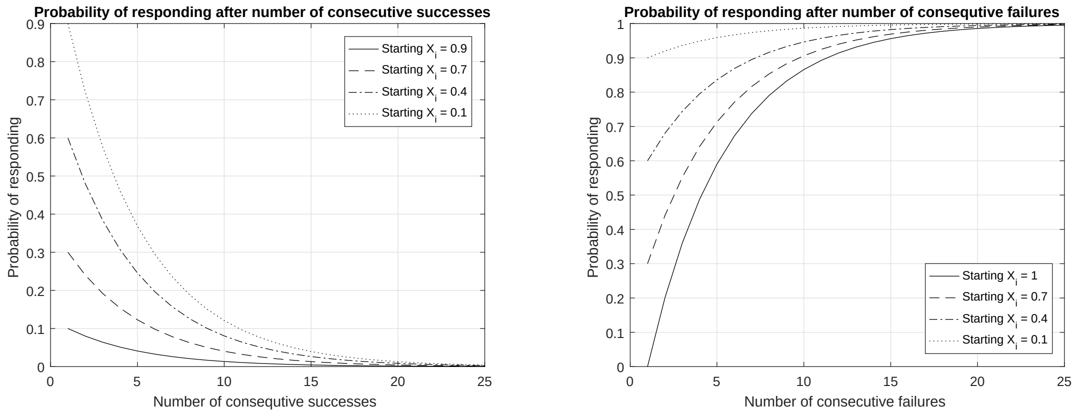
where X_i is the approximated mean after iteration i and X_{new} is a new sample. In E-MAC, X_{new} represents the outcome of the i^{th} transmission (0 or 1 for failure or success respectively). Updating X_i at each transmission outcome gives an approximate average (probability of success). It provides a way to track the current probability of success at each node. This can be then used as the intensity of stimulus for appropriate agent action. α weights current experience against the prior approximation of the mean.

This forms the response threshold discussed in Section 6.2.2. If we draw a random number between 0 and 1 and take action if the drawn number is larger than X_i then the probability of responding is $1 - X_i$. For a lower X_i value the algorithm will be more likely to respond. Figure 6.3 show the response probabilities given the starting value and number of consecutive events (success or failure). The curves also show very similar trends to the exponential response functions shown in Figure 6.1 and Equations 6.1 and 6.2.

The Basic E-MAC Algorithm

Using the stimulus proposed in Section 6.3.1 we implement Algorithm 3 that determines the changes to the transmission delay.

When a node experiences contention on the channel, there is a greater likelihood of corrective action otherwise there is none (lines 12 - 13 and 20 - 21). This corresponds to the response threshold formed by approximated probability of success ($pSuccess$) via the Robbins Monro algorithm. Consecutive failures exponentially increase the chance



(a) Probability of responding (take action) after consecutive successes starting at different X_i values

(b) Probability of responding (take action) after consecutive failures starting at different X_i values

Figure 6.3: Probability of responding after consecutive events when using Robbins Monro algorithm for approximated $pSuccess$, $\alpha = 0.2$

of corrective action, whereas consecutive successes quickly diminish the chance to act. In addition to responding to acknowledgement failures (lines 14 -15), we want a node to react to the historical performance of the adjacent downstream node (lines 22 - 23) to prevent congestion. The whole algorithm mimics the way in which swarm colonies react to their immediate environment which is usually represented as the stimuli intensity.

Experimentation shows that direct response to acknowledgement performance (`Ack Effect` at lines 12 - 19) effectively controls transmission delay. The network settles at reasonable delay values throughout and avoids collisions along a multi-hop chain. Nevertheless, congestion can build up at nodes. To alleviate queue build-up we have added another action (`Queue Effect` at lines 20 - 25). It requires each node to share its queue size with its adjacent upstream node by adding this small amount of information to every transmission and acknowledgement.

Multiple source operation

The protocol, in the form depicted in Algorithm 3, does not search for a transmission delay that gives fair operation when more than one source node exists in a multi-hop chain. Only very specific transmission delay solutions would offer perfect, collision free operation. A second source would have to have a multiple of transmission delay that is used by the first source. In addition a certain phase shift of transmission delay would be required. This could take very long to achieve. Therefore, some ex-

Algorithm 3: The E-MAC Algorithm

```
// Initialisation
set α
set change_scale
set pSuccess
set tx_delay
while network running do
    // Update
    if ack failed then
        recent_outcome = 0
    else
        recent_outcome = 1
    end
    pSuccess = (1 − α) ∗ pSuccess + α ∗ recent_outcome
    // Ack Effect
    R = generate random number between 0 and 1
    if R > pSuccess then
        if recent_outcome = 0 then
            tx_delay = tx_delay + change_scale
        else
            tx_delay = tx_delay − change_scale
        end
    end
    // Queue Effect
    R2 = generate random number between 0 and 1
    if R2 > pSuccess then
        if queue at next hop > my queue then
            tx_delay = tx_delay + change_scale
        end
    end
end
```

tra functionality is necessary. In the spirit of the biological social metaphor, the chain continues to use established information and forces nodes that become active to join the flow rather than disrupt it through dissonant transmission delays.

Using a simple extension, if a relay node also starts to function as a source node (or source node also starts to function as a relay), its packets join the flow by limiting its own transmissions to the incoming receptions. Essentially the node that is both source and relay would only transmit a packet directly after receiving one - this avoids interfering with the the built up transmission delays on the chain.

Fair Queuing

We have adopted a fair queuing strategy in E-MAC. This implies that packets in the queue from different sources are treated fairly to avoid the formation of dominant nodes. The queue prioritises packets from different sources in a round-robin fashion. In addition if there is more than one packet in the queue from a specific source, the most recent one is transmitted and older ones associated with that source are discarded. The use of such queueing is justified by the emergency environmental monitoring scenario which requires the most up-to-date data. Nevertheless, the queuing itself does not guarantee fair operation as the relayed packets can be lost due to MAC behaviour and collisions further down the chain.

This strategy may seem wasteful as not all packets coming from upstream sources are passed on, but all are acknowledged - despite some later being dropped. However, the pay-off is that the chain can quickly adapt to new sources arising along a chain using one simple protocol. Given the scenario described at the start, the availability of information from all active source nodes at a high and sustainable data rate is important.

We could have taken a more parsimonious approach where new-source nodes inform those upstream to send only every n^{th} packet. However, our experimentation shows that if a particular node then stops sourcing packets, it takes a lot longer for upstream source nodes to re-adapt and begin to send data more frequently in an appropriate fashion.

Overall protocol process

Several different events take place during wireless node operation at the MAC layer. These are packet reception from the physical payer, packet reception from the network layer, acknowledgement timeout and back-off timeout or, in E-MAC, transmission delay expiry.

Initialisation of Algorithm 3 occurs during node startup. When packet reception from the physical layer occurs, the MAC layer passes the packet to network layer if appropriate, and an acknowledgement is sent back. When packet reception from the network layer occurs, if the node is currently not receiving a packet at the physical layer and/or a transmission delay is not in progress, the node passes the packet for transmission to the physical layer immediately and the transmission delay is then engaged. Otherwise it waits until the current transmission delay expires. Once the acknowledgement is received or a timeout occurs, Algorithm 3 lines 6-27 execute to update the transmission delay value.

6.3.2 Simulation Parameters and Assumptions

The Basic Scenario

We evaluate E-MAC as a 12-node multi-hop chain, indexed from 0 to 11 (0 is the sink). All nodes use the same channel for transmission and reception. All nodes are identical and can act either as relays, sources, or both. There is no direct synchronisation between the nodes and the inter-hop distance is 200 meters. This scenario is shown for clarity in Figure 6.4. This type of scenario is also discussed in Section 3.1.1.

Propagation and Radio

A traditional hop based model is used for the communication and interference where nodes are able to transmit their data over 1 hop (nearest neighbours) but interference is experienced over 2 hops (as shown in Figure 6.4). Later, we increase the interference range to observe the adaptability and performance of the protocols in different conditions. Packets are only received correctly if no interference or other transmissions are present. We define propagation delay based on the distance between the nodes. Given that real device hardware can only perform one action, transmit or receive, in simulation nodes are not permitted to transmit if they are in a successful reception state. Such a model is also discussed in Section 3.1.6.

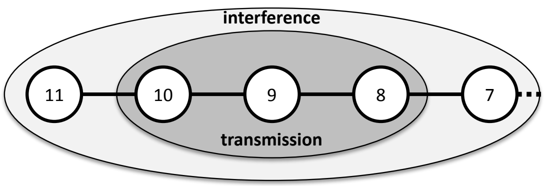


Figure 6.4: The chain scenario

Traffic

We use saturated traffic to simulate packet generation. This is to test the stability and maximum performance of the protocol. Also, we want to mimic the behaviour that would be required during critical monitoring situations where as much data as possible needs to be generated and conveyed along the chain. A new packet is generated as soon as one is successfully transmitted and that node is available to transmit again. The initial packet transmissions start within the first second of simulation according to the uniform random distribution. The purpose of this is to avoid starting multiple sources at the same time. Saturated traffic is also discussed in Section 3.1.7.

The Comparison Scheme

This is the IEEE 802.11 standard - CSMA/CA with RTS/CTS and Binary Exponential Backoff (BEB). Compared to many other much newer WSN protocols, CSMA is very low in complexity but offers good performance without synchronisation even for substantial networks. It is also perhaps one of the best known schemes which aids understanding of performance for the reader. This is why we chose it for comparison. E-MAC is of even lower complexity as it does not employ RTS/CTS messages or any hardware sensing to avoid collisions. E-MAC exploits collisions as part of the notion of reaction to the stimulus intensity of neighbouring agents. Note, the same fair queuing policy is adopted in the CSMA scheme. The comparison scheme is also discussed in more detail in Section 3.3.

Simulation Parameters

Table 6.1 shows the simulation parameters.

Table 6.1: Simulation Parameters

Parameters	Values
Channel bit rate	250 Kbits/s
Data packet length	1000 bits
ACK packet length	20 bits
RTS/CTS packet length	20 bits
Transmit range	200 m (1 hop)
Interference range	400 m (2 hops)

6.3.3 Results

Metrics for Analysis

We assume that routing would be pre-initialised using Djikstra's shortest path routing (through a simple pure ALOHA scheme). We plot results as Cumulative Distribution Functions (CDF) over 1000 simulations using different random number seeds. We use CDF because it provides an informative statistical view of protocol operation. Mean and standard deviation tables are additionally provided.

The metrics used in this section are summarised in Table 6.2. More details on each can be found in Section 3.2.1.

Table 6.2: E-MAC performance metrics

Metric	Unit	Reason
Throughput	Erlang	Evaluates the data transfer capabilities
Fairness	Jain's fairness index	Evaluates the ability to share capacity between different nodes.
End-to-End delay	seconds	Evaluates the time it takes for packet to reach sink.

Performance evaluation

Theoretically the maximum throughput that could be achieved on a long multi-hop chain is 0.25 and 0.2 Erlangs for 2 hop and 3 hop interference cases respectively. Given a chain network where nodes can communicate over 1 hop but interfere within 2 hops, transmissions can only happen at every 4th node at the same time. In the case of 3 hop interference, only every 5th node can be transmitting at the same time.

Figures 6.5 & 6.6 and Table 6.3 show throughput performance of E-MAC and the comparison CSMA scheme. There are two sources on a chain - one at the end furthest from the sink node (node 11) and one in the middle (node 5). For the 2-hop interference model, E-MAC significantly outperforms CSMA in all the simulations. For 3-hop interference, it is better in 97% of the simulations. Even though CSMA employs interference detection on the channel and avoids the hidden node problem through RTS/CTS, it is still not fully able to exploit channel capacity. The significantly simpler E-MAC protocol nevertheless achieves much better results. Furthermore, the results also incorporate the period during which E-MAC is self-organising and settling towards the best transmission delay. This self-organisation of transmission delay indirectly synchronises the network to source transmissions, thereby avoiding collisions. If a source places packets on the network at the correct rate, they will move sufficient hops downstream before the next packet is sent, thereby avoiding collisions. Through the emergence of rate searching, hop-by-hop flow control occurs. Once settled to the correct rate, the end-to-end flow control becomes operational and throughput quickly rises close to the theoretical bounds. Under E-MAC, without the need for an explicit timing mechanism, the network achieves very good throughput performance.

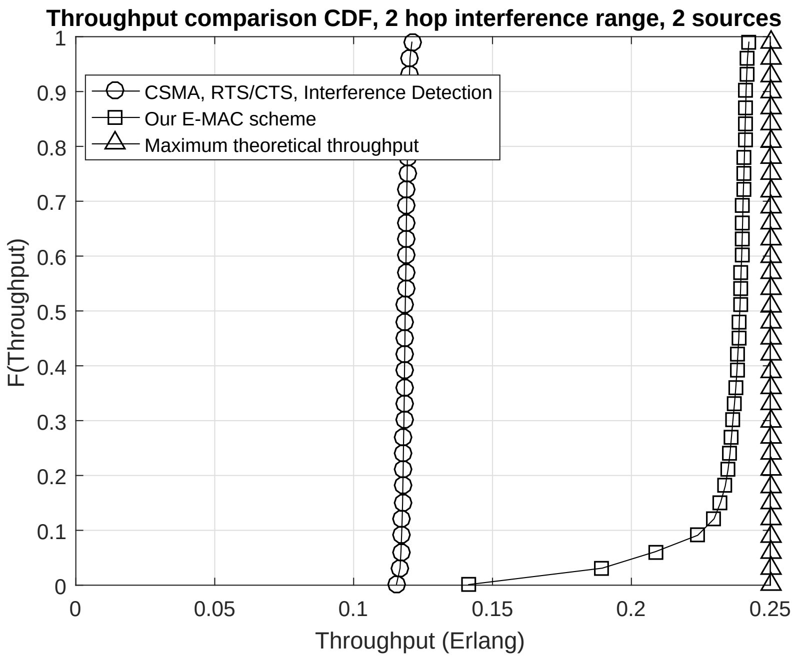


Figure 6.5: Throughput for system with 2 sources and 2 hop interference range

Table 6.3: Throughput (Erlangs) (mean ± standard deviation)

	CSMA	E-MAC	Maximum Theoretical
2 hop interference	0.1186 ± 0.0010	0.2344 ± 0.0140	0.25
3 hop interference	0.1015 ± 0.0009	0.1853 ± 0.0186	0.20

Figure 6.7 and 6.8 show the packet end-to-end delay results for 2 hop and 3 hop interference respectively. Both graphs represent delay for packets arriving from 2 different sources (nodes 11 and 5) for both schemes. Again, significantly better end-to-end delay performance can be seen using the E-MAC protocol. Note, from Table 6.4, that end-to-end delay statistics for E-MAC and CSMA delay performance are both fairly consistent over the 1000 simulations. The minimal latency of E-MAC also arises through the same rate interactions. Once source nodes find a good transmission delay, the packets travel through the route with minimal collision or interference. This en-

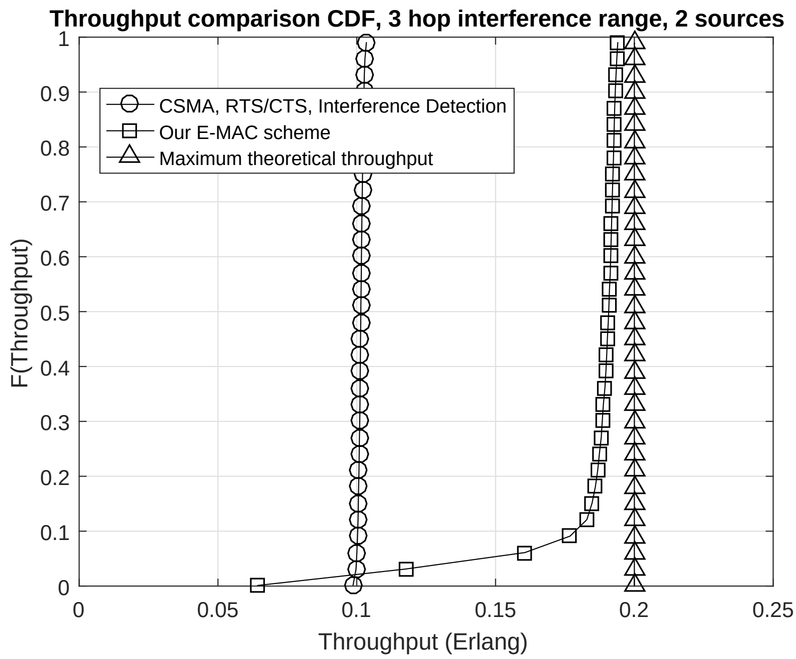


Figure 6.6: Throughput for system with 2 sources and 3 hop interference range

sures that a packet is not held up at any node due to back-off or failure. The outcome is reduced end-to-end delay.

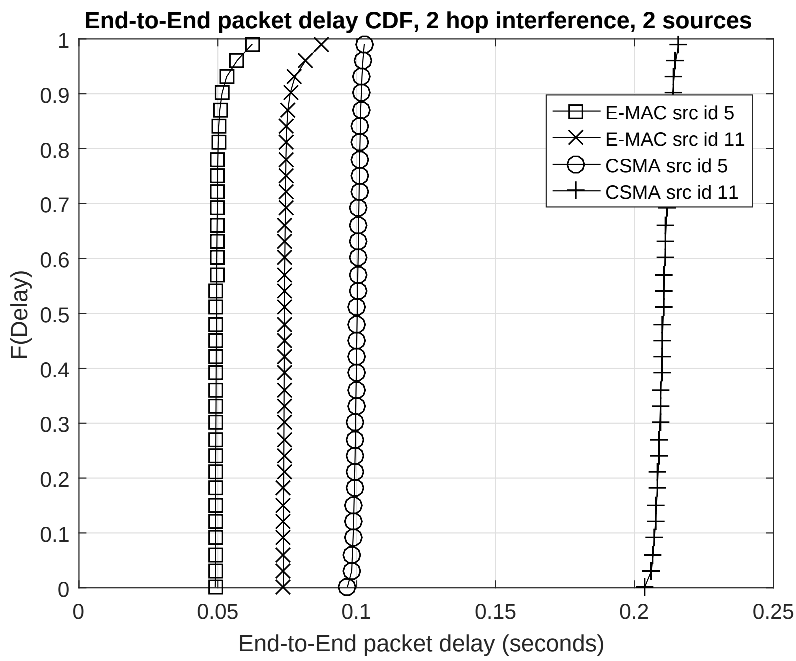


Figure 6.7: End-To-End delay for system with 2 sources and 2 hop interference range

Figure 6.9 and Table 6.5 show fairness results (Jain's Fairness Index, as described in Section 6.3.3) for the 2 hop and 3 hop interference models using the E-MAC and CSMA protocols. The results indicate ideal performance from the E-MAC scheme and near

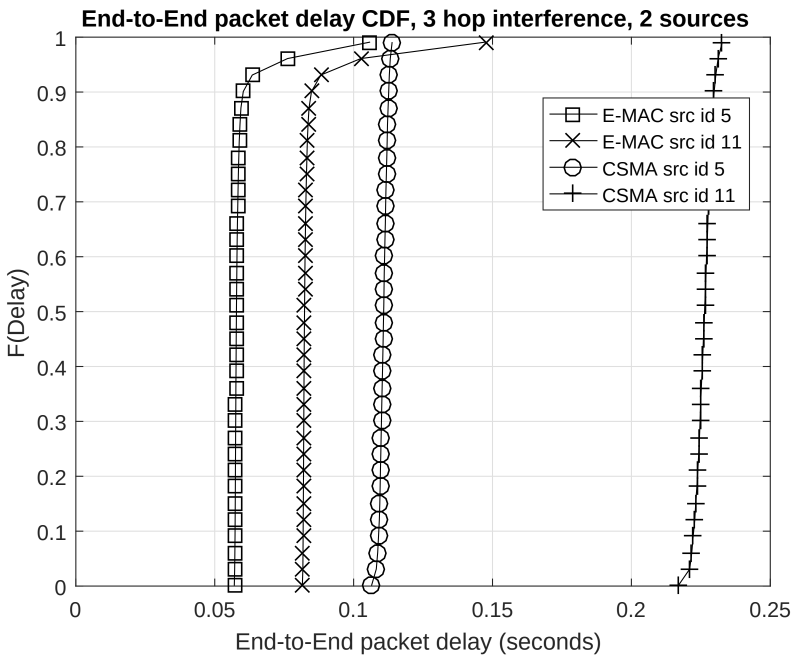


Figure 6.8: End-To-End delay for system with 2 sources and 3 hop interference range

Table 6.4: End-to-end delay (mean ± standard deviation)

		CSMA	E-MAC
2 hop interference	src 5	0.1002s ± 0.0011	0.0503s ± 0.0027
	src 11	0.2104s ± 0.0024	0.0748s ± 0.0027
3 hop interference	src 5	0.1108s ± 0.0014	0.0598s ± 0.0079
	src 11	0.2262s ± 0.0029	0.0847s ± 0.0104

ideal performance from CSMA. Despite both schemes using the same fair queuing mechanism, some packets get lost under the CSMA protocol, due to collisions. This slightly reduces CSMA fairness.

To extend the scope of the results to show the performance of E-MAC with different numbers of source nodes ranging from 1 to 10, we consider the chain scenario where the specified number of source nodes are placed at the end of the chain. These results are shown in Figures 6.10 and 6.11, which exhibit the same trends as the previous results. Note that the throughput results for the two node case in Figures 6.10 differ slightly from the results presented in Table 6.3, due to the different placement of source nodes in the original topology (where one of the two source nodes is located in the middle of the chain). E-MAC clearly outperforms CSMA RTS/CTS and performance is very close to theoretical boundaries in the scenario. We can see a sudden variance in CSMA RTS/CTS fairness results. Even with a fair queuing, CSMA

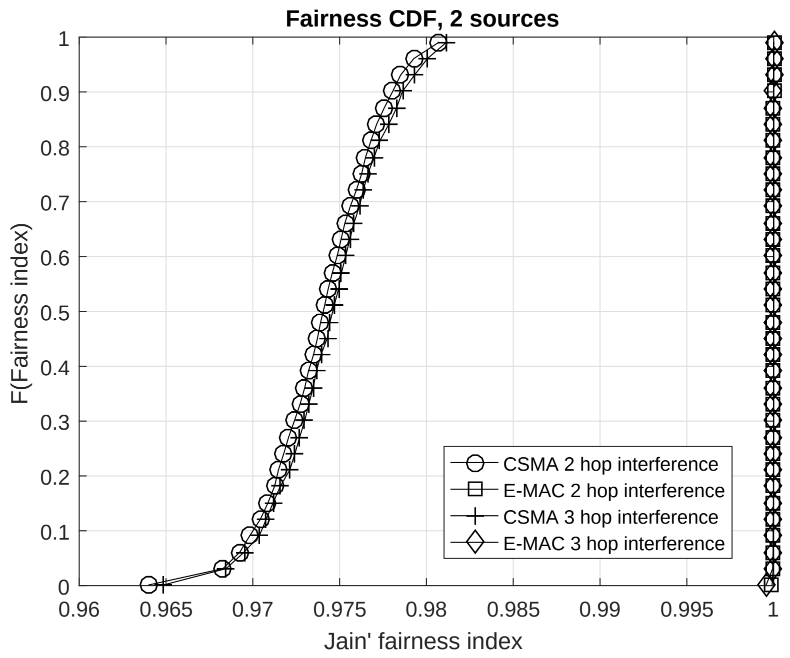


Figure 6.9: Fairness for system with 2 sources

RTS/CTS seems to become unstable once a clear dominating node appears in the network. Under 10 source operation, every-node in the network is a source. The source closest to the sink is only 1 hop away. This source, due to its success and quick delivery, starts dominating the network, thereby operating as a single hop (breaking throughput bounds) and blocking out other transmissions (significant drop in fairness).

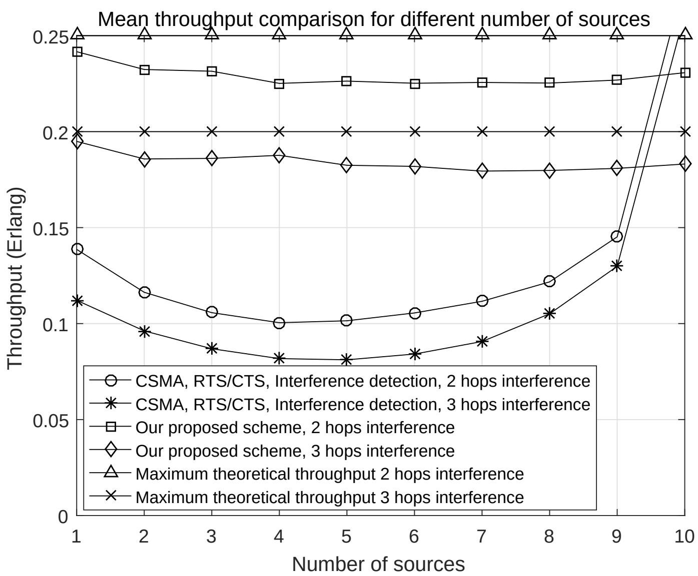


Figure 6.10: Mean throughput comparison for different numbers of sources

Overall, we have observed significant performance benefits of E-MAC over CSMA

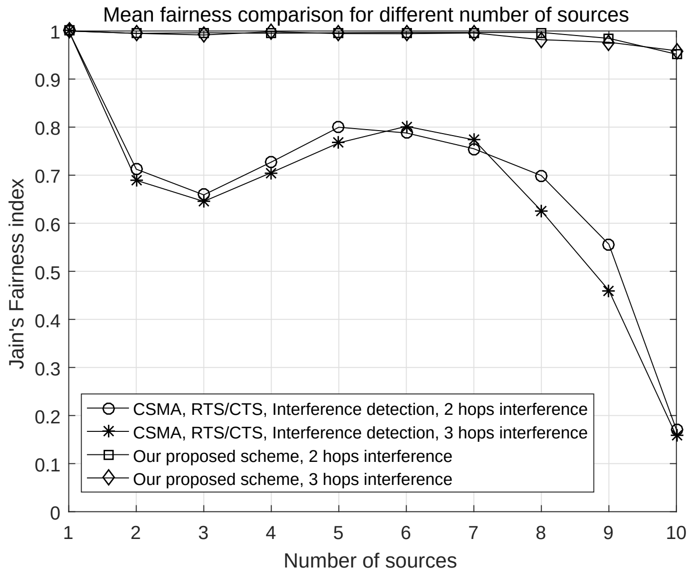


Figure 6.11: Mean fairness comparison for different numbers of sources

Table 6.5: Fairness (Jain's Fairness Index) (mean $\pm$ standard deviation)

	CSMA	E-MAC
2 hop interference	0.9740 ± 0.0031	$\sim 1 \pm 0.00000797$
3 hop interference	0.9745 ± 0.0032	$\sim 1 \pm 0.00002833$

in two measured performance criteria (throughput and end-to-end delay) and better performance for fairness. The simplicity of E-MAC, in terms of hardware and computational requirements, is truly encouraging. The basis for this is the exploitation of emergence through simple exchanges of information piggy-backing on an otherwise trivial MAC protocol. The network is able to self-organise and adapt to different scenarios without requiring extra parameters or a shift in the simple agent behaviours. Emergence provides us with indirect synchronisation which boosts throughput and reduces end-to-end delay. Furthermore, the reduced number of collisions improves overall fairness.

Parameters

Earlier, we stated that a property of biological systems is a lack of scenario-specific parameter tunings. The reader will have noted that two parameters seem to abuse this notion in E-MAC: α and $change\_scale$.

Figure 6.12 and 6.13 are contour plots which show the variation of throughput when α and $change\_scale$ are varied. Actual $change\_scale$ values are related to packet length.

It is clear that performance is generally insensitive to these parameter values. However some trends can be observed.

We have set $change\_scale$ values to be 10% of packet length, where E-MAC performs well. Greatly increasing the value causes the resolution of transmission delay to be too coarse so that E-MAC does not perform well. An excessively low value causes very slow settling and adaptivity.

We also see from Figure 6.12 and 6.13 that the value of α should be in the general region of 0.2. Choosing extreme values will cause the transmission delay to settle slowly (low α), or away from a value commensurate with good throughput and reduced ability to adapt (high α). In fact when α approaches value of 1, the Robbins Monro algorithm no longer tracks past values and essentially only line 16 in Algorithm 3 remains active. The protocol will only respond to the last acknowledgement outcome, leading to unstable behaviour. Nonetheless almost any chosen value will provide better performance than that achieved by CSMA.

The same observations can be seen in the end-to-end delay performance for different parameter values given in contour plots in Figures 6.14, 6.15, 6.16 and 6.17.

It is important to note from contour plots that, given almost *any* values for these parameters, in the scenarios presented, E-MAC will perform better than CSMA.

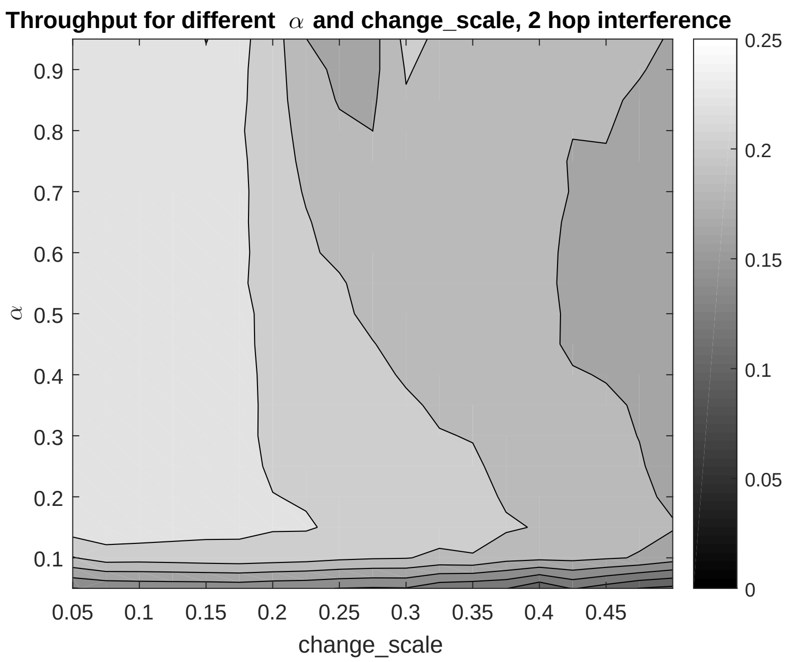


Figure 6.12: Throughput variation when α and $change\_scale$ are varied

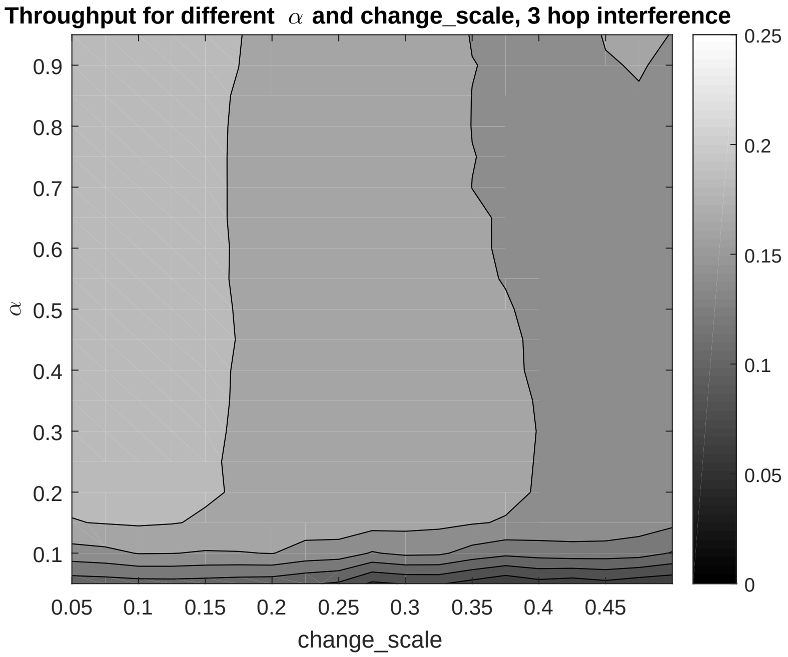


Figure 6.13: Throughput variation when α and $change\_scale$ are varied

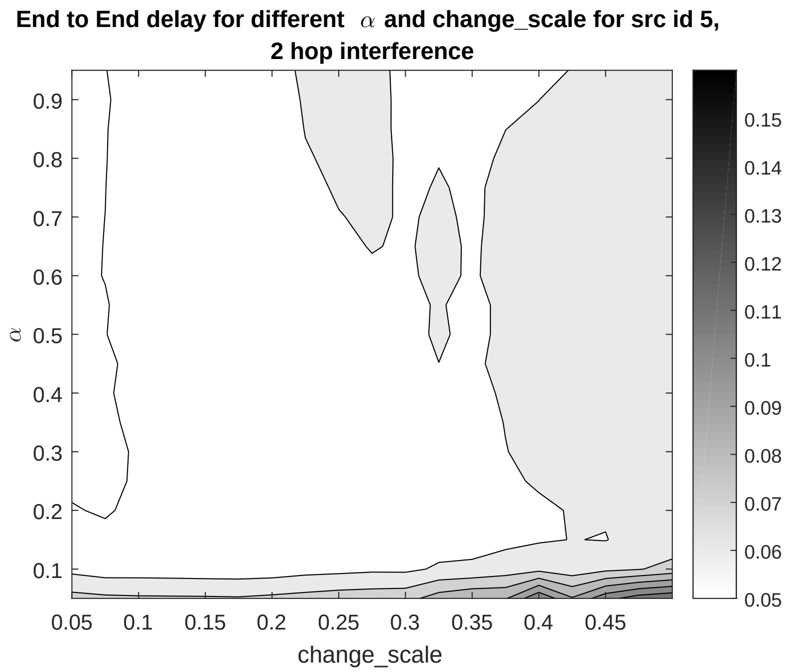


Figure 6.14: End-to-end delay variation from source 5 when α and $change\_scale$ are varied

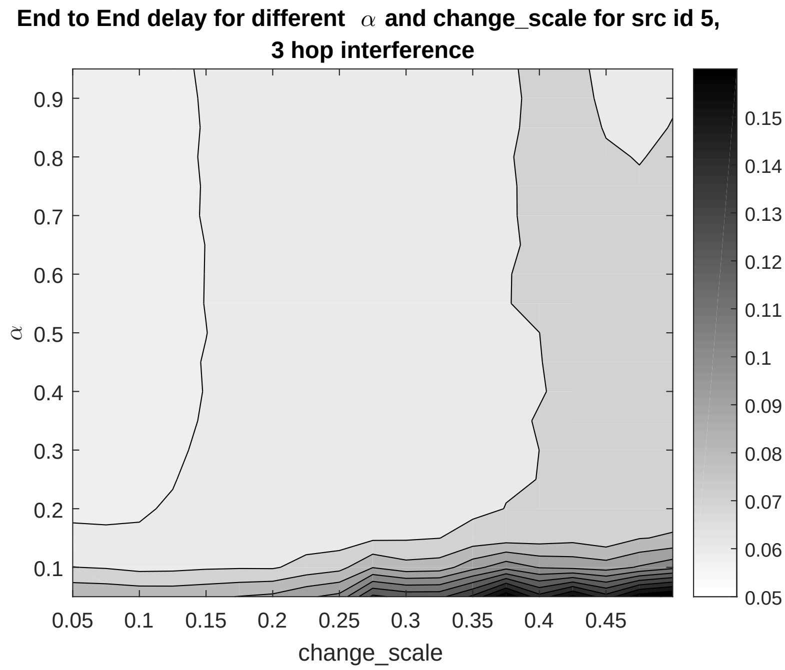


Figure 6.15: End-to-end delay variation from source 5 when α and $change\_scale$ are varied

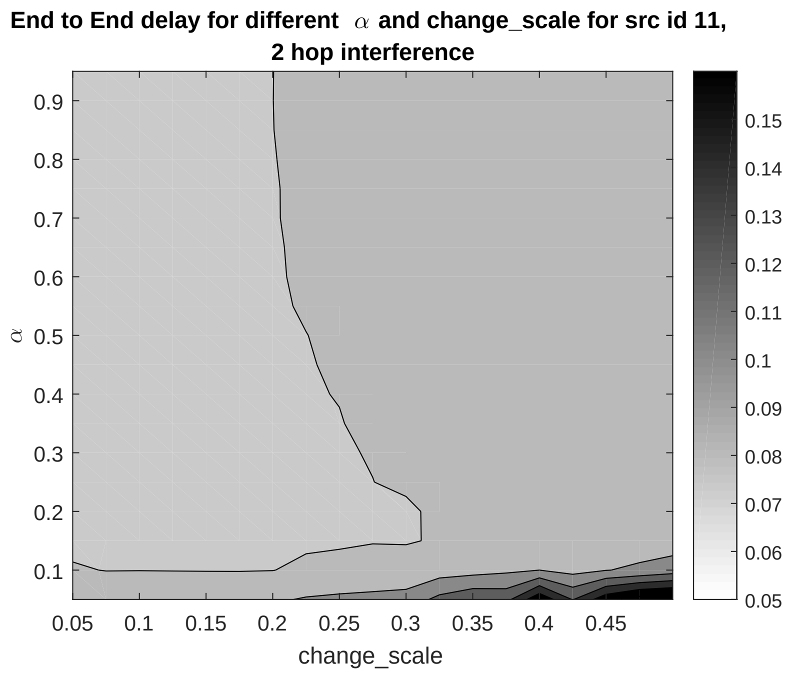


Figure 6.16: End-to-end delay variation from source 11 when α and $change\_scale$ are varied

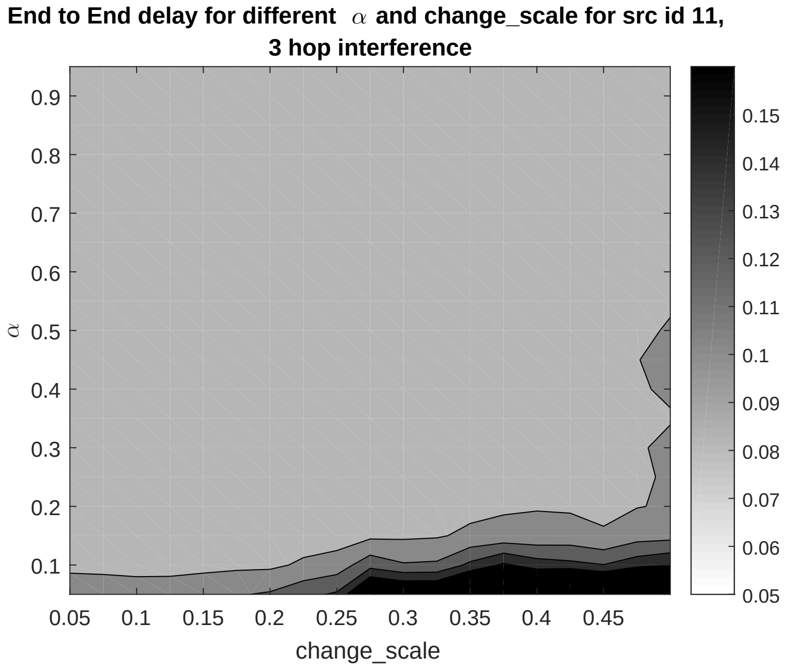


Figure 6.17: End-to-end delay variation from source 11 when α and $change\_scale$ are varied

6.4 Settling Time and Theoretical Analysis

To further reinforce knowledge of E-MAC operation we provide theoretical models of its operation and settling time. We also show the general behaviour trends and show that E-MAC will always tend to settle.

6.4.1 Definition of the Markov Model

By default, the algorithm is based on memory in the system, through the Robbins Monro (RM) algorithm that calculates $pSuccess$ (affecting the urge to change transmission delay). In addition, transmission delay varies, which also affects what the outcome will be (lower transmission delay - greater chance of collision).

In order to satisfy the Markov property, state transitions cannot depend on previously visited states. Therefore we can specify each state of the system as having a specific transmission delay value and specific $psucc$ values. This removes their dependency on previous states (removing the memory component from the process).

Consider a simple multi-hop chain with 1 source at the end. This source will send packets that pass along the chain to the sink at the other end. We use the previous scenario from Figure 6.18. This entails that we have 1 hop transmission and 2 hop interference ranges where the source is node 11 and the sink is node 0. This specifies a simplified environment for the system in order to build a state based model to observe the tendencies of the system and demonstrate its settling. We will only look at the state value of the source rather than all nodes due to complexity. We will also not model the `Queue Effect` (in Algorithm 3 in the Markov model. Nevertheless the benefits of the `Queue Effect` will be discussed.

Transmission delay specification and its effect on states

Given a simple chain network as shown in Figure 6.18, we can observe the relationship between transmission delay and the chance of collision (at node 9). We have simulated different static transmission delays in such scenario. The trends across different transmission delay ranges arise (note: the transmission delay value is normalised to the packet transmission duration):

Transmission delay greater than or equal to 4.1: Due to packets travelling sufficiently far away along the chain, no collisions will occur (100% chance of successful

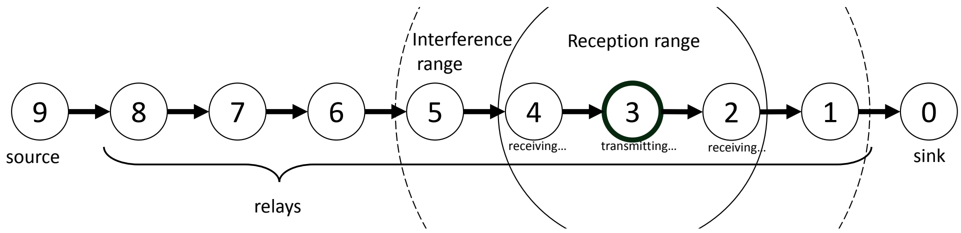


Figure 6.18: Multi-hop chain network

transmission). This relates to the maximum possible throughput on such multi-hop chain (0.25 Erlang). Delaying transmission for 4 packet lengths after each transmission would relate directly to offered traffic of 0.25 Erlangs. Additional 0.1 delay is added due to propagation delays and waiting for an acknowledgement.

Transmission delay between 3.1 and 4.1: The first sent packet will go through. However, at node 7 the acknowledgement will be interfered with due to a new transmission from node 9. In addition the transmission from node 9 will fail due to the acknowledgement sent from node 6 interfering with its reception at node 8. After collision is ended, node 7 will retransmit the packet and receive the acknowledgement successfully. Node 6 will detect a duplicate and drop it instead of passing it through. The new transmission from node 9 will follow the same process again. This means that every other packet sent from node 9 will fail. In other words there is a 50% chance of successful transmission).

Transmission delay between 2.1 and 3.1: In this situation, once node 9 sends the packet, node 8 has enough time to pass it to node 7. However node 7 starts transmission at the same time as node 9 starts a new one. However node 9 does not interfere with reception at node 6. Hence the packet goes through but the acknowledgement coming back to node 7 is lost due to interference from node 9. This causes nodes 9 and 7 both to retransmit and fail again and again (even though a packet actually got through). This chain of events ends once the number of retry limit is reached. Therefore there is only 12.5% chance on average of a successful transmission at this stage.

Transmission delay between 1 and 2.1: Packets transmitted from node 9 will reach node 8 but a new packet from node 9 will be sent during the transmission of a packet from node 8 to node 7. The only successful transmission at node 9 will appear to be the very first one. The other successful transmission would occur when node 8 reaches a retransmission limit hence stopping transmission. Therefore the same situation as in the previous cases happens and there is only 12.5% chance of success from node 9 perspective.

Simulations were run with static transmission delays set at the sources to verify these statements. The results are shown in Figure 6.19.

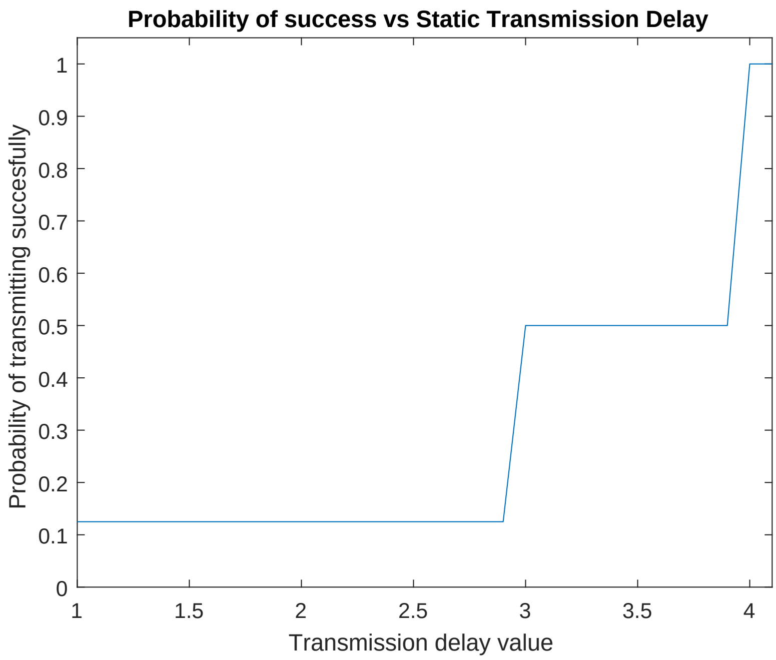


Figure 6.19: Probability of successful transmission given static transmission delay

RM $pSucc$ specification and effect on states

Based on the Algorithm 3, $pSucc$ has a direct effect on the likelihood of increasing or decreasing transmission delay. The higher the $pSucc$ value the less chance there is to act at all. Probabilities to act and not to act can be modelled as:

$$p_{act} = 1 - pSucc \tag{6.6}$$

$$p_{not\ act} = pSucc \tag{6.7}$$

However, it is not straightforward to set discrete $pSucc$ values to be used for states. As $pSucc$ is varied through the RM algorithm it creates a non linear space for change in $pSucc$. Figure 6.20 shows the general change in $pSucc$ given all successess and failures of transmitting starting at 0 and 1 respectively. One can immediately notice that this would form a continuous space of values since failure and success can occur at anytime and given the number of events it could potentially change $pSucc$ value to anything between 0 and 1. However we can adopt a technique used by Kosunalp [152] to discretise the space based on the change in value going upwards (all success). Once the transition occurs that changes the $pSucc$ value we can round it to the nearest available value based on a defined scale. Because the scale is non linear and most states are at higher $pSucc$ values. This enables us to model more accurately the region closer to convergence, reducing inaccuracies caused by modelling.

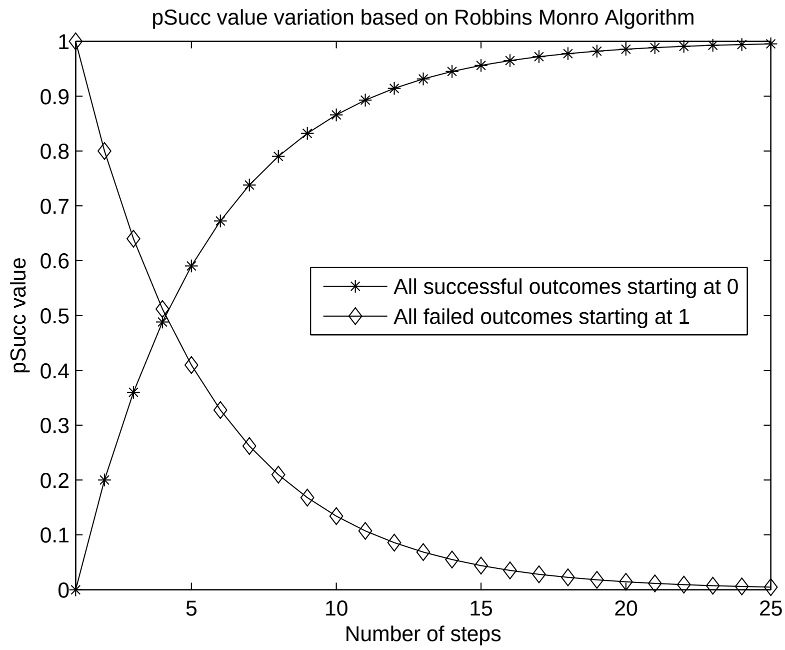


Figure 6.20: Change in pSucc value based on Robbins Monro algorithm given success or failure ($\alpha = 0.2$)

The State Grid

Based on specified $txDelay$ and $pSuccess$ values we can define a grid of states. Varying values of txDelay can be represented horizontally and varying discrete $pSucc$ values can be spaced vertically as shown in Figure 6.21.

Essentially there are four possible transitions at each state. These are:

1. Fail to transmit and act
2. Fail to transmit and do not act
3. Succeed in transmitting and act
4. Succeed in transmitting and do not act

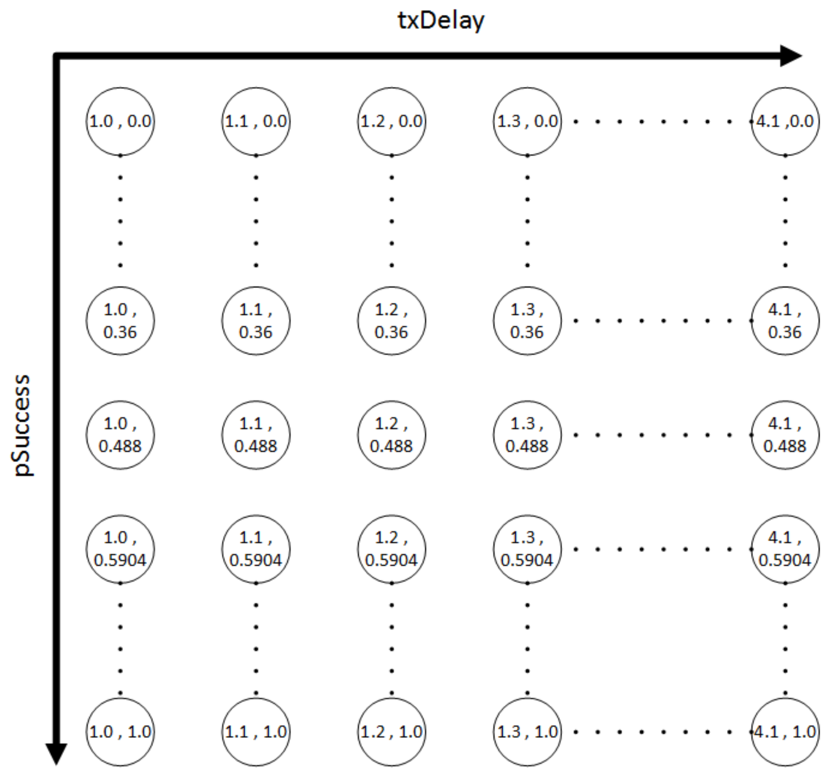


Figure 6.21: State Grid with states holding value of transmission delay and $pSucc$

Transitions 1 and 3 consider acting. Therefore these transitions will change the $txDelay$. Transitions 2 and 4 will not change $txDelay$ (do not act). Nevertheless every transition will change the $pSucc$ value as the event is only considered when success or failure happens which triggers the calculation of new $pSucc$ value. Given the transitions we can also visualise the general movement across the grid in terms of directions. Figure 6.22 shows the movement caused by transitions in terms of next state values.

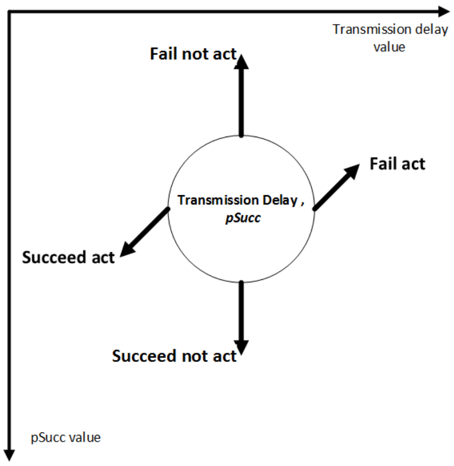


Figure 6.22: Transitions directions in terms of next state values

Probability and next state definitions

We can define the probabilities of failure or success based on the state transmission delay values (from Figure 6.19):

$$p_{succeed} = \begin{cases} 0.125 & \text{transmission delay } \in \text{ [1,3]} \\ 0.5 & \text{transmission delay } \in \text{ [3.1,4]} \\ 1.0 & \text{transmission delay } > 4 \end{cases} \quad (6.8)$$

$$p_{fail} = 1 - p_{succeed} \quad (6.9)$$

Decision to act or not to act is dependant on the $pSuccess$ value calculated via the RM algorithm ($\alpha = 0.2$ as defined at the end of Section 6.3.3). Each state has this value. However, based on the probability of success of recent transmission, this value will be changed for the transition to the next state. We can define two different values for $pSuccess$ given success or failure occurs:

$$pSuccess_{after\ success} = pSuccess \cdot (1 - \alpha) + \alpha \cdot 1 = pSuccess \cdot 0.8 + 0.2 \quad (6.10)$$

$$pSuccess_{after\ failure} = pSuccess \cdot (1 - \alpha) + \alpha \cdot 0 = pSuccess \cdot 0.8 \quad (6.11)$$

Given these probabilities, we can also define the probabilities to act or not act given success or failure:

$$p_{\text{act } if\ fail} = 1 - pSuccess_{after\ failure} \quad (6.12)$$

$$p_{\text{not } act\ if\ fail} = pSuccess_{after\ failure} \quad (6.13)$$

$$p_{\text{act } if\ succeed} = 1 - pSuccess_{after\ success} \quad (6.14)$$

$$p_{\text{not } act\ if\ succeed} = pSuccess_{after\ success} \quad (6.15)$$

Finally we can define the probabilities of moving to other states as shown in Figure 6.22:

$$p_{fail\ not\ act} = p_{fail} \cdot p_{not\ act\ if\ fail} \quad (6.16)$$

$$p_{fail\ act} = p_{fail} \cdot p_{act\ if\ fail} \quad (6.17)$$

$$p_{succeed\ not\ act} = p_{succeed} \cdot p_{not\ act\ if\ succeed} \quad (6.18)$$

$$p_{succeed\ act} = p_{succeed} \cdot p_{act\ if\ succeed} \tag{6.19}$$

These equations define the transition probabilities but we also need to define the next state for these transitions. In both non-act cases, transition will be to the state of the same transmission delay. The next state $pSuccesss$ value will be $pSuccess_{after\ success}$ or $pSuccess_{after\ failure}$ (rounded to nearest discrete value) based on the outcome. The next state $pSuccess$ value for act cases will follow the same principle but transmission delay for the fail case will be incremented by 0.1 and decremented by 0.1 for the success case.

This model is able to accurately simulate transitions. However, the settling time would be hard to define due to ambiquity caused by the transmission delay itself. Transmission delay prevents the system from taking any action based on its value. For example if the given transmission delay is 3, the system will not start a transmission for 3 packet lengths and that will prevent any transitions. Nevertheless we can model the timing via probabilities by stating that the probability to stay in the same state is:

$$p_{stay} = 1 - 1/transmission\ delay \tag{6.20}$$

In addition if we we multiply all the transitions probabilities by $1 - p_{stay}$ we are able to normalise transitions to packet lengths and therefore define a realistic settling delay time.

6.4.2 Tendencies and Convergence

Based on the transition probabilities we can interpret general movement tendency between different states in a grid based on their values. Figure 6.23 shows such results and the general movement tendencies from particular points on the grid. Arrow directions are formed based on the movement directions and their probabilities from Figure 6.22. It clearly shows that from any point in the grid the general movement is towards the state of transmission delay value 4.1 and $pSuccess$ value of 1 (which is the final settled state with best performance). Values lower than 4.1 indicate collisions therefore lowering performance. Values larger than 4.1 cause system to not utilise the capacity. Theoretically, a maximum value would be 4 (assuming no acknowledgement delays and propagation delays).

Figure 6.24 shows the results of settling time calculated through running Monte Carlo

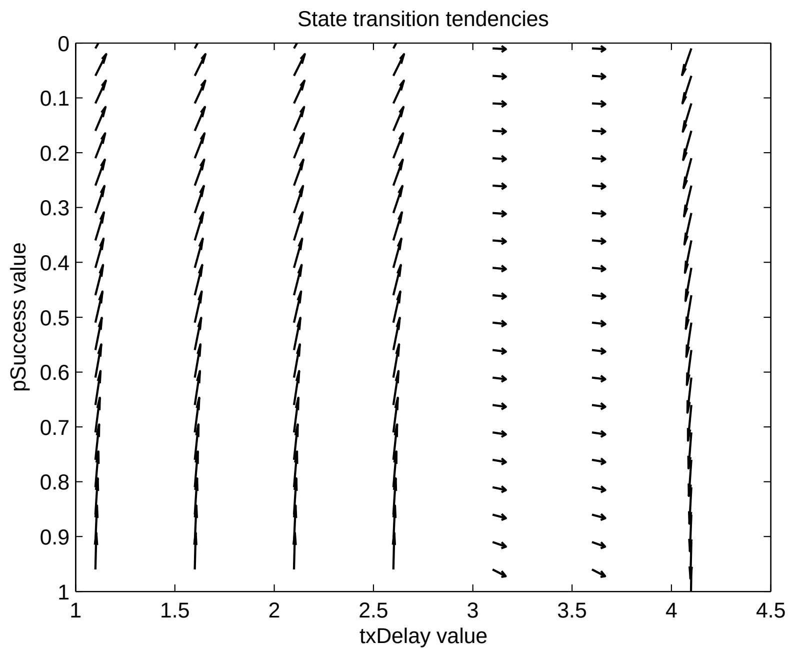


Figure 6.23: State transition tendencies

simulations on this Markov model and simulating the E-MAC protocol with a single source and a single sink. The Markov model was run by drawing random probabilities and deciding which state to jump to next based on the state values. The default starting state used was $txDelay$ of 1 and $pSuccess$ of 0.5. CDFs are plotted from a 1000 runs. Uniform random number generator was used. The original simulation results are very close to those from our theoretical analysis. However the E-MAC simulation results show a slightly faster settling time than anticipated by the Markov model. This happens due to other nodes operating in the system. The Markov model was built on the assumption that there is only one node changing transmission delay for us to be able to model memoryless states. However in E-MAC every node contributes to the system and can affect the source nodes in a positive manner to find a quicker solution.

Figure 6.25 shows the transmission delay value after 100s of running. Both the theoretical model and simulation results are shown which are comparable. The difference is that the simulation tends to have 3.8% of values settled higher than 4.1. This happens due to nodes overshooting slightly (higher than 4.1 $txDelay$) due to neighbouring nodes affecting the environment (causing additional collisions) and then settling prematurely. Nevertheless the settled delay values still correspond to significant performance improvement over the comparison schemes (these values also explain the slightly lower throughputs in Figures 6.5 and 6.6).

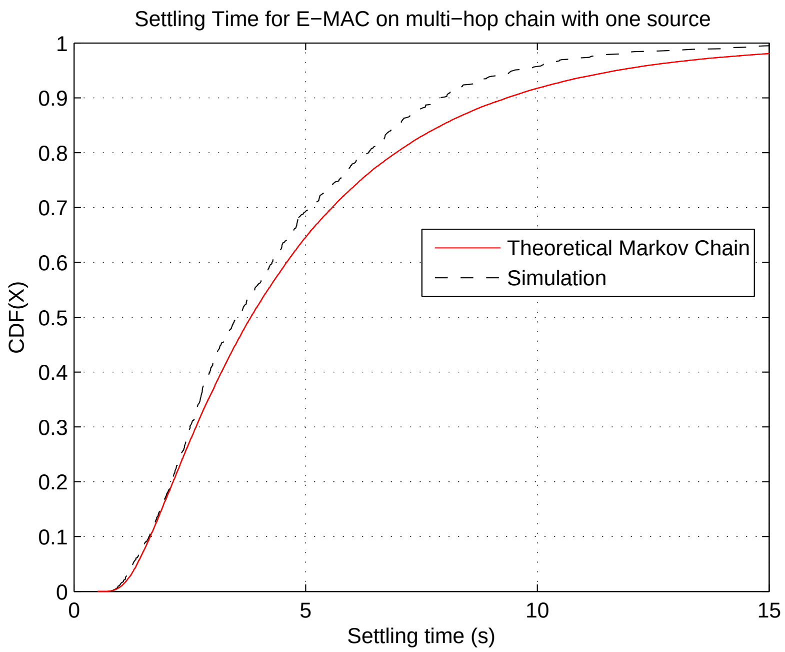


Figure 6.24: Theoretical and simulation values for settling time of E-MAC

In addition, Figure 6.25 presents the results of simulation with the `Queue Effect` implemented. Without the `Queue Effect`, about 0.8% simulations and 0.3% theoretical model values show lower than 4.1 transmission delay. This corresponds to the node not settling after 100s of simulation. Nevertheless adding the `Queue Effect` improves this behaviour significantly (all settled) at the cost of slightly more prematurely settled values higher than 4.1. In addition, the `Queue Effect` prevents bottlenecks forming on the chain and ensures queues are stable.

6.5 Conclusions

In this chapter we have discussed some notions derived from a biological metaphor and applied them to the development of a new type of MAC protocol for WSNs. E-MAC follows very simple rules based on the reaction of social agents to the intensity of a localised environmental stimulus. Without explicit synchronisation and using very simple hardware it is able to out-perform its comparator, the widely-known IEEE 802.11 CSMA/CA RTS/CTS scheme. Throughput, end-to-end delay and fairness were compared using multi-hop chain networks. E-MAC exhibits self-organisation, flow control on both hop-by-hop and end-to-end basis, indirect synchronisation between the nodes as packets are relayed and minimal latency. Its parameter insensitivity means that it can be adopted in different environmental conditions given same

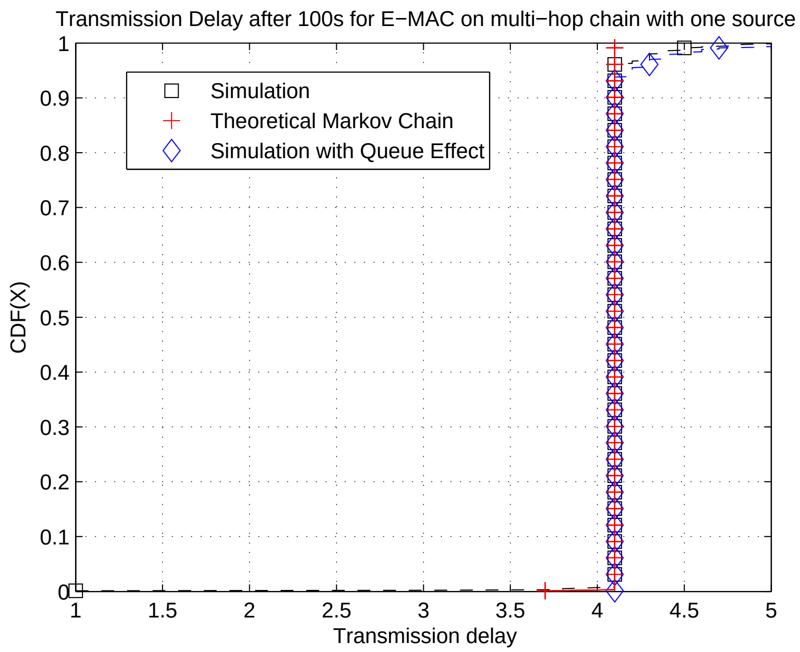


Figure 6.25: Theoretical and simulation based results for transmission delay values after 100s

topology without the need for specific set-up tuning. The results and improvements for different topologies are discussed in Chapter 7.

We have also developed a simplified Markov model to analyse the general algorithm behavioural tendencies and convergence properties. The theoretical results closely match the simulation results and provide some insight into the behaviour of the algorithm. The analysis explains the shape of the curves in our simulation results as well as the effects of additional processes in the algorithm.

Chapter 7

Enhancing The Emergence Medium Access Control protocol

Contents

7.1 Motivation **116**
7.1.1 E-MAC Lock-Up Description 116
7.2 Biological Inspiration **117**
7.2.1 Firefly Behaviour and Synchronisation 117
7.2.2 De-Synchronisation 118
7.2.3 Diffusion 120
7.3 E-MAC+ **121**
7.3.1 Protocol Design 121
7.4 Empirical Evaluation on a Merging Chain **123**
7.4.1 Simulation Parameters and Assumptions 123
7.4.2 Performance Metrics 124
7.4.3 Performance Evaluation 125
7.4.4 Gradient Formation and Basic Operation 128
7.4.5 Discussion 130
7.5 Empirical Evaluation on an Ad-Hoc Network **132**
7.5.1 Simulation Parameters and Assumptions 132
7.5.2 Performance Metrics 133
7.5.3 Results 133
7.6 Conclusions **140**

This chapter introduces an improved E-MAC protocol, which we refer to as E-MAC+. The simple extensions to E-MAC, inspired by biology, further reinforce the robustness and adaptability of E-MAC under more complex conditions. We define the concepts used, the protocol additions and present results for merging multi-hop chains and ad-hoc networks. These are also compared to the popular and widely known IEEE

802.11 standard protocol. Some interesting emergent decision making is observed in E-MAC+ and we present our discussion of the results and E-MAC+ operation.

7.1 Motivation

The E-MAC protocol exhibits effective performance in multi hop chain scenarios. Its scalability and robustness at maintaining the performance with increasing numbers of nodes is encouraging. Nevertheless in its basic form E-MAC protocol is susceptible to *lock-up*.

Due to the periodic nature of operation, a situation can arise where two nodes (from different chains or routes) persistently interfere with each others transmissions through using similar $txDelay$ values, continuously incrementing them due to acknowledgement timeout. This is an undesirable behaviour which makes E-MAC unstable under specific situations. In this chapter we propose a mechanism enabling nodes to change the $phase$ of their periodic transmissions. This introduces dispersion between colliding nodes to prevent *lock-up*. An additional mechanism is also introduced to diffuse transmission delay values between the different nodes, thereby equalising transmission rates across different routes/chains without direct communication. Improved co-operation and means of increasing rate of transmission for under-performing nodes is achieved. The two proposed mechanisms complement each other and improve the ability to learn $txDelay$ values that achieve both fair and high throughput operation.

7.1.1 E-MAC Lock-Up Description

E-MAC lock-up can be described by taking a simple case of merging multi-hop chains as shown in Figure 7.1. We have a system comprised of two chains joining at node 2. Reception range is one hop and interference is caused within two hops. Therefore node 3 and node 7 which are on different chains cannot detect each others transmissions but can interfere with each others receptions. Other nodes close to the chain merging point encounter the same situation.

If we look at the case where nodes 3 and 7 suffered many consequent acknowledgement timeouts, both will maintain very low $pSuccess$ value (close to 0). This triggers the reaction of the nodes to increase $txDelay$ more often. Given that both of these nodes have similar or the same $txDelay$ and transmit at the same time, a collision will occur. Therefore no acknowledgement is received and both nodes will continue to increment $txDelay$ values. Hence, both nodes keep transmitting with similar timing and with similar $txDelay$ values.

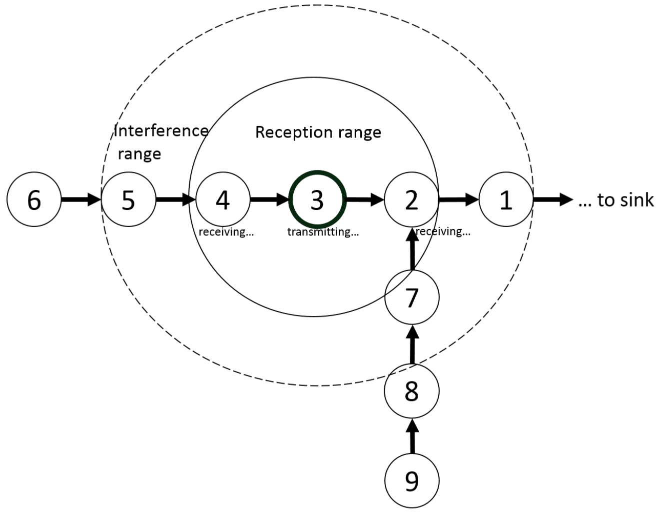


Figure 7.1: E-MAC lock-up example case

7.2 Biological Inspiration

Once again, to achieve our goals in this chapter, we draw inspiration from simple mechanisms in nature. We draw attention to firefly behaviour which provides distributed synchronisation by adjusting phase of flashing [153]. Very simple reaction to the flashes of neighbouring fireflies emerge into large scale synchronisation across millions of entities. By using a similar mechanism we can achieve the reverse process of firefly synchronisation. Our intention is to de-synchronise transmission between repetitively colliding nodes. In addition, we look at the simplest diffusion cases in nature. We believe that, by forming gradients, $txDelay$ can be equalised on different routes that do not communicate with each other, aiding organisation. This section introduces the concepts that we use for enhancing E-MAC: general firefly synchronisation, de-synchronisation and simple diffusion.

7.2.1 Firefly Behaviour and Synchronisation

Firefly synchronised flashing is one of the most exciting views in nature where millions or even billions of fireflies, spread across large distances, manage to flash in perfect synchrony together. It is a powerful example of the complex behaviour a simple distributed multi-agent system can achieve with only very basic rules. Firefly flashing can be explained as a simple charge accumulation over time and release of it once a threshold is reached. Flashes of other fireflies in the surrounding neighbourhood reduce the threshold. A reduced threshold leads to an earlier flash. Threshold reduction is not linear. It is dependent on charge accumulated at the time of observed neighbour flashing. When the threshold is reached, a flash occurs. Charge is used up to flash, the threshold is reset, and the process starts over again.

It is not obvious that this response to flashes and changing threshold can lead to synchronisation. An equivalent engineering concept is pulse-coupled oscillator [153]. If we take an example with two nodes x_1 and x_2 (initialised to a random value between 0 and 1). Both nodes slowly increment their values towards a threshold of 1 in each iteration (accumulation of charge). When the value of 1 is reached, nodes fire (flash) and reset to 0. Given that nodes do not interact, they will simply keep firing at the same period at different phases.

Lets assume nodes can observe each other firing. Once firing is observed, the value is additionally incremented as defined by a Phase Response Curve (PRC):

$$x_t = x_{t-1} + min(\alpha * x(t-1) + \beta, 1) \tag{7.1}$$

$$\alpha = e^{\beta * \epsilon} \tag{7.2}$$

$$\beta = \frac{e^{\beta * \epsilon} - 1}{e^{\beta} - 1} \tag{7.3}$$

where x is value of node and t stands for iteration. α and β define the PRC and given that $\beta > 0$ and $\epsilon > 0$ the nodes would always converge to synchronised firing [154].

An example of the two nodes is shown in Figure 7.2. Both ϵ and β are set to 0.2. The effect of firing can be easily observed. The different response to firing at different values causes two nodes to come into phase after a number of repetitions. This simple process is also scalable and allows millions of simple entities to fire in synchrony.

7.2.2 De-Synchronisation

E-MAC lock-up happens due to repetitive collision and interference. Low $pSuccess$ causes nodes to increment $txDelay$ on every acknowledgement timeout. If there is another node with similar $txDelay$ and low $pSuccess$ within interfering range, the two

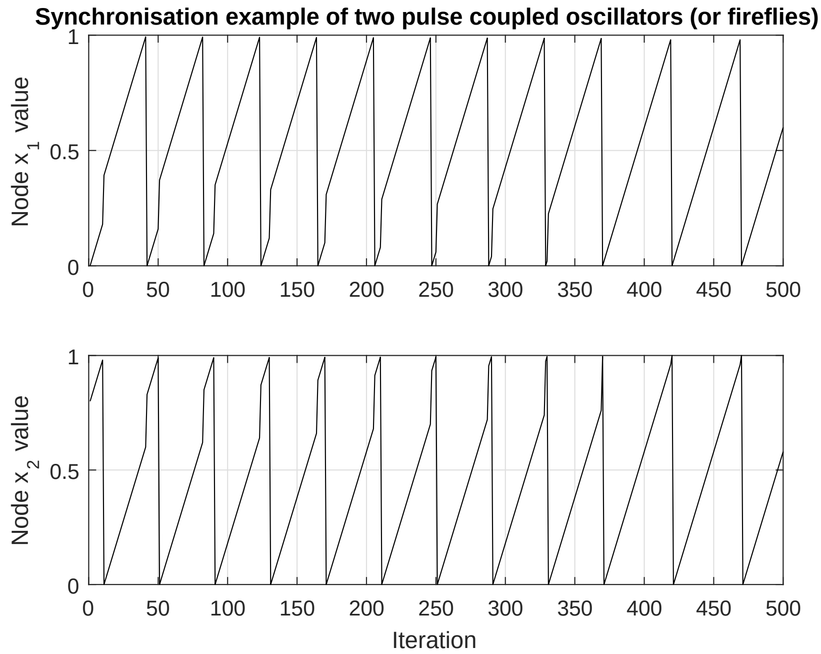


Figure 7.2: Synchronisation example of two pulsed couple oscillators (fireflies)

nodes go into repetitive $txDelay$ incrementation and repetitive interference. Changing the $txDelay$ update logic would cause the system to lose the wanted behaviours it already has. However we can adjust phases of different chain rates to avoid collision. The lock-up would only occur if rates were similar, hence changing phase could improve overall performance, keeping the same rate and making nodes share the medium better.

The firefly flashing synchronisation uses the phase-shift initiated by neighbour flashes to synchronise. A reverse process can be created to de-synchronise these.

Let's take an example of two nodes (the same as the firefly synchronisation example). Lets assume that these nodes do not respond to each other resetting. Instead, nodes trigger a response if they both flash within 15 iterations of each other. This corresponds directly to a packet collision. If we relate the transmission of a packet to a flash and that transmission takes 15 iterations to finish then flashing of two nodes within 15 iterations represents a collision. Packet collision can occur anywhere between the start and the end of the packet. But neither node knows exactly when the collision began. Given this available trigger, a phase can be adjusted. However, there is no knowledge of exact time of flash (collision). Therefore we cannot use an approach with PRC here. The simplest method is to apply a random phase shift within bounds.

If we define the response by Equation 7.4, the behaviour of two nodes gives the results shown in Figure 7.3. After multiple attempts, two nodes go out of phase at about the 320^{th} iteration in this example (after about 6 flashes).

This very simple method can be easily applied to E-MAC to shift the periodic transmissions into different phases when collisions repeat.

$$x_t = x_{t-1} - 0.1 * rand(0, 1) \tag{7.4}$$

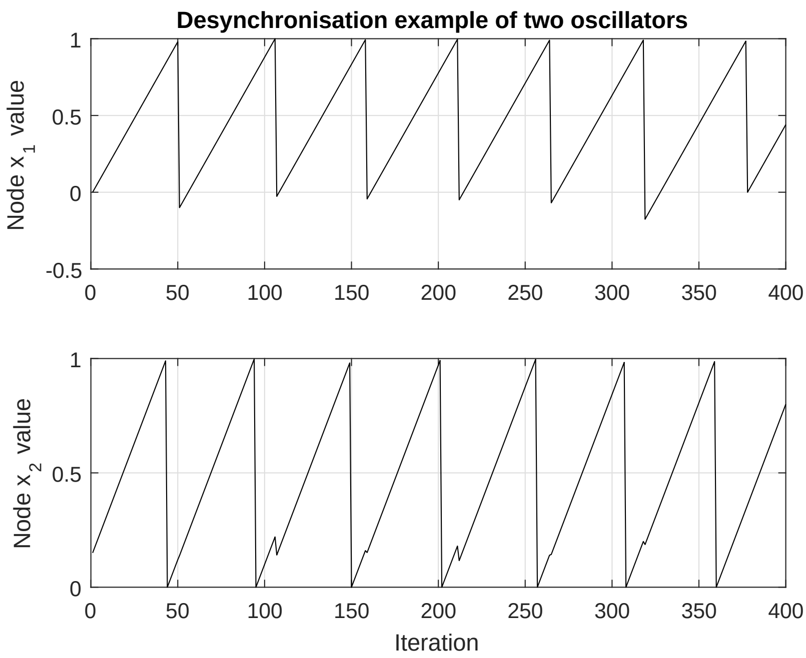


Figure 7.3: Desynchronisation example of two nodes

7.2.3 Diffusion

Diffusion is a natural process where there is a chemical between regions of different concentrations [155]. Given two adjacent regions, a substance in solution in a region of higher concentration will move toward a lower concentration region. This will continue until both regions contain the same concentrations of a substance.

In fact, during the process of diffusion, a continuous concentration gradient will develop [155].

Forming a $txDelay$ gradient in the same way could benefit E-MAC. Firstly, it would eliminate randomness in $txDelay$ values accross the relays and it could carry indirect information down the chain about the state of the network. In addition, in merging chains or multiple route joining cases, such a gradient would serve as a means to propagate information between the two routes. The two routes would essentially aim to maintain similar $txDelay$ values through this phenomenon and could regulate the transmission rates coming in to the junction node.

7.3 E-MAC+

We apply the simple concepts of diffusion and phase-shift, inspired by fireflies, to E-MAC+. The two additional processes improve the behaviour of the algorithm. The key purpose of these is to firstly ensure that nodes avoid lock-up and, in addition, explore different phasings of their periodic transmissions. Secondly the diffusion-like mechanism spreads $txDelay$ values more by trying to mimic neighbours, thereby diffusing a $txDelay$ value through neighbourhood and allowing indirect communication between otherwise distant nodes. In fact, such a method forces different routes to explore similar $txDelay$ values, thereby reinforcing fairness. Diffusion also enforces similarity of $txDelay$ values in nearby nodes, which can cause more repetitive collisions. Without phase-shift, nodes within interference range collide more often. Therefore the two processes complement each other.

7.3.1 Protocol Design

In addition to the basic E-MAC protocol we define two more processes. These are phase-shift and $txDelay$ diffusion, forming the new E-MAC+ protocol. This is the only addition we make. In fact, the original processes within E-MAC remain unchanged.

Algorithm 4: Phase-shift logic for Enhanced E-MAC

while *network running* **do**
    // ack failure is considered when acknowledgement is not received during the timeout period
    if *ack failed* **then**
        // Extend currently engaged transmission delay by a fraction of $txDelay$ using $shiftFrac$
        R = generate random number between 0 and 1
        $Expiry\_time = Expiry\_time + packet\_duration * txDelay * shiftFrac * R$;
    else
    end
end

Algorithm 5: Difussion logic for E-MAC+

```
while network running do
    // Neighbour information is extracted from every packet and
       ack that was addressed to the node
    if Neighbour information received then
        R = generate random number between 0 and 1
        if R > pSuccess then
            if neighbour txDelay > txDelay then
                tx_delay = tx_delay + change_scale
            else
                tx_delay = tx_delay − change_scale
            end
        end
    end
end
```

To provide a phase-shift of periodic $txDelay$ transmissions we introduced the logic shown in Algorithm 4. This follows the same procedure as explained in Section 7.2.2. Each time an acknowledgement fails, the current transmission delay is extended by a small randomly chosen amount. The extension takes a value within the defined interval between zero and $shiftFrac * txDelay$. Results of the $shiftFrac$ effect will be shown in later result sections. To obtain the time in seconds to delay, we need to multiply by the packet duration as $txDelay$ represents a normalised value.

To obtain diffusion behaviour, we use the same principles of response as in the original algorithm. Greater $pSuccess$ inhibits less corrective action and vice versa. This ensures that nodes only act based on stimuli and are able to settle. Every time new neighbour information is received there is a chance (based on $pSuccess$) to perform the diffusion action. Algorithm 5 shows the behaviour. A small amount of information in the form of the $txDelay$ value is shared with each transmission. Nodes try to mimic the neighbours by bringing their $txDelay$ value closer to the neighbour value in increments of $changeScale$ as in the original algorithm. Nodes with neighbours on both sides will be affected by both sets of received acknowledgements and packets, hence forming a gradient. In addition, the source nodes have a tendency to increase the $txDelay$ and regulate the flow, whereas relays exhibit fewer failures and keep $txDelay$ lower. This should logically form a gradient through the route with the highest value starting at the source. Therefore we would have two clear actions on a route from source and sink. Sources cause the route nodes to increase transmission delays, whereas nodes close to sink do the opposite. Just as in many biological systems, we have positive and negative forces adjusting the system.

7.4 Empirical Evaluation on a Merging Chain

To verify the benefits of these additional processes in the E-MAC+ we firstly look at the simple case of a chain-based network with two routes merging together. This presents a more complex problem than a linear multi hop chain. It provides an easy to understand framework when analysing protocol operation before deployment of ad-hoc networks. Essentially it is a simplified case of multiple routes from different sources to the sink.

7.4.1 Simulation Parameters and Assumptions

The Scenario

We evaluate protocols on a merging chain comprised of 16 nodes. The principles and example of the merging chain are explained in Section 3.1.2. Identical nodes have simple radio capabilities. There are two routes, each comprising 12 nodes. The merging point is in the middle and provides 5 hops between the merging node and sources. Such distance complicates organisation as interference from different sources cannot be picked up by merging node and visa versa. The topology of this scenario is shown in Figure 7.4 for clarity.

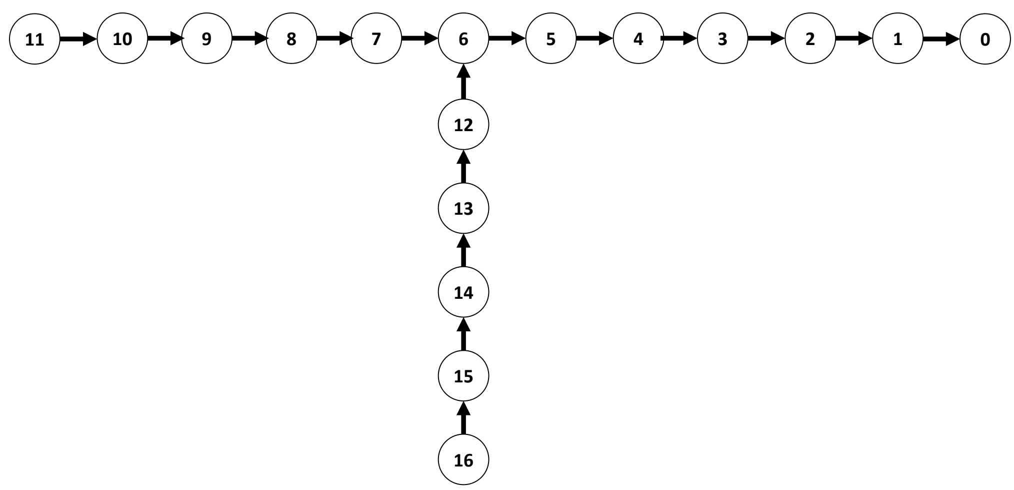


Figure 7.4: Merging chain scenario

Propagation and Radio

A traditional hop based model is used for the communications. Nodes are able to transmit their data over 1 hop (200 meters), but interference is experienced over 2 hops (400 meters). This hop based model is also explained in Section 3.1.6. Successful packet reception only occurs when a channel is clear of other transmissions and free of interference. Propagation delay is not assumed but is calculated based on the distance between nodes. Switching a node to transmission mode when the receiver is locked onto reception of an incoming packet is not allowed. Only one channel is available for both transmission and reception. No synchronisation is available.

Traffic

As in the majority of results presented in this thesis, we use saturated traffic to simulate packet generation. This shows us system stability under difficult conditions. More detail of saturated traffic is presented in Section 3.1.7.

Comparison Schemes

We use the IEEE 802.11 CSMA/CA with RTS/CTS standard, as in previous chapters, to compare our protocol performance. More details of the scheme can be found in Section 3.3.

7.4.2 Performance Metrics

The metrics used in this section are summarised in Table 7.1. More details of each can be found in Section 3.2.1.

Table 7.1: E-MAC+ merging chain scenario performance metrics

Metric	Unit	Reason
Throughput	Erlang	Evaluates the data transfer capabilities
Fairness	Jain's fairness index	Evaluates the ability to share capacity between different nodes.
End-to-End delay	seconds	Evaluates the time it takes for packet to reach the sink node.

7.4.3 Performance Evaluation

Figures 7.5 and 7.6 show the throughput performance of both E-MAC+ and CSMA protocols. Figure 7.5 shows average throughput over the whole simulation. The averaging here takes into account the starting phase of the system. During this phase, E-MAC+ is searching for solution, so throughput is lower. Over-time E-MAC+ adapts to the environment and achieves significantly better throughput than CSMA. Figure 7.6 show this better throughput, as results are based on the last 50 received packets towards the end of the simulation.

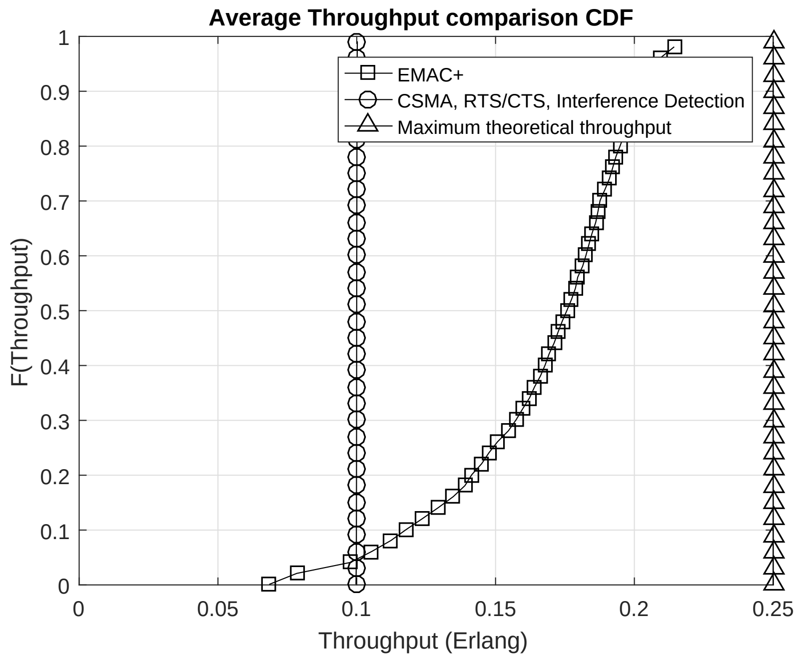


Figure 7.5: Average Throughput CDF

Figure 7.7 shows fairness of the two protocols in this scenario with 2 sources only. About 5% of E-MAC+ simulations exhibit slightly lower than ideal fairness whereas CSMA maintains it in this case. Even though fairness of CSMA shown here is better - the overall number of packets received from individual sources is higher for E-MAC+ due to a much higher throughput. CSMA in such a case, with 2 sources, performs very well in terms of fairness. Both sources are the same distance away from the sink node and the merging node, so neither of them can easily dominate over the other. However, in any case where the source node is closer to the sink than this, the fairness performance for CSMA drops significantly. This is shown in Figure 7.8, where node 6 (Figure 7.4) is also a source. CSMA employs RTS/CTS and carrier sense detection to avoid collisions and loss of packets in the neighbourhood. However these techniques

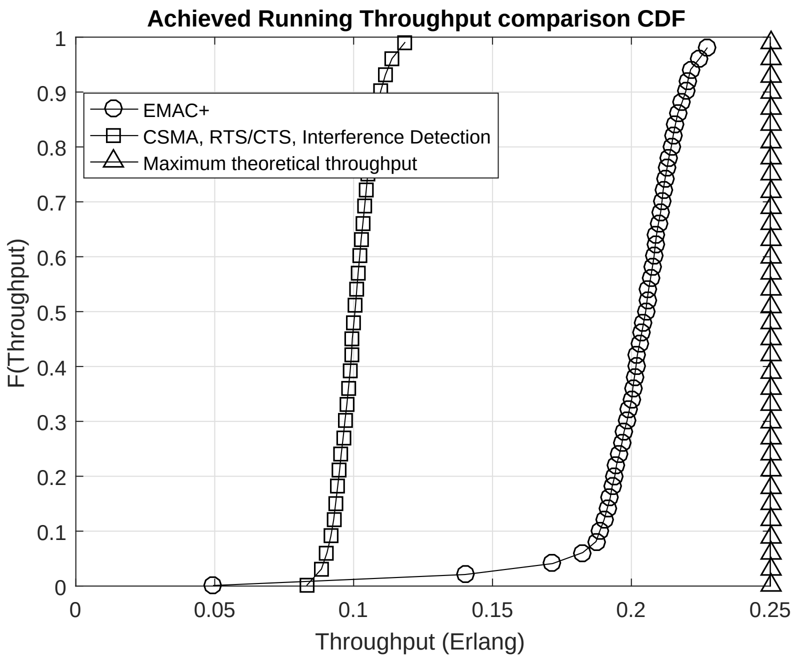


Figure 7.6: Achieved Running Throughput CDF

do not offer benefits along the longer routes. E-MAC+ on the other hand uses the gradients and adjustment of rates. Even though information is shared locally, the gradient disperses along the chain affecting sources further away.

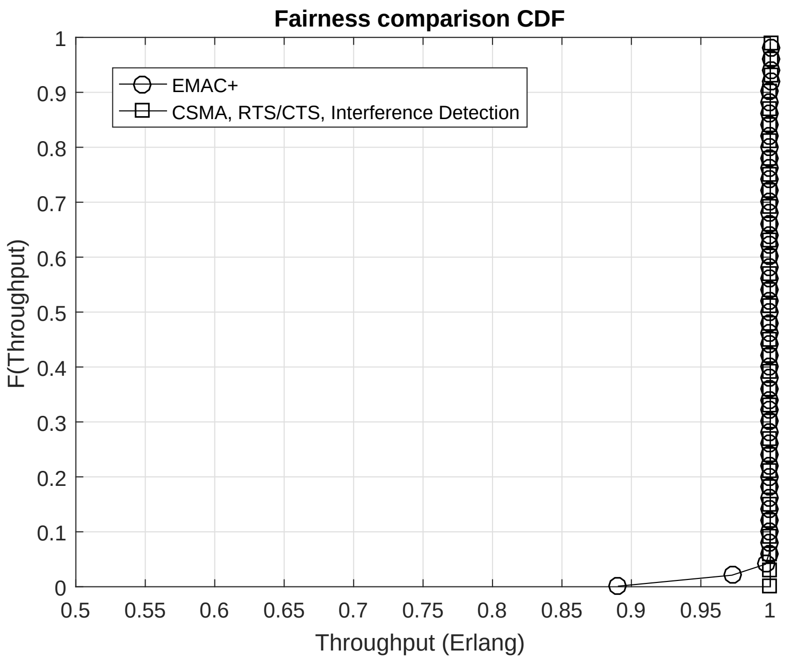


Figure 7.7: Fairness CDF

Figures 7.9 and 7.10 show the end-to-end delay results for the two schemes. We can see that E-MAC+ is significantly better than CSMA in both average and settled running

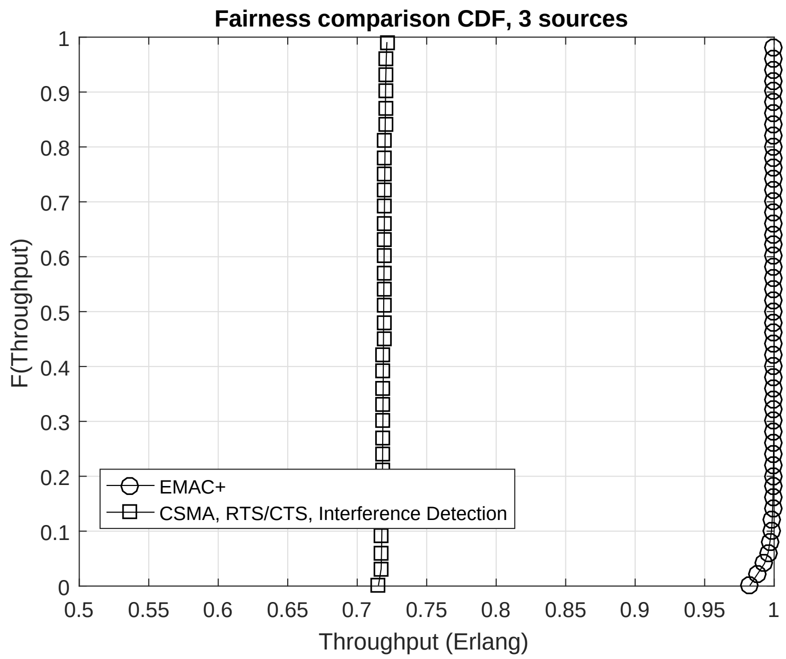


Figure 7.8: Fairness CDF

end-to-end delays. The operation of E-MAC+ ensures that, once a packet is generated, it has almost a non-stop path towards the sink with no collisions. When the protocol settles, the two (or more) sources indirectly synchronise such that packets travelling across chain do not interfere each other. Once a packet is sent forward, a new packet is generated based on the saturated traffic model. This packet now has to wait for $txDelay$ to expire therefore adding up to delay. This is also the cause for the change of steepness of the curves seen in the results of Figure 7.9 and 7.10. Several nodes along the route, close to the source, can settle with same delay as the source (but slightly out of phase). Therefore they will hold up packets for a set period. This prevents packets from different sources colliding at the merging node. In fact, this is an emergent feature of the system. The primarily expected behaviour was that two sources would settle with similar $txDelay$ but out of phase to avoid collisions. The sources are the main contributors towards collision reduction. However in many cases it is the combination of a source being on a different phase as well as relays on a path to the merging node that hold up the packet (helping the source) to avoid collision further down the chain. This arises from the interactions between the nodes and shows simple collective intelligence. To clarify the results - in this case the steep portion of the E-MAC+ curve at 0.13-0.14s in Figure 7.10 shows one node holding up the packet, where the last steep portion of the curve (delay more than 0.15s) present cases where two nodes are holding up the packet.

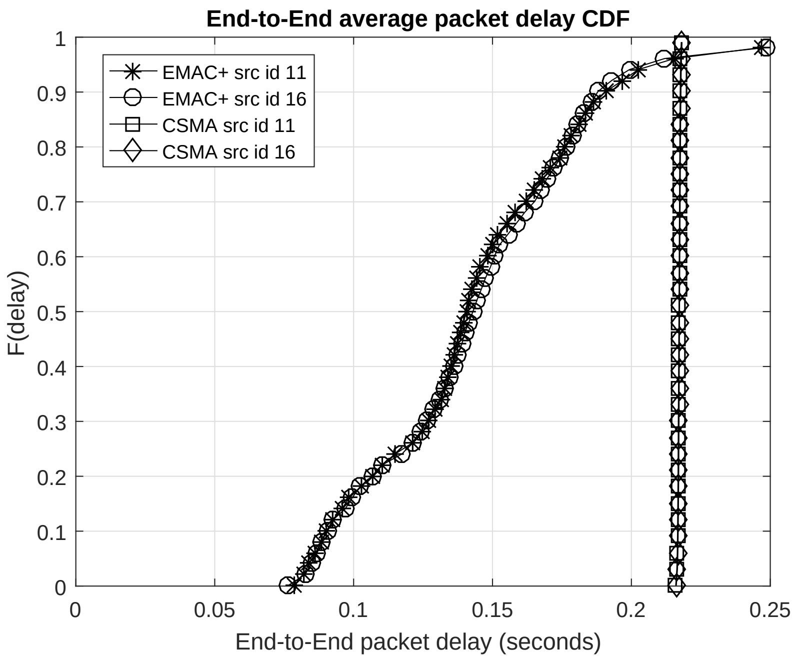


Figure 7.9: Average End-to-End packet delay CDF

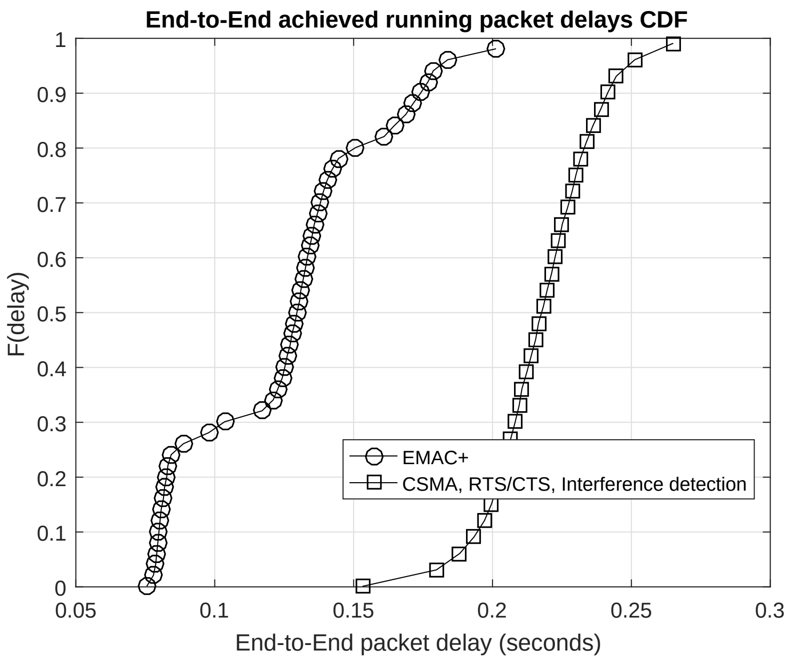


Figure 7.10: Achieved Running End-to-End packet delay CDF

7.4.4 Gradient Formation and Basic Operation

This section summarises the system running performance and shows how nodes adapt $txDelay$ values.

Firstly, we show one of the solutions found by E-MAC+ ($txDelay$ values at each node). These are generally similar across many simulations, but we choose to show this one as it explains several features at once. Figure 7.11 shows such results. We can clearly observe a small increase in $txDelay$ over nodes 1-6 which do not experience many collisions. However immediately after node 6 $txDelay$ rises dramatically. Routes with nodes 7-11 and 12-16 merge at node 6 but do not communicate with each other. Nevertheless we can see similar gradients on both routes formed with source nodes 11 and 16 settling on same $txDelay$. This is the effect of using the diffusion logic in Algorithm 5. Also, on the route involving nodes 1-11, we can observe that nodes 9, 10, 11 all have the same $txDelay$ value. This is the case where relay nodes hold up the packets travelling down to prevent collisions on the merging point as shown by the change of the steepness of the curves in Figure 7.10.

Figure 7.12 shows how $txDelay$ values change throughout time on both sources. In this specific case, node 11 becomes a source at the beginning of simulation and node 16 becomes a source 5 seconds from the start of the simulation. We can immediately see that, once a second source becomes active, the $txDelay$ of both sources increases significantly. Once it crosses a value of 8, it starts to slow down and finally settle to a value. Ideal values of $txDelay$ would be around 8.1. This is the minimum possible, given nodes are shifted in phase from each other by exactly half of this $txDelay$ value. Also, once a value higher than 8 is reached, the system starts to exhibit significantly less packet loss. This reduces the stimulus for increasing $txDelay$, hence the reduced slope in the results. Then the process of diffusion and adjusting transmission delay becomes more active, followed by phase-shift. A solution that offers the best probability of success at each source is found independently. Even though the system is aiming for the highest probability of success, the diffusion keeps pushing the system towards faster rates as nodes in front convey their perspective of a less busy channel through diffusion.

Figures 7.13 and 7.14 show the throughput results over time. We can observe an immediate drop in throughput when a second source is introduced. This is experienced by both E-MAC+ and CSMA. After the second source comes in, E-MAC+ starts adapting, therefore experiencing more collisions and lower throughput. As E-MAC+ source nodes are searching for good values, running throughput is slowly increasing (at 20-80 seconds in Figure 7.14). Then the system immediately "clicks" in position once two routes settle at similar rate and out of phase. At this point packets are travelling to the sink with no interruptions other than nodes delaying their transmissions. Collisions are being avoided both at merging point and throughout relays.

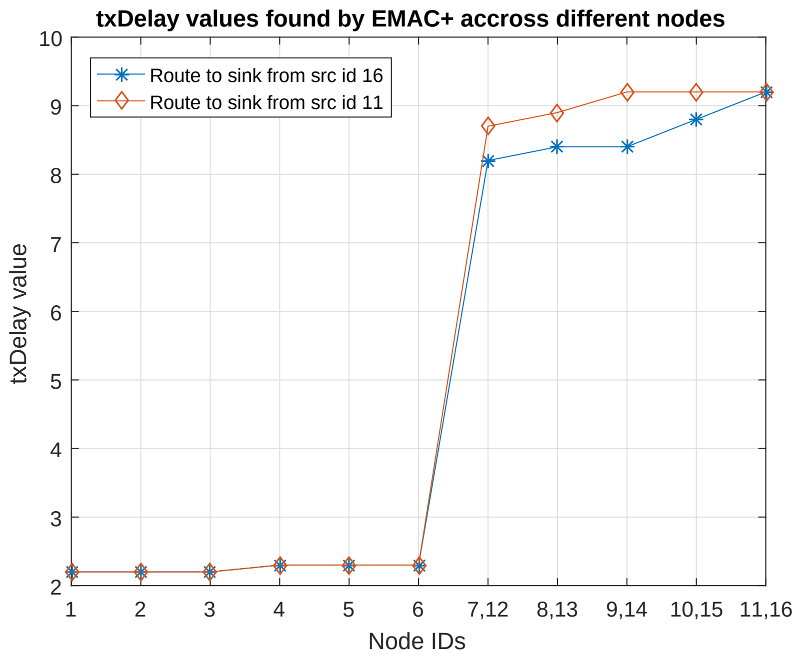


Figure 7.11: $txDelay$ value gradients accross routes

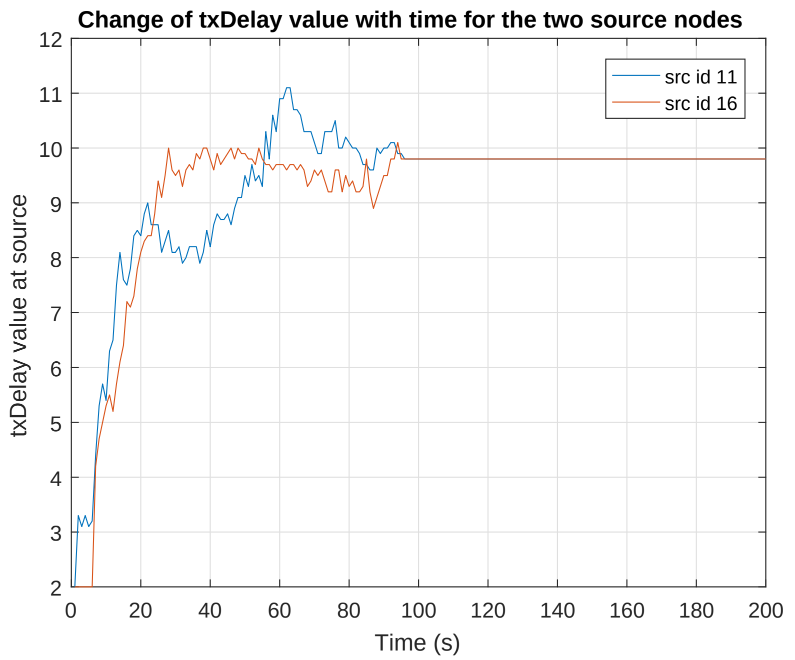


Figure 7.12: Change of $txDelay$ value over time

7.4.5 Discussion

The E-MAC+ algorithm shows some significant emergent properties. In addition to the original E-MAC protocol abilities, we observe cooperative organisation through the addition of diffusion and phase shift. This cooperation gives rise to features that we do not program into the algorithm, such as equalising rates between independent

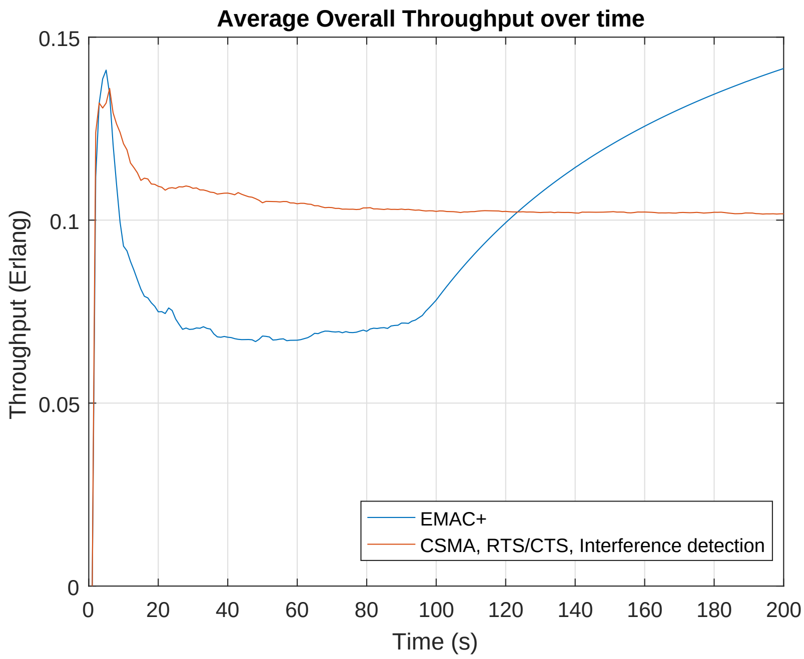


Figure 7.13: Average Overall Throughput over time

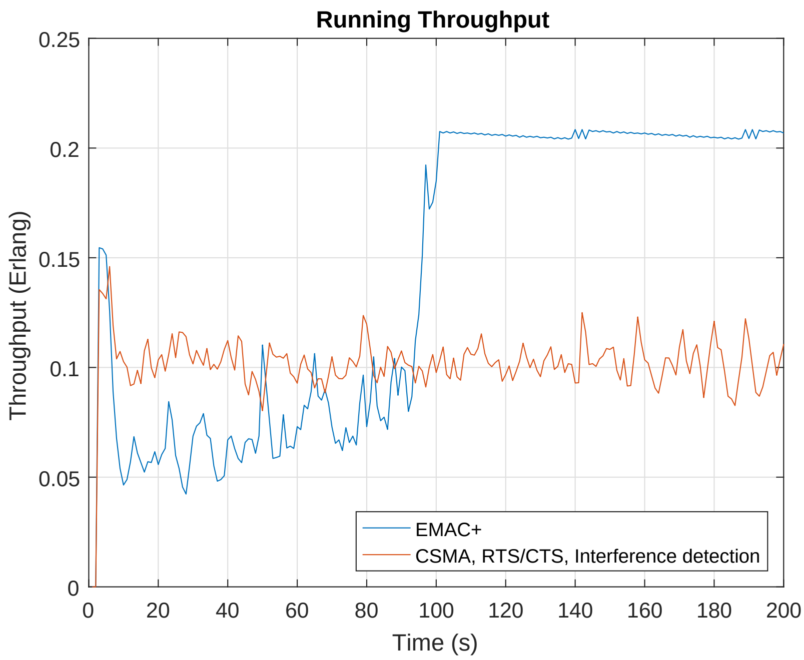


Figure 7.14: Running Throughput

sources on different routes, making relay nodes delay packets to avoid colliding with packets coming from other route and a push to try faster rates.

The protocol shares information of each transmission: the queue size and $txDelay$. However, this data can readily be stored within 20 bits and is minimal. Different processes in the logic operate separately. These processes self-organise within themselves, becoming less or more active during operation. This individual node behaviour then leads to the organisation of the whole system enabling good performance. The value of $txDelay$ at each node can also aid upper layers to decide how much data can be passed through.

7.5 Empirical Evaluation on an Ad-Hoc Network

In this section we look at the performance of the protocol under a more realistic scenario. This scenario presents real node deployment more closely with varying density of nodes and distances between them throughout the network. The evaluation shows how E-MAC+ operates under such complex conditions. The scenario has a complex structure and a large number of nodes operating at once. Under simple conditions E-MAC+ shows complete settling. In such a large scenario (as well as in real-world applications) settling is not important. E-MAC+ is a protocol that adapts to the environment and does not imply one single persistent solution.

7.5.1 Simulation Parameters and Assumptions

The Scenario

100 nodes are deployed (under a uniform distribution) in a $4km^2$ area. The positions are point-processed after random generation such that no two nodes are closer to each other than 150 meters. This creates a realistic deployment case where nodes are not just scattered randomly, but instead are approximately placed into positions. An example of such a map is shown in Figure 3.5 in Section 3.1.5.

Propagation and Radio

A traditional hop based model is used for the communications (as discussed in Section 3.1.6). To have the required coverage over the map and to form routes, the nodes are able to transmit successfully over 400 meters. Interference is experienced over 800 meters. Successful packet reception as before only occurs when a channel is free. Propagation delay is calculated based on distance. Only one channel is used for both transmission and reception. No synchronisation is available.

Traffic

Saturated Traffic is used for generating packets. More details are presented in Section 3.1.7.

Comparison Schemes

The IEEE 802.11 CSMA/CA with RTS/CTS scheme, as before, is used for comparison due to its popularity, good performance and simplicity. More details are presented in Section 3.3.

7.5.2 Performance Metrics

The metrics used in this section are summarised in Table 7.2. More details on each can be found in Section 3.2.1.

Table 7.2: E-MAC+ ad-hoc scenario performance metrics

Metric	Unit	Reason
Throughput per node	Erlang	Evaluates the data transfer capabilities
Fairness	Jain's fairness index	Evaluates the ability to share capacity between different nodes.
End-to-End delay	seconds	Evaluates the time it takes for packet to reach the sink node.

We plot most of the results using box plots. Boxplots are explained in Section 3.2.2. The whiskers on the boxplots represent the 5th and 95th percentile - this allows for accurate and uncluttered statistical representation of the data.

7.5.3 Results

In order to obtain an accurate representation of performance we run the simulations over different numbers of sources in the network. This allows evaluation of protocols under varying conditions. As the number of sources increases it gets more and more challenging to avoid interference and collisions. Keeping high fairness and throughput also becomes significantly more difficult.

Figures 7.15 and 7.16 show the throughput per node boxplots for E-MAC+ and CSMA. The boxplots are plotted after taking 100 maps and combining throughput from each source. We use two different interference detection ranges for CSMA to observe the tradeoffs. We denote the one hop equivalent detection (carrier sensing) range as '1hd' and two hop detection range as '2hd' to simplify the labels. Both figures present the same data, but Figure 7.16 shows a zoomed in view for clarity.

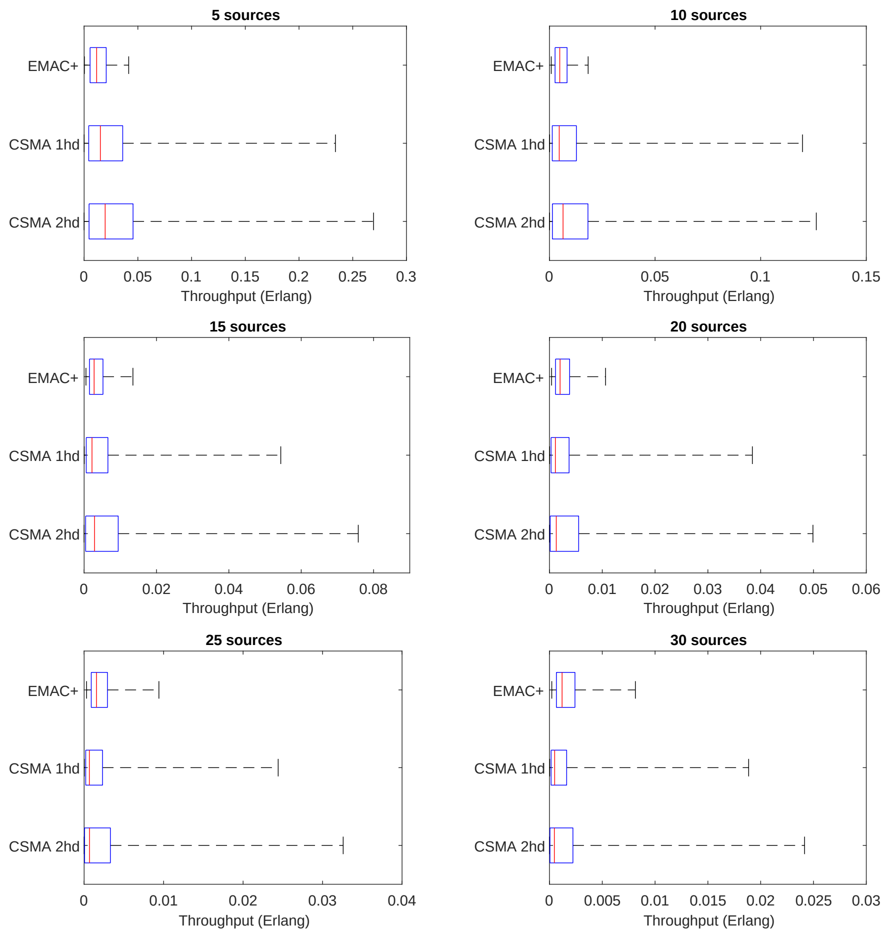


Figure 7.15: Throughput per node on ad-hoc network give different number of sources (full-size)

Firstly, the results indicate a clear difference in consistency between E-MAC+ and CSMA. E-MAC+ has much less variable results and is therefore more predictable. The reader might notice a longer whisker towards higher throughput for CSMA. However, this is not actually better performance. As the results in data represent throughput per node, the longer whiskers are actually due to lower fairness. Some nodes achieve abnormally high throughput by blocking others. As a general trend we observe that CSMA performs very well (even though not consistently) for low numbers of sources in the network. When the number of sources increases to 10 and beyond E-MAC+ provides better median performance. This is also shown by Figure 7.17, which rep-

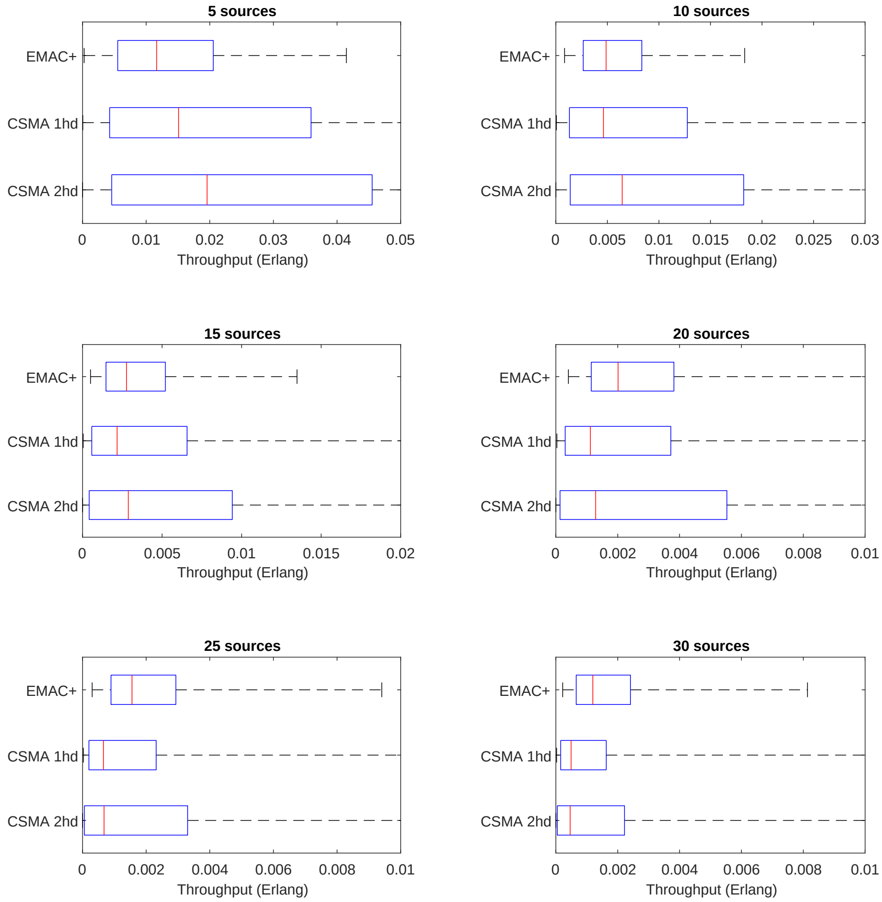


Figure 7.16: Throughput per node on ad-hoc network given different number of sources (zoomed view)

resents only the median values. E-MAC+ noticeably outperforms CSMA under more difficult conditions. In addition, E-MAC+ remains consistent across the range of different conditions. In order to reconcile the throughput results, we also need to look at the fairness achieved by both protocols.

Figures 7.18 and 7.19 show the fairness results as box plots and median curves respectively. These results clearly indicate the significant benefit of E-MAC+. In all the cases E-MAC+ outperforms the CSMA variants. These results also explain the long tails of the throughput results for CSMA. Due to unfair operation, CSMA exhibits domination by some nodes. These nodes provide high throughput at the expense of other nodes. In addition, CSMA results in Figure 7.16 show the tendency for blocked sources by having the distributions tailed closely towards 0 Erlangs for throughput per node. This means that some of the nodes are barely getting any packets through

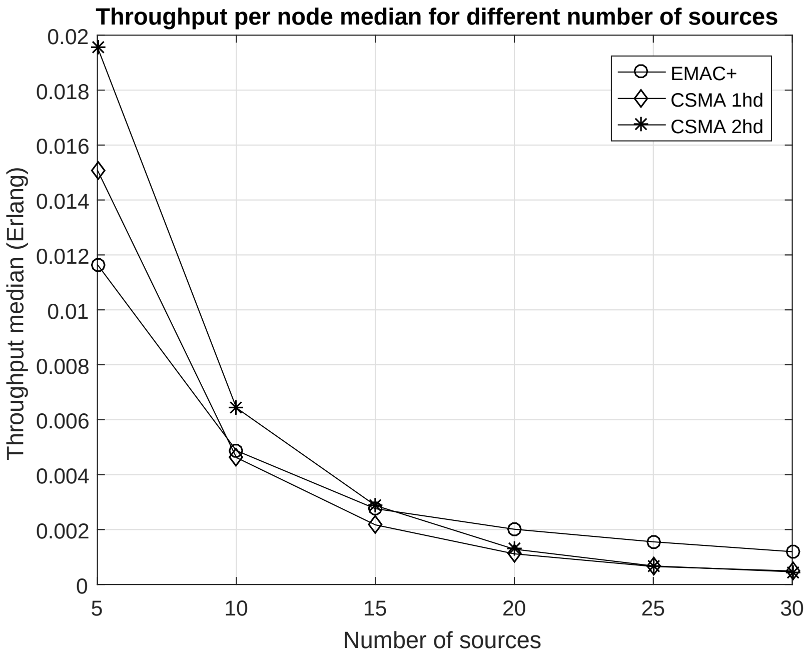


Figure 7.17: Throughput per node on ad-hoc network median comparison for different number of sources

the network.

Significantly lower fairness of CSMA makes it difficult to compare end-to-end packet delays as some sources are completely blocked out and there is significant difference in throughputs from different nodes. Nevertheless we show such comparison in Figure 7.20. It is clear that EMAC+ outperforms CSMA. The reduction in end-to-end delay for the CSMA 2hd case for more sources can be explained by the significantly lower fairness. Large numbers of packets are coming from nearby sources that exhibit lower delay therefore improving the overall mean results. Further sources, however, are completely blocked out. CSMA 1hd fails to deliver good mean end-to-end packet delay overall and is significantly worse than both EMAC+ and CSMA 2hd.

There is a trade-off between the two CSMA variants as well. The two hop detection range (CSMA 2hd) seem to offer higher median throughput when the number of sources in the network is low. However one hop detection (CSMA 1hd) exhibits slightly higher fairness and slightly better throughput results for higher numbers of sources. In all cases, E-MAC+ maintains good performance for both two metrics.

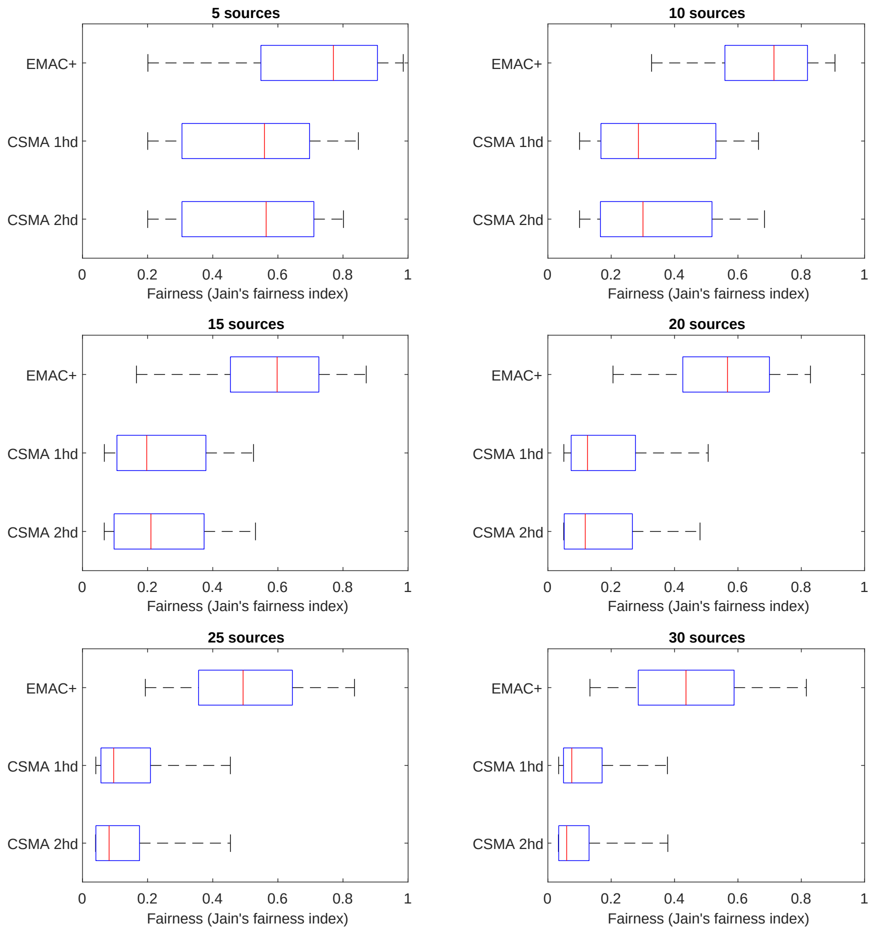


Figure 7.18: Fairness comparison on ad-hoc network for different number of source

Parameter consideration

E-MAC+ contains one additional parameter - $shiftFrac$. We explore the effects and sensitivity of these parameters through simulation.

Figures 7.21 and 7.22 show the sweep results of $shiftFrac$ for E-MAC+ for throughput per node and fairness respectively. Generally we can see that the parameter is not particularly sensitive to $shiftFrac$. Values 0.1 and 0.2 for $shiftFrac$ are observed to have marginally higher fairness on less crowded network. In addition, a lower value of $shiftFrac$ seems to offer a very small improvement in consistency for throughput results. This also matches with a very small improvement of fairness using smaller values of $shiftFrac$.

These parameter effects can also be explained through behaviour of E-MAC+. High values of $shiftFrac$ cause more disturbance (phase-shift) and additional delays at

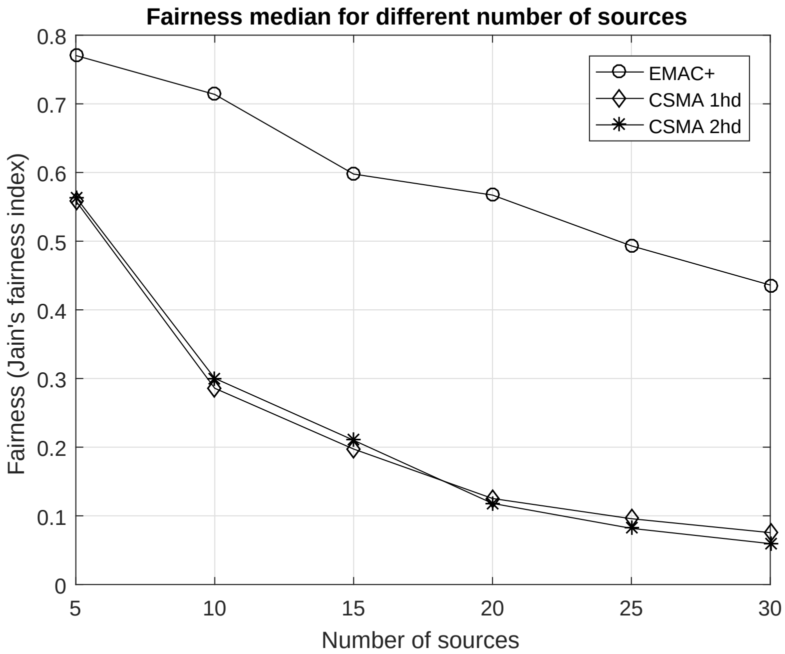


Figure 7.19: Fairness median comparison for different number of sources

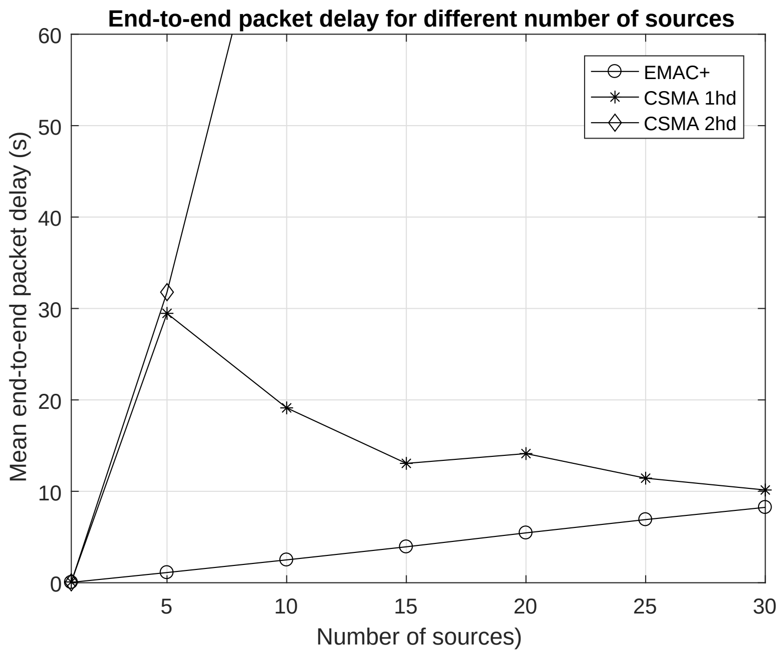


Figure 7.20: End-to-end packet delay comparison for different number of sources

nodes. This can disturb the flow of packets through the routes. Therefore, a slight decrease in performance can be observed. Smaller values of $shiftFrac$ disturb the network less, therefore maintaining slightly better performance. A smaller $shiftFrac$ also tends to offer a finer search for the better phase-shifts, making it easier to achieve when more nodes are present.

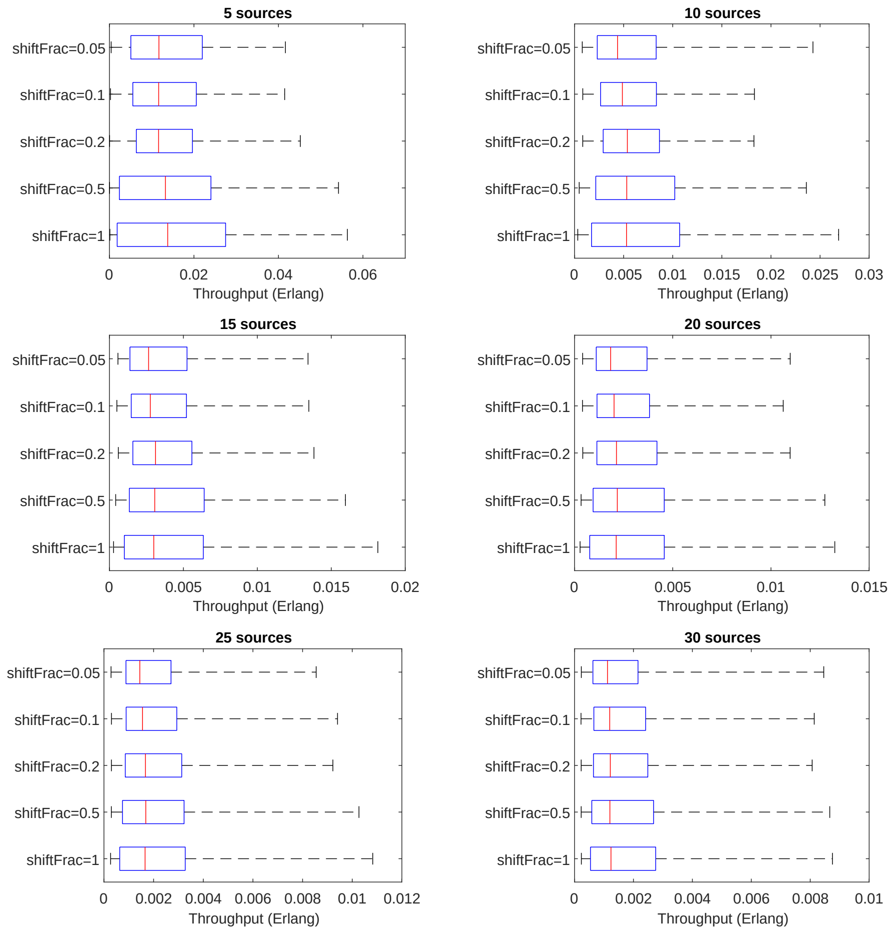


Figure 7.21: Throughput per node for E-MAC+ for different $shiftFrac$ values given different number of sources

Discussion

Overall we can see the significant fairness benefits that E-MAC+ offers for ad-hoc networks. It clearly outperforms CSMA. Given there are only a few sources in the network CSMA can perform well. However even under a less crowded network conditions, CSMA has issues with fairness. E-MAC+, on the other hand, maintains equally good performance throughout over varying number of sources and across different maps. It shows much fairer operation and consistently good throughput when compared to CSMA. Please note that E-MAC+, just like CSMA, does not have any global control and is a fully distributed protocol with no synchronisation.

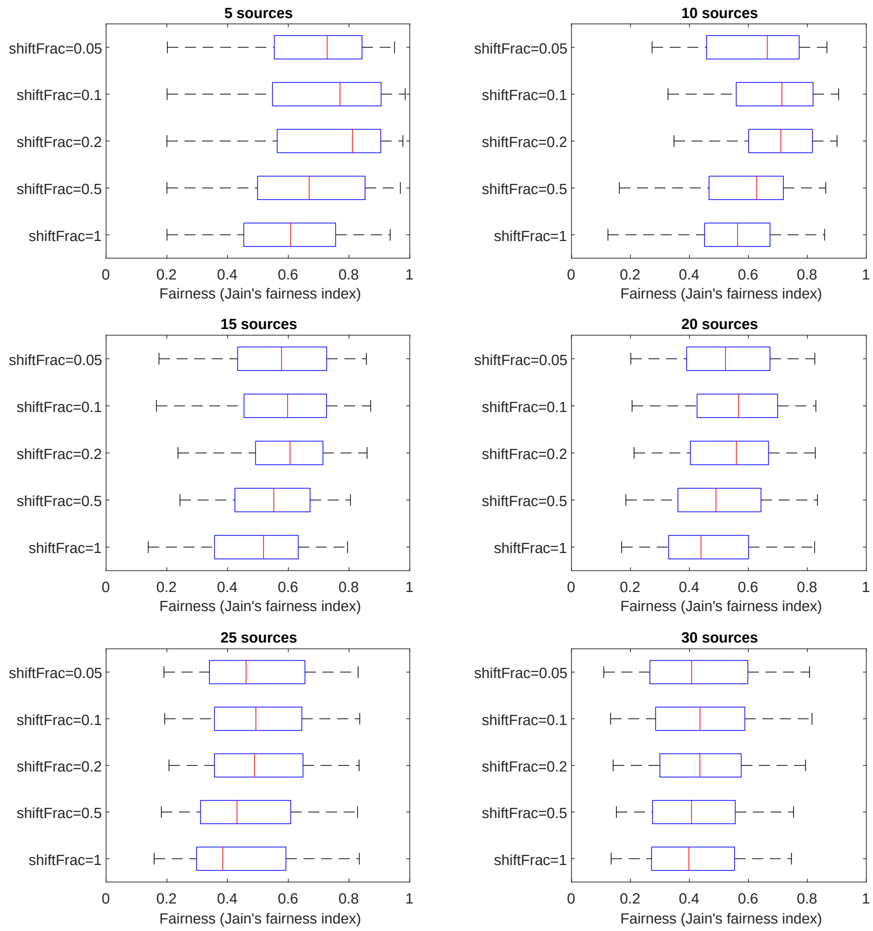


Figure 7.22: Fairness results for E-MAC+ for different $shiftFrac$ values given different number of sources

7.6 Conclusions

In this chapter we presented E-MAC+. Simple biological concepts were used to enhance E-MAC and make E-MAC+ possible. These concepts give rise to additional emergent properties, which enable nodes to cooperate even better together. The addition of phase-shift and diffusion-like processes forms gradients of $txDelay$ over the network. Not only does it regulate the flow down the route but also on multiple individual routes that join together. As shown by the results, good throughput with reliable and fair operation is achieved even under difficult conditions. Empirical evaluation shows that E-MAC+, a simple protocol, outperforms the widely accepted IEEE 802.11 CSMA/CA RTS/CTS scheme. E-MAC+ exhibits good throughput and excellent fairness while maintaining the performance across variety of conditions. The results also demonstrate desirable emergent behaviour arising, which aids in organisation of the network and reduction of collisions.

Chapter 8

Conclusions and Further Work

Contents

8.1 Conclusions **142**
8.1.1 Original Contributions 143
8.1.2 Hypothesis 145
8.2 Recommendations for Further Work **146**

8.1 Conclusions

Wireless Sensor Networks (WSNs) present a challenging environment for communications. Increasing and changing numbers of nodes in the network, that are simple and cheap, call for unconventional methods to facilitate good performance. While conventional methods perform satisfactorily in these conditions, scalability soon becomes an issue. Larger-scale WSNs start to under perform significantly in some important performance metrics. There are state-of-art schemes that do solve some problems, presenting near optimal solutions. However, these often need fine tuning and do not operate effectively if synchronisation is lost. In other cases some form of additional feedback is required to maintain high performance.

Inspired by nature and biology, we look at the WSN as a swarm: large numbers of simple entities communicating with each other to reach some higher goal. They are able to exhibit sophisticated, collective behaviours without central control. This emergence arises from the simplest interactions between individuals and their environment. Inspired by this emergence and simplicity, we research algorithms inspired by swarms and also the biological processes themselves. We then apply these methods to the MAC layer of a WSN to solve scheduling and contention issues. In order to apply these methods we firstly propose a way of incorporating these via a simple control engineering theory based on negative feedback action on the system controlling transmission delay.

In Chapters 1 and 2 we introduced the thesis and present a literature review. Chapter 4 summarises the simulation methodologies and the techniques used to evaluate the performance of our proposed schemes. Chapter 4 introduces some early research and presents a preliminary approach before applying bio-inspired methods. This chapter presents exploitation of the similarity between MAC concepts and control engineering theory. We propose a new back-off mechanism which we call transmission delay. This mechanism is engaged after every single transmission. The value (the length) of a persistent backoff is varied via a negative feedback loop (through packet acknowledgements). To prove the concept we also apply an offline optimiser to a chained networks. Chapter 5 follows on and presents a modification of a Particle Swarm Optimiser (PSO) to work on-line for WSNs. While showing good performance, we realise that the complexity could be reduced and some tuning parameters removed. Chapter 6 presents the outcome of this. Instead of looking into algorithms inspired by biology, we looked directly into concepts found in nature. The Emergence MAC (E-MAC) protocol is the outcome. Inspired by insect colony behaviour, it demonstrates useful emergent behaviours and proves to be self-organising and adaptable without outside intervention. Chapter 7 then follows on to enhance E-MAC and presents E-MAC+ that addresses more complex scenarios and uses additional biological processes to improve cooperation between nodes.

8.1.1 Original Contributions

Swarming Medium Access Control

In Chapter 5 we propose a scheme inspired by the PSO. The scheme uses concepts from PSO and additional mechanisms to adjust $txDelay$ of each node. The protocol is fully distributed and each node operates independently. A small amount of information is shared upon transmission of packets. This allows cooperative behaviour and decision making based on both local and neighbourhood experiences (as found in a PSO). To enable efficient and adaptive operation we introduce taxis into the PSO formulation. This excites the nodes ("particles") to change and adapt the $txDelay$ more quickly. The outcome shows interesting results, where without additional hardware sensing features or additional passed messages to avoid unwanted interference, the protocol achieves comparable or even better results than a comparison scheme with higher complexity.

Applications of insect swarm task allocation and sorting behaviour towards MAC scheduling (E-MAC)

Inspired by biology, we explored mechanism that enables insect swarms to develop highly complex behaviours. The combination of ant task allocation, sorting and the Robbins Monro algorithm allowed us to design an algorithm that controls $txDelay$ in a significantly simpler manner. Nevertheless, results show useful emergent properties. The protocol adapts to deliver significantly better performance than CSMA on chained networks. E-MAC is able to adapt and settle to show good throughput, very low latency and excellent fairness. Without adjustment of the parameters, it offers very high utilisation of available capacity under different communication and interference conditions. The protocol shares minimal information with neighbours.

Theoretical analysis of E-MAC settling time

Using Markov model concepts we create a representation of E-MAC operation. We propose a method to account for operation in real-time via additional state transitions that loop back. This enables it to account for delays in transmission and enables the Markov model to be used to calculate the settling time of the algorithm. Simulation results, when compared to the Markov model, show close similarity without significant error. The error is present due to the assumption that nodes are not dependent on each other. The simulated system shows slightly faster settling time than the theoretical Markov model. This is because of the cooperation activity of the former.

Application of diffusion and de-synchronisation to enhance E-MAC (E-MAC+)

E-MAC relies on acknowledgements as a form of feedback. In some specific cases, interference and collisions can cause lock-up forcing E-MAC to increase $txDelay$ significantly, therefore reducing throughput. In Chapter 7 we look at this issue and propose a method, inspired by firefly behaviour and physical diffusion processes, to improve cooperation between nodes thereby avoiding locking-up. In addition, the improved cooperation shows benefits for long chains due to its ability to indirectly affect different routes that otherwise could not communicate. Empirical evaluation is presented for several cases with merging chain and in ad-hoc scenarios. It shows much better and fairer operation than the comparison scheme. The cooperation enables intelligent control of $txDelay$ over whole chains and routes across the network. Very low latency, good throughput and fairness can be observed. These are emergent properties.

8.1.2 Hypothesis

The hypothesis of this thesis was stated as:

"By combining some concepts taken from control theory and Distributed Artificial Intelligence (DAI), it is possible to achieve performance levels comparable with conventional MAC schemes whilst exhibiting self-organisation with greater robustness, scalability and lower complexity."

The contributions described in Section 8.1.1 are all aimed towards proving the hypothesis. They can be summarised as follows:

- The proposed transmission delay approach provides a simple framework which enables application of a variety of DAI techniques.
- The applied offline PSO algorithm, shows the feasibility of using transmission delay in place of backoff. It proves that settled values across a chain networks can, in fact, provide very good performance with extremely low complexity.
- The proposed Swarming MAC protocol is a distributed algorithm inspired by a PSO and utilises the transmission delay. It operates online, enabling each node to make decisions and adjust transmission delay values based on its own and neighbouring experiences. The performance shown is comparable to or better than the comparison scheme.
- E-MAC is a biologically inspired protocol. Motivated by insect colony behaviours, we designed E-MAC using very simple rules based on the task allocation and the sorting behaviour of insects. This distributed algorithm shows emergent behaviours that enable high throughput, low latency and fair operation over different operating conditions. It operates using very simple rules with no additional hardware requirements, but achieves great performance, adaptability and self-organisation.
- E-MAC+ is an enhanced algorithm, which provides improved cooperation between individual nodes. The emergent robustness and fault-tolerance are bet-

ter than in the case of the original E-MAC. The E-MAC+ algorithm is significantly simpler than its comparison scheme but provides very good performance, fairness and robustness under a variety of scenarios. Self-organisation and reorganisation enable the protocol to adapt to different conditions without the need for tuning.

The stated contributions were empirically evaluated and show that simplicity in implemented DAI can truly outperform conventional MAC methods. The outcome is the emergence of self-organisation, high throughput operation, low end-to-end delay and very high fairness. Even with low complexity, the proposed protocols are able to achieve better scalability and robustness. These protocols are able to operate under a variety of conditions without the need for tuning. The performance also maintains the same qualities with increasing numbers of source nodes in a network. Therefore the stated contributions prove the hypothesis.

8.2 Recommendations for Further Work

There are a number of ideas that stem from this thesis and could be explored further.

Advanced fitness functions and topologies for Swarming Medium Access Control

In Chapter 5 we proposed a Swarming MAC protocol which is inspired by a PSO. Inherently the protocol uses a fitness function to define the behaviour of the nodes once observation of channel occurs. We defined the fitness function simply as an equal combination of throughput and probability of success. There is scope to explore more advanced fitness functions here to improve the behaviour. Fitness functions could include a variety of factors and could be based both on neighbours and self observations. It could potentially significantly improve the system performance as well as improve cooperation between the nodes.

Additionally, different topologies in terms of neighbour data sharing could be explored. It may be beneficial to not only consider the best neighbour, but consider a combination of neighbours when calculating the velocities. This could further enhance cooperation and improve the altruistic behaviour of nodes.

Hardware implementation/Practical evaluation of E-MAC, E-MAC+

While we empirically evaluated E-MAC and E-MAC+ in a variety of scenarios it would be extremely beneficial to implement and run these protocols in real environments. E-MAC, during the empirical evaluation, shows significantly better performance than the more complex comparison scheme - the IEEE 802.11 standard. However simulations assume the absence of many phenomena that hardware exhibits. Hardware implementation is important in progressing the E-MAC and E-MAC+ concept.

Incorporating energy saving control into E-MAC, E-MAC+

Power limitation in many WSN scenarios is a critical requirement. The use of transmission delay opens up a number of ways for applying sleep schedules to nodes. By providing efficient energy management for E-MAC+, its applicability could be significantly enhanced. E-MAC+ shares transmission delay values with its neighbours. Therefore, it is possible to use this information to define a sleeping schedule. While transmission delay is engaged sources could be potentially sleeping. The reception of other transmission delays allows determination of the time when the next reception is likely to occur. This could provide a significant power saving without the need for additional information sharing or complexity requirements.

Enhancing E-MAC+ with routing information, message passing beyond neighbours

E-MAC+ was designed with multiple routes joining in mind. Multiple routes present a challenging task where maintaining fairness and avoiding interference becomes difficult. The lack of information parsing between different routes renders it difficult to avoid. Nevertheless, by application of diffusion and de-synchronisation, E-MAC+ is able to control multiple routes and indirectly pass information in some form through. We believe that considering the routing information at the joining points and sharing it back up to the routes could potentially improve the speed of adaptivity and performance of the system.

Enhancing settling time of E-MAC and E-MAC+

There is scope to look at modifying E-MAC and E-MAC+ protocols to settle quicker. Reinforcement Learning is known to use a technique called WoLF (Win or Learn Fast). By applying different α values in the learning RM algorithm, depending on transmission outcome it is able to speed up the convergence of the learning process. It may be possible to apply similar techniques for calculating the $pSuccess$ value and improve the speed of settling. Potential improvements could also be made to the $change\_scale$ value adaptivity. $change\_scale$ is the value that adjusts transmission delay. On network start-up the system may benefit from having higher $change\_scale$ values to move out of the low performance range. The $change\_scale$ value could be lowered for greater precision when close to settling and therefore could improve overall speed of settling.

Abbreviations

AEA Adaptive Election Algorithm

ACK Acknowledgement

ACO Ant Colony Optimisation

AI Artificial Intelligence

AIS Artificial Immune Systems

AP Access Point

BEB Binary Exponential Backoff

B-MAC Versatile Low Power Medium Access Control

CA Cellular Automata

CDF Cumulative Distribution Function

CDMA Code Division Multiple Access

CSMA Carrier Sense Multiple Access

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance

CSMA/CD Carrier Sense Multiple Access with Collision Detection

DAI Distributed Artificial Intelligence

DI Distributed Intelligence

DNI Distributed Natural Intelligence

DRAND Distributed Randomised TDMA scheduling

DS-MAC Dynamic Sensor Medium Access Control

EA Evolutionary Algorithm

E-MAC Emergence Medium Access Control

E-MAC+ Enhanced Emergence Medium Access Control

FDMA Frequency Division Multiple Access

GA Genetic Algorithm

GP Genetic Programming

ID-MAC Identity-Based Medium Access Control

LAN Local Area Network

LB-MAC Lifetime-Balanced Medium Access Control

LEACH Low-Energy Adaptive Clustering Hierarchy

MAC Medium Access Control

MAS Multi-Agent System

ML Machine Learning

NI Natural Intelligence

NN Neural Network

OFDMA Orthogonal Frequency Division Multiple Access

PCS Physical Carrier Sensing

PEDAMACS Power-Efficient and Delay-Aware Medium Access Control

PRC Phase Response Curve

PRIMA Priority-Based Medium Access Control

PSO Particle Swarm Optimisation

PW-MAC Energy-Efficient Predictive-Wake up Medium Access Control

RL Reinforcement Learning

RM Robbins Monro

RSSI Received Signal Strength Indication

RTS/CTS Request To Send and Clear To Send

QoS Quality of Service

SDMA Space Division Multiple Access

SI Swarm Intelligence

S-MAC Sensor Medium Access Control

TDMA Time Division Multiple Access

TRAMA Traffic-Adaptive Medium Access Control

T-MAC Timeout Medium Access Control

WSN Wireless Sensor Network

X-MAC Short Preamble Medium Access Control

Z-MAC Zebra Medium Access Control

Bibliography

[1] Raghavendra V Kulkarni, Anna Forster, and Ganesh Kumar Venayagamoorthy. Computational intelligence in wireless sensor networks: a survey. *Communications Surveys & Tutorials, IEEE*, 13(1):68–96, 2011.

[2] BBC-Earth. Termite mound, 2015. URL http://ichef.bbci.co.uk/wwfeatures/wm/live/1280_720/images/live/p0/2y/b5/p02yb5r0.jpg. [Online; accessed February 15, 2017].

[3] Because birds (www.becausebirds.com). Flock of starlings, 2014. URL http://becausebirds.com/wp-content/uploads/2014/07/alain-delorme-5.jpg. [Online; accessed February 15, 2017].

[4] Norman Abramson. The throughput of packet broadcasting channels. *IEEE Transactions on Communications*, 25(1):117–128, 1977.

[5] Ian F Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.

[6] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (IoT): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[7] Guillermo Barrenetxea, François Ingelrest, Gunnar Schaefer, and Martin Vetterli. Wireless sensor networks for environmental monitoring: the sensorscope experience. In *IEEE International Zurich Seminar on Communications*, pages 98–101, 2008.

[8] Vehbi C Gungor and Gerhard P Hancke. Industrial wireless sensor networks: challenges, design principles, and technical approaches. *IEEE Transactions on Industrial Electronics*, 56(10):4258–4265, 2009.

[9] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, pages 88–97, 2002.

[10] Meng Shuai, Kunqing Xie, Xiujun Ma, and Guojie Song. An on-road wireless sensor network approach for urban traffic state monitoring. In *11th International IEEE Conference on Intelligent Transportation Systems*, pages 1195–1200, 2008.

[11] Anish Arora, Prabal Dutta, Sandip Bapat, Vinod Kulathumani, Hongwei Zhang, Vinayak Naik, Vineet Mittal, Hui Cao, Murat Demirbas, Mohamed Gouda, et al. A line in the sand: a wireless sensor network for target detection, classification, and tracking. *Computer Networks*, 46(5):605–634, 2004.

[12] Jason Hill, Mike Horton, Ralph Kling, and Lakshman Krishnamurthy. The platforms enabling wireless sensor networks. *Communications of the ACM*, 47(6): 41–46, 2004.

[13] Sameer Tilak, Nael B Abu-Ghazaleh, and Wendi Heinzelman. A taxonomy of wireless micro-sensor network models. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6(2):28–36, 2002.

[14] Cian O Mathuna, Terence O'Donnell, Rafael V Martinez-Catala, James Rohan, and Brendan O'Flynn. Energy scavenging for long-term deployable wireless sensor networks. *Talanta*, 75(3):613–623, 2008.

[15] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. *Computer Networks*, 52(12):2292–2330, 2008.

[16] Chee-Yee Chong and Srikanta P Kumar. Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, 2003.

[17] Sang Hyuk Lee, Soobin Lee, Heecheol Song, and Hwang Soo Lee. Wireless sensor network design for tactical military applications: remote large-scale environments. In *Military Communications Conference*, pages 1–7. IEEE, 2009.

[18] Gyula Simon, Miklós Maróti, Ákos Lédeczi, György Balogh, Branislav Kusy, András Nádas, Gábor Pap, János Sallai, and Ken Frampton. Sensor network-based countersniper system. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pages 1–12. ACM, 2004.

[19] Jay Chang, William Mendyk, Lisa Thier, Paul Yun, Andy LaRow, Scott Shaw, and William Schoenborn. Early attack reaction sensor (EARS), a man-wearable gunshot detection system. In *Defense and Security Symposium*, pages 62011T–62011T. International Society for Optics and Photonics, 2006.

[20] Jer Hayes, Stephen Beirne, K-T Lau, and Dermot Diamond. Evaluation of a low cost wireless chemical sensor network for environmental monitoring. In *Sensors*, pages 530–533. IEEE, 2008.

[21] Tian He, Sudha Krishnamurthy, John A Stankovic, Tarek Abdelzaher, Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu, Jonathan Hui, and Bruce Krogh. Energy-efficient surveillance system using wireless sensor networks. In *Proceedings of the 2nd International Conference on Mobile systems, Applications, and Services*, pages 270–283. ACM, 2004.

[22] Tian He, Sudha Krishnamurthy, Liqian Luo, Ting Yan, Lin Gu, Radu Stoleru, Gang Zhou, Qing Cao, Pascal Vicaire, John A Stankovic, et al. Vigilnet: an integrated sensor network system for energy-efficient surveillance. *ACM Transactions on Sensor Networks*, 2(1):1–38, 2006.

[23] Ning Wang, Naiqian Zhang, and Maohua Wang. Wireless sensors in agriculture and food industry – recent development and future perspective. *Computers and Electronics in Agriculture*, 50(1):1–14, 2006.

[24] Koen Langendoen, Aline Baggio, and Otto Visser. Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture. In *20th International Parallel and Distributed Processing Symposium*, page 8. IEEE, 2006.

[25] Kirk Martinez, Royan Ong, and Jane Hart. Glacsweb: a sensor network for hostile environments. In *First Annual Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, pages 81–87. IEEE, 2004.

[26] Jessica D Lundquist, Daniel R Cayan, and Michael D Dettinger. Meteorology and hydrology in Yosemite national park: a sensor network application. In *Information Processing in Sensor Networks*, pages 518–528. Springer, 2003.

[27] D Choi and D Kim. Wireless fieldbus for networked control systems using LR-WPAN. *International Journal of Control Automation and Systems*, 6(1):119, 2008.

[28] Bin Lu and Vehbi C Gungor. Online and remote motor energy monitoring and fault diagnostics using wireless sensor networks. *IEEE Transactions on Industrial Electronics*, 56(11):4651–4659, 2009.

[29] Jianping Song, Song Han, Aloysius K Mok, Deji Chen, Mike Lucas, and Mark Nixon. WirelessHART: applying wireless technology in real-time industrial process control. In *Real-Time and Embedded Technology and Applications Symposium*, pages 377–386. IEEE, 2008.

[30] M Castillo-Effer, Daniel H Quintela, W Moreno, R Jordan, and W Westhoff. Wireless sensor networks for flash-flood alerting. In *Proceedings of the 5th IEEE International Caracas Conference on Devices, Circuits and Systems*, volume 1, pages 142–146. IEEE, 2004.

[31] Majid Bahrepour, Nirvana Meratnia, Mannes Poel, Zahra Taghikhaki, and Paul JM Havinga. Distributed event detection in wireless sensor networks for disaster management. In *2nd International Conference on Intelligent Networking and Collaborative Systems*, pages 507–512. IEEE, 2010.

[32] Geoffrey Werner-Allen, Jeff Johnson, Mario Ruiz, Jonathan Lees, and Matt Welsh. Monitoring volcanic eruptions with a wireless sensor network. In *Proceeedings of the Second European Workshop on Wireless Sensor Networks*, pages 108–120. IEEE, 2005.

[33] Sukun Kim, Shamim Pakzad, David Culler, James Demmel, Gregory Fenves, Steven Glaser, and Martin Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *6th International Symposium on Information Processing in Sensor Networks*, pages 254–263. IEEE, 2007.

[34] Albert Ko and Henry YK Lau. Robot assisted emergency search and rescue system with a wireless sensor network. *International Journal of Advanced Science and Technology*, 3:69–78, 2009.

[35] Eugene Shih, Seong-Hwan Cho, Nathan Ickes, Rex Min, Amit Sinha, Alice Wang, and Anantha Chandrakasan. Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, pages 272–287. ACM, 2001.

[36] Marcos Augusto M Vieira, Claudionor N Coelho, DC Da Silva, and José Monteiro da Mata. Survey on wireless sensor network devices. In *Proceedings of IEEE Conference on Emerging Technologies and Factory Automation*, volume 1, pages 537–544. IEEE, 2003.

[37] Farhan Simjee and Pai H Chou. Everlast: long-life, supercapacitor-operated wireless sensor node. In *Proceedings of the International Symposium on Low Power Electronics and Design*, pages 197–202. IEEE, 2006.

[38] Yen Kheng Tan, Truc Phuong Huynh, and Zizhen Wang. Smart personal sensor network control for energy saving in DC grid powered LED lighting system. *IEEE Transactions on Smart Grid*, 4(2):669–676, 2013.

[39] Fan Ye, Haiyun Luo, Jerry Cheng, Songwu Lu, and Lixia Zhang. A two-tier data dissemination model for large-scale wireless sensor networks. In *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking*, pages 148–159. ACM, 2002.

[40] Geoffrey Werner-Allen, Konrad Lorincz, Mario Ruiz, Omar Marcillo, Jeff Johnson, Jonathan Lees, and Matt Welsh. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10(2):18–25, 2006.

[41] Chenyang Lu, Brian M Blum, Tarek F Abdelzaher, John A Stankovic, and Tian He. Rap: A real-time communication architecture for large-scale wireless sensor networks. In *Real-Time and Embedded Technology and Applications Symposium, 2002. Proceedings. Eighth IEEE*, pages 55–66. IEEE, 2002.

[42] Antonio-Javier Garcia-Sanchez, Felipe Garcia-Sanchez, and Joan Garcia-Haro. Wireless sensor network deployment for integrating video-surveillance and data-monitoring in precision agriculture over distributed crops. *Computers and Electronics in Agriculture*, 75(2):288–303, 2011.

[43] Gurkan Tuna, Tarik Veli Mumcu, Kayhan Gulez, Vehbi Cagri Gungor, and Hayrettin Erturk. Unmanned aerial vehicle-aided wireless sensor network deployment system for post-disaster monitoring. In *International Conference on Intelligent Computing*, pages 298–305. Springer, 2012.

[44] Ahmad Abed Alhameed Alkhatib and Gurvinder Singh Baicher. Wireless sensor network architecture. In *International Conference on Computer Networks and Communication Systems*, 2012.

[45] Tommaso Melodia, Mehmet C Vuran, and Dario Pompili. The state of the art in cross-layer design for wireless sensor networks. In *Wireless Systems and Network Architectures in Next Generation Internet*, pages 78–92. Springer, 2006.

[46] Deborah Estrin, Ramesh Govindan, John Heidemann, and Satish Kumar. Next century challenges: scalable coordination in sensor networks. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 263–270. ACM, 1999.

[47] Pei Huang, Li Xiao, Soroor Soltani, Matt W Mutka, and Ning Xi. The evolution of MAC protocols in wireless sensor networks: a survey. *Communications Surveys & Tutorials, IEEE*, 15(1):101–120, 2013.

[48] Sinem Coleri Ergen and Pravin Varaiya. Tdma scheduling algorithms for wireless sensor networks. *Wireless Networks*, 16(4):985–997, 2010.

[49] Ilker Demirkol, Cem Ersoy, Fatih Alagoz, et al. MAC protocols for wireless sensor networks: a survey. *IEEE Communications Magazine*, 44(4):115–121, 2006.

[50] Mohammad Hossein Sedighi Gilani, Iman Sarrafi, and Maghsoud Abbaspour. An adaptive CSMA/TDMA hybrid MAC for energy and throughput improvement of wireless sensor networks. *Ad Hoc Networks, Elsevier*, 11(4):1297–1304, 2013.

[51] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, page 10. IEEE, 2000.

[52] Jalel Ben-Othman, Lynda Mokdad, and Bashir Yahya. An energy efficient priority-based QoS MAC protocol for wireless sensor networks. In *International Conference on Communications (ICC)*, pages 1–6. IEEE, 2011.

[53] Sinem Coleri Ergen and Pravin Varaiya. PEDAMACS: power efficient and delay aware medium access protocol for sensor networks. *IEEE Transactions on Mobile Computing*, 5(7):920–930, 2006.

[54] Leonard Kleinrock and Simon Lam. Packet switching in a multiaccess broadcast channel: performance evaluation. *IEEE Transactions on Communications*, 23(4): 410–423, 1975.

[55] Wei Ye, John Heidemann, and Deborah Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Transactions on Networking*, 12(3):493–506, 2004.

[56] Tijs Van Dam and Koen Langendoen. An adaptive energy-efficient MAC protocol for wireless sensor networks. *ACM Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, pages 171–180, 2003.

[57] Peng Lin, Chunming Qiao, and Xin Wang. Medium access control with a dynamic duty cycle for sensor networks. In *Wireless Communications and Networking Conference*, volume 3, pages 1534–1539. IEEE, 2004.

[58] Joseph Polastre, Jason Hill, and David Culler. Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pages 95–107. ACM, 2004.

[59] Michael Buettner, Gary V Yee, Eric Anderson, and Richard Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pages 307–320. ACM, 2006.

[60] I Chlamtec, A Farago, AD Myers, VR Syrotiuk, and G Zaruba. A performance comparison of hybrid and conventional MAC protocols for wireless networks. In *Proceedings on 51st Vehicular Technology Conference*, volume 1, pages 201–205. IEEE, 2000.

[61] Injong Rhee, Ajit Warrier, Mahesh Aia, Jeongki Min, and Mihail L Sichitiu. Z-MAC: a hybrid MAC for wireless sensor networks. *IEEE/ACM Transactions on Networking*, 16(3):511–524, 2008.

[62] Ajinkya Rajandekar and Biplab Sikdar. A survey of MAC layer issues and protocols for machine-to-machine communications. *IEEE Internet of Things Journal*, 2(2):175–186, 2015.

[63] Gerhard Weiss. *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT press, 1999.

[64] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: a survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[65] Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*. MIT Press, 1998.

[66] George Klir and Bo Yuan. *Fuzzy sets and fuzzy logic*, volume 4. New Jersey: Prentice Hall, 1995.

[67] Lotfi A Zadeh. Fuzzy logic= computing with words. *IEEE Transactions on Fuzzy Systems*, 4(2):103–111, 1996.

[68] Suman Lata and Mohammad Ayyub. Fuzzy inference system. *International Journal of Applied Engineering Research*, 9(7):805–813, 2014.

[69] Mandavilli Srinivas and Lalit M Patnaik. Genetic algorithms: a survey. *Computer*, 27(6):17–26, 1994.

[70] Melanie Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[71] Julian F Miller and Peter Thomson. Cartesian genetic programming. In *Genetic Programming*, pages 121–132. Springer, 2000.

[72] J Smith and A Eiben. Introduction to evolutionary computing. *Springer Natural Computing Series*, 2003.

[73] Lawrence J. Fogel. *Intelligence Through Simulated Evolution: Forty Years of Evolutionary Programming*. John Wiley & Sons, Inc., New York, USA, 1999. ISBN 0-471-33250-X.

[74] Simon S Cross, Robert F Harrison, and R Lee Kennedy. Introduction to neural networks. *The Lancet*, 346(8982):1075–1079, 1995.

[75] Dipankar Dasgupta. Advances in artificial immune systems. *Computational Intelligence Magazine*, 1(4):40–49, 2006.

[76] Stephen Wolfram. Universality and complexity in cellular automata. *Physica D: Nonlinear Phenomena*, 10(1):1–35, 1984.

[77] Martin Gardner. Mathematical games: the fantastic combinations of John Conway's new solitaire game 'life'. *Scientific American*, 223(4):120–123, 1970.

[78] S Nandi, BK Kar, and P Pal Chaudhuri. Theory and applications of cellular automata in cryptography. *IEEE Transactions on computers*, 43(12):1346–1357, 1994.

[79] Renan O Cunha, Aloizio P Silva, Antonio AF Loureiro, and Linnyer B Ruiz. Simulating large wireless sensor networks using cellular automata. In *Proceedings of the 38th annual Symposium on Simulation*, pages 323–330. IEEE Computer Society, 2005.

[80] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Number 1. Oxford University Press, 1999.

[81] Riccardo Poli, James Kennedy, and Tim Blackwell. Particle swarm optimization. *Swarm Intelligence*, 1(1):33–57, 2007.

[82] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. Ant colony optimization. *IEEE Computational Intelligence Magazine*, 1(4):28–39, 2006.

[83] DT Pham, A Ghanbarzadeh, E Koc, S Otri, S Rahim, and M Zaidi. The bees algorithm-a novel tool for complex optimisation problems. In *Proceedings of the 2nd Virtual International Conference on Intelligent Production Machines and Systems*, pages 454–459, 2006.

[84] S Binitha, S Siva Sathya, et al. A survey of bio inspired optimization algorithms. *International Journal of Soft Computing and Engineering*, 2(2):137–151, 2012.

[85] C Peeters, J Liebig, and B Hölldobler. Sexual reproduction by both queens and workers in the ponerine ant harpegnathos saltator. *Insectes Sociaux*, 47(4):325–332, 2000.

[86] Yi Chu, Selahattin Kosunalp, Paul D Mitchell, David Grace, and Tim Clarke. Application of reinforcement learning to medium access control for wireless sensor networks. *Engineering Applications of Artificial Intelligence*, 46:23–32, 2015.

[87] Nils Morozs, David Grace, and Tim Clarke. Distributed Q-learning based dynamic spectrum access in high capacity density cognitive cellular systems using secondary LTE spectrum sharing. In *International Symposium on Wireless Personal Multimedia Communications*, pages 462–467. IEEE, 2014.

[88] Liang Zhao and Qilian Liang. Fuzzy deployment for wireless sensor networks. In *Proceedings of the International Conference on Computational Intelligence for Homeland Security and Personal Safety*, pages 79–83. IEEE, 2005.

[89] Patrick N Ngatchou, Warren LJ Fox, and Mohamed A El-Sharkawi. Distributed sensor placement with sequential particle swarm optimization. In *Proceedings of the Swarm Intelligence Symposium*, pages 385–388. IEEE, 2005.

[90] Jianming Hu, Jingyan Song, Xiaojing Kang, and Mingchen Zhang. A study of particle swarm optimization in urban traffic surveillance system. In *Multiconference on Computational Engineering in Systems Applications*, pages 2056–2061. IEEE, 2006.

[91] Sergio González-Valenzuela, Son T Vuong, and Victor CM Leung. A reinforcement-learning approach to service directory placement in wireless ad-hoc networks. In *5th International Workshop on Applications and Services on Wireless Networks*. IEEE, 2005.

[92] Guo-Fang Nan, Min-Qiang Li, and Jie Li. Estimation of node localization with a real-coded genetic algorithm in WSNs. In *International Conference on Machine Learning and Cybernetics*, volume 2, pages 873–878. IEEE, 2007.

[93] Michael Marks and Ewa Niewiadomska-Szynkiewicz. Two-phase stochastic optimization to sensor network localization. In *Sensor Technologies and Applications, 2007. SensorComm 2007. International Conference on*, pages 134–139. IEEE, 2007.

[94] Raghavendra V Kulkarni, Ganesh K Venayagamoorthy, and Maggie X Cheng. Bio-inspired node localization in wireless sensor networks. In *International Conference on Systems, Man and Cybernetics*, pages 205–210. IEEE, 2009.

[95] Justin A Boyan and Michael L Littman. Packet routing in dynamically changing networks: a reinforcement learning approach. *Advances in Neural Information Processing Systems*, pages 671–671, 1994.

[96] Rocıo Arroyo-Valles, Rocío Alaiz-Rodriguez, Alicia Guerrero-Curieses, and Jesús Cid-Sueiro. Q-probabilistic routing in wireless sensor networks. In *3rd International Conference on Intelligent Sensors, Sensor Networks and Information*, pages 1–6. IEEE, 2007.

[97] Jim Dowling, Eoin Curran, Raymond Cunningham, and Vinny Cahill. Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 35(3):360–372, 2005.

[98] A Forstert and Amy L Murphy. FROMS: feedback routing for optimizing multiple sinks in WSN with reinforcement learning. In *3rd International Conference on Intelligent Sensors, Sensor Networks and Information*, pages 371–376. IEEE, 2007.

[99] Shailesh Kumar and Risto Miikkulainen. Dual reinforcement Q-routing: an on-line adaptive routing algorithm. In *Artificial Neural Networks In Engineering*, 1997.

[100] Shaoqiang Dong, Prathima Agrawal, and Krishna Sivalingam. Reinforcement learning based geographic routing protocol for UWB wireless sensor network. In *Global Telecommunications Conference*, pages 652–656. IEEE, 2007.

[101] Tiago Camilo, Carlos Carreto, Jorge Sá Silva, and Fernando Boavida. An energy-efficient ant-based routing algorithm for wireless sensor networks. In *Ant Colony Optimization and Swarm Intelligence*, pages 49–59. Springer, 2006.

[102] Mesut Gunes, Udo Sorges, and Imed Bouazizi. ARA-the ant-colony based routing algorithm for MANETs. In *Proceedings of International Conference on Parallel Processing Workshops*, pages 79–85. IEEE, 2002.

[103] Ziqiang Wang, Xia Sun, and Dexian Zhang. A PSO-based multicast routing algorithm. *IEEE Computer Society International Conference on Computing, Networking and Communications*, 4:664–667, 2007.

[104] SM Guru, SK Halgamuge, and S Fernando. Particle swarm optimisers for cluster formation in wireless sensor networks. In *Proceedings of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing Conference*, pages 319–324. IEEE, 2005.

[105] Dong-Chul Park and Seung-Eok Choi. A neural network based multi-destination routing algorithm for communication network. In *Proceedings of World Congress on Computational Intelligence, Neural Networks*, volume 2, pages 1673–1678. IEEE, 1998.

[106] Yi Chu, Paul D Mitchell, and David Grace. ALOHA and Q-Learning based medium access control for wireless sensor networks. In *International Symposium on Wireless Communication Systems*, pages 511–515. IEEE, 2012.

[107] Zhenzhen Liu and Itamar Elhanany. Rl-mac: A QoS-aware reinforcement learning based MAC protocol for wireless sensor networks. In *Proceedings of the In-*

ternational Conference on Networking, Sensing and Control, pages 768–773. IEEE, 2006.

[108] Ming-Hui Jin, D Frank Hsu, and Cheng-Yan Kao. Compact genetic algorithm for performance improvement in hierarchical sensor networks management. *Journal of Interconnection Networks*, 7(01):101–115, 2006.

[109] Weilian Xue and Zhongxian Chi. An immune algorithm based node scheduling scheme of minimum power consumption and no collision for wireless sensor networks. In *International Conference on Network and Parallel Computing Workshops*, pages 630–635. IEEE, 2007.

[110] Qingchun Ren and Qilian Liang. Secure media access control (MAC) in wireless sensor networks: intrusion detections and countermeasures. In *15th International Symposium on Personal, Indoor and Mobile Radio Communications*, volume 4, pages 3025–3029. IEEE, 2004.

[111] Raghavendra V Kulkarni and Ganesh K Venayagamoorthy. Neural network based secure media access control protocol for wireless sensor networks. In *International Joint Conference on Neural Networks*, pages 1680–1687. IEEE, 2009.

[112] Debraj Basu, Giovanni Moretti, Gourab Sen Gupta, and Stephen Marsland. Wireless sensor network based smart home: sensor selection, deployment and monitoring. In *Sensors Applications Symposium*, pages 49–54. IEEE, 2013.

[113] Stephen M George, Wei Zhou, Harshavardhan Chenji, Myounggyu Won, Yong Oh Lee, Andria Pazarloglou, Radu Stoleru, and Prabir Barooah. Distressnet: a wireless ad hoc and sensor network architecture for situation management in disaster response. *Communications Magazine*, 48(3), 2010.

[114] Abdelrahman M Ibrahim, Tamer ElBatt, and Amr El-Keyi. Coverage probability analysis for wireless networks using repulsive point processes. In *24th International Symposium on Personal Indoor and Mobile Radio Communications*, pages 1002–1007. IEEE, 2013.

[115] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[116] Raj Jain, Dah-Ming Chiu, and William R Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer system. *Eastern Research Laboratory, Digital Equipment Corporation Hudson, MA*, 38, 1984.

[117] IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. *IEEE Std 802.11-1997*, pages 1–445, Nov 1997. doi: 10.1109/IEEESTD.1997.85951.

[118] Wei Ye, John Heidemann, and Deborah Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings of Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 3, pages 1567–1576. IEEE, 2002.

[119] Melike Yigit, Eyup Alper Yoney, and V Cagri Gungor. Performance of MAC protocols for wireless sensor networks in harsh smart grid environment. In *First International Black Sea Conference on Communications and Networking*, pages 50–53. IEEE, 2013.

[120] Xiangheng Liu and Andrea Goldsmith. Wireless network design for distributed control. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 3, pages 2823–2829. IEEE, 2004.

[121] Christophe J Merlin and Wendi B Heinzelman. Duty cycle control for low-power-listening MAC protocols. *IEEE Transactions on Mobile Computing*, 9(11): 1508–1521, 2010.

[122] Paul Patras, Albert Banchs, and Pablo Serrano. A control theoretic framework for performance optimization of ieee 802.11 networks. In *INFOCOM Workshops 2009, IEEE*, pages 1–2. IEEE, 2009.

[123] Craig W Reynolds. Flocks, herds and schools: a distributed behavioral model. *ACM SIGGRAPH Computer Graphics*, 21(4):25–34, 1987.

[124] Reza Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, 2006.

[125] Kennedy James and Eberhart Russell. Particle swarm optimization. In *Proceedings of International Conference on Neural Networks*, pages 1942–1948, 1995.

[126] James Kennedy. Thinking is social: experiments with the adaptive culture model. *Journal of Conflict Resolution*, 42(1):56–76, 1998.

[127] Jagdish Chand Bansal, PK Singh, Mukesh Saraswat, Abhishek Verma, Shimpi Singh Jadon, and Ajith Abraham. Inertia weight strategies in particle swarm optimization. In *Third World Congress on Nature and Biologically Inspired Computing*, pages 633–640. IEEE, 2011.

[128] Cheolhee Park and Theordore S Rappaport. Short-range wireless communications for next-generation networks: Uwb, 60 ghz millimeter-wave wpan, and zigbee. *IEEE Wireless Communications*, 14(4), 2007.

[129] James Kennedy. Particle swarm optimization. In *Encyclopedia of Machine Learning*, pages 760–766. Springer, 2010.

[130] Marco Dorigo and Mauro Birattari. Ant colony optimization. In *Encyclopedia of Machine Learning*, pages 36–39. Springer, 2010.

[131] Ed Callaway, Paul Gorday, Lance Hester, Jose A Gutierrez, Marco Naeve, Bob Heile, and Venkat Bahl. Home networking with ieee 802.15. 4: a developing standard for low-rate wireless personal area networks. *IEEE Communications magazine*, 40(8):70–77, 2002.

[132] Han-Fu Chen. *Robbins-Monro Algorithm*. Stochastic Approximation and Its Applications, ser. Nonconvex Optimization and Its Applications, Springer, vol. 64, pp. 1-24, 2002. ISBN 978-1-4020-0806-1.

[133] Ioan Cristian Trelea. The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters*, 85(6):317–325, 2003.

[134] E Maltby, CJ Legg, and MCF Proctor. The ecology of severe moorland fire on the North York moors: effects of the 1976 fires, and subsequent surface and vegetation development. *The Journal of Ecology*, pages 490–518, 1990.

[135] Kewei Sha, Weisong Shi, and Orlando Watkins. Using wireless sensor networks for fire rescue applications: requirements and challenges. *IEEE International Conference on Electro/Information Technology*, pages 239–244, 2006.

[136] Shu Du, Amit Kumar Saha, and David B Johnson. RMAC: A routing-enhanced duty-cycle MAC protocol for wireless sensor networks. *26th IEEE International Conference on Computer Communications*, pages 1478–1486, 2007.

[137] Changle Li, Hanxiao Zhang, Binbin Hao, and Jiandong Li. A survey on routing protocols for large-scale wireless sensor networks. *Sensors*, 11(4):3498–3526, 2011.

[138] Jamal N Al-Karaki and Ahmed E Kamal. Routing techniques in wireless sensor networks: a survey. *IEEE Wireless Communications*, 11(6):6–28, 2004.

[139] Atsushi Tero, Seiji Takagi, Tetsu Saigusa, Kentaro Ito, Dan P Bebber, Mark D Fricker, Kenji Yumiki, Ryo Kobayashi, and Toshiyuki Nakagaki. Rules for biologically inspired adaptive network design. *Science*, 327(5964):439–442, 2010.

[140] Félix Gómez Mármol and Gregorio Martínez Pérez. Providing trust in wireless sensor networks using a bio-inspired technique. *Telecommunication Systems*, 46 (2):163–180, 2011.

[141] Raghavendra V Kulkarni and Ganesh Kumar Venayagamoorthy. Bio-inspired algorithms for autonomous deployment and localization of sensor nodes. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 40 (6):663–675, 2010.

[142] Pedro G Espejo, Sebastián Ventura, and Francisco Herrera. A survey on the application of genetic programming to classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 40(2):121–144, 2010.

[143] Simon Garnier, Jacques Gautrais, and Guy Theraulaz. The biological principles of swarm intelligence. *Swarm Intelligence*, 1(1):3–31, 2007.

[144] Marco Dorigo, Eric Bonabeau, and Guy Theraulaz. Ant algorithms and stigmergy. *Future Generation Computer Systems*, 16(8):851–871, 2000.

[145] Eric Bonabeau. Social insect colonies as complex adaptive systems. *Ecosystems*, 1(5):437–443, 1998.

[146] J Camhi, G Sumbre, and G Wendler. Wing-beat coupling between flying locust pairs: preferred phase and lift enhancement. *The Journal of Experimental Biology*, 198(4):1051–1063, 1995.

[147] R Christopher Plowright and Catherine MS Plowright. Elitism in social insects: a positive feedback model. Westview Press, 1988.

[148] Jean-Louis Deneubourg, Simon Goss, Nigel Franks, Ana Sendova-Franks, Claire Detrain, and Laeticia Chrétien. The dynamics of collective sorting robot-like ants and ant-like robots. In *Proceedings of the First International Conference on Simulation of Adaptive Behavior, From Animals to Animats*, pages 356–363, 1991.

[149] Stefania Gallucci and Polly Matzinger. Danger signals: SOS to the immune system. *Current Opinion in Immunology*, 13(1):114–119, 2001.

[150] Tony Yu-Chen Tsai, Yoon Sup Choi, Wenzhe Ma, Joseph R Pomerening, Chao Tang, and James E Ferrell. Robust, tunable biological oscillations from interlinked positive and negative feedback loops. *Science*, 321(5885):126–129, 2008.

[151] James D Bever, Kristi M Westover, and Janis Antonovics. Incorporating the soil community into plant population dynamics: the utility of the feedback approach. *Journal of Ecology*, pages 561–573, 1997.

[152] Selahattin Kosunalp, Paul Daniel Mitchell, David Grace, and Tim Clarke. Practical implementation and stability analysis of ALOHA-Q for wireless sensor networks. *ETRI Journal*, 38(5):911–921, 2016.

[153] Renato E Mirollo and Steven H Strogatz. Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics*, 50(6):1645–1662, 1990.

[154] Steven H Strogatz, Ian Stewart, et al. Coupled oscillators and biological synchronization. *Scientific American-American Edition*, 269:68–68, 1993.

[155] Jean Philibert. One and a half century of diffusion: Fick, Einstein, before and beyond. *Diffusion Fundamentals*, 4(6):1–19, 2006.