

Swarm Control for Mass UAS Deployments in Disconnected Environments



Swarm Control for Mass UAS Deployments in Disconnected Environments

Dissertation Thesis

Bradley Robert Fraser

GCert (Defence Science), BSc. (Astronomy and Astrophysics)

Hons. (Information and Communication Science)

Senior Research Engineer, Defence Science and Technology Group

27th January 2022

Submitted in fulfilment of the requirements
for the degree of Doctor of Philosophy

to the

School of Computer Science
at The University of Adelaide

Advisors:

Dr Claudia Szabo

Associate Professor, The University of Adelaide

Dr Andrew Coyle

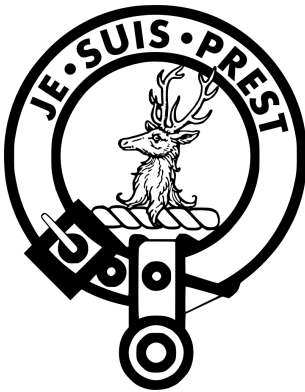
Adjunct Deputy Director, Tele-Traffic Research Centre at The University of Adelaide

Specialist Researcher, Defence Science and Technology Group

Dr Robert Hunjet

Associate Professor, University of New South Wales at the Australian Defence Force
Academy

Group Leader, Defence Science and Technology Group



¹https://asante.dev/publication/phd_thesis/

²<https://bitbucket.org/amiede/classicthesis/>

³<https://github.com/Tufte-LaTeX/tufte-latex>

IMPRINT

Swarm Control for Mass UAS Deployments in Disconnected Environments

Copyright © 2021 by Bradley Robert Fraser.

All rights reserved. Printed in Australia.

Published by The University of Adelaide, Adelaide, South Australia, Australia.

COLOPHON

This thesis was typeset using $\text{\LaTeX}$ and the memoir document class. It is based on Friedrich Weimer's thesis *Security Arguments and Tool-based Design of Block Ciphers*¹, itself a mixture of classicthesis² by André Miede and tufte-latex³, based on Edward Tufte's *Beautiful Evidence*.

The bibliography was processed by Biblalex.

The body text is set 10/14pt (long primer) on a 26pc measure. The margin text is set 8/9pt (brevier) on a 12pc measure. Matthew Carter's Charter acts as both the text and display typeface. Monospaced text uses Jim Lyles's Bitstream Vera Mono ("Bera Mono").

This document contains approximately 73,752 words and 463,018 characters.

For Bailey. The goodest of dogs.

Contents

CONTENTS	vii
ABSTRACT	ix
DECLARATION	xi
THANKS	xiii
LIST OF THESIS RESEARCH PROBLEMS	xv
LIST OF THESIS DEFINITIONS	xvi
LIST OF THESIS CONJECTURES	xvii
LIST OF ALGORITHMS	xix
LIST OF FIGURES	xxii
LIST OF TABLES	xxiii
ACRONYMS AND ABBREVIATIONS	xxv
DEFINITIONS	xxix
1 THESIS INTRODUCTION	1
1.1 Introduction	1
1.2 Contributions	5
1.3 Publications	6
1.4 Authorship Notes	8
1.5 Acknowledgements	8
1.6 Thesis Organisation	9
2 LITERATURE REVIEW	11
2.1 Data-ferrying and Remote Sensing	11
2.2 Data-relaying	27
2.3 Control Mechanisms for Swarm Systems	27
2.4 Swarming-based Role Allocation	31
2.5 Characterisation of Emergent Behaviour in UAS Swarms	32
2.6 Policy Transplantation in Multi-agent Learning	33
2.7 Open Challenges	34
3 BACKGROUND AND FORMULATIONS	39
3.1 Radio-Frequency, Communications Network and Agent Modelling	39
3.2 Random Geometric Graphs	43
3.3 Multi-Agent Deep Reinforcement Learning	46
3.4 Multi-criteria Decision Making	48
4 ON THE EQUIDISTANT FERRY SPACING FOR STATIC PATH DATA FERRY ROUTES	51
4.1 Introduction	51

4.2	Experimental Analysis	55
4.3	Mathematical Model	58
4.4	Summary	74
5	ADAPTABLE DATA TRANSFER UNDER VARIABLE GRAPH CONNECTIVITY CONDITIONS	75
5.1	Introduction	76
5.2	Experimental Analysis	80
5.3	Mathematical Model	86
5.4	Additional Investigation	99
5.5	Summary	109
6	EXTENDING COORDINATION ACROSS HETEROGENEOUSLY-CAPABLE NODES FOR MULTI-ROLE MISSIONS	111
6.1	Introduction	111
6.2	Experimental Analysis	116
6.3	Mathematical Model	121
6.4	Summary	124
7	TOWARDS SCALABLE POLICY TRANSPLANTATION	125
7.1	Introduction	125
7.2	Experimental Analysis	131
7.3	Summary	142
8	QUANTIFYING THE BREAKDOWN OF EMERGENT BEHAVIOUR USING STATE AUTOCORRELATION	143
8.1	Introduction	143
8.2	Experimental Analysis	146
8.3	Summary	155
9	CONCLUSION AND FUTURE DIRECTIONS	157
9.1	On the Equidistant Ferry Spacing for Static Path Data Ferry Routes	157
9.2	Adaptable Data Transfer under Variable Graph Connectivity Conditions	158
9.3	Extending Coordination Across Heterogeneously-capable Nodes for Multi-role Missions	159
9.4	Towards Scalable Policy Transplantation	159
9.5	Quantifying the Breakdown of Emergent Behaviour Using State Autocorrelation	160
9.6	Concluding Statement	161
A	STATEMENTS OF AUTHORSHIP	163
	BIBLIOGRAPHY	175

Abstract

Uninhabited Aerial Systems (UASs) are one of the best examples of recent disruptive technology and have justifiably become the workhorses for myriad novel military and civilian applications. UAS technology has advanced to the point where the deployment of swarms of UASs is a reasonably viable activity from a logistics point-of-view. However, fully autonomous and distributed control of such swarms remains elusive. In particular, the case where swarm members, or other network nodes they are supporting, exist in a disconnected communications state makes swarm coordination especially difficult. The high-stakes nature and potentially dangerous consequences of such research and development activities also makes their implementation exceedingly rare. Furthermore, issues pertaining to scalability of algorithms from the perspective of automated design and deployment still exist. This thesis aims to address these problems through simulated and robotic field experiments, using bio-inspired and reinforcement learning approaches to generate swarm control schemes for common UAS applications.

Of the technical parts of the thesis, Chapters 4 to 6 propose several novel swarm control algorithms to support communications and other positional-based missions. Mathematical analysis of the resultant emergent behaviour provides insight into how the coordination takes place. It further examines the way in which these algorithms adapt to different environmental conditions such as communications connectivity, swarm size and role-requirements. Chapters 7 and 8 turn to issues in swarm scalability both from the perspective of automated algorithm design and practical communications. In the former, it is shown that the transplantation of control policies generated through multi-agent reinforcement learning architectures is dependent on the way in which swarm agents observe their environment; to the best of the author's knowledge, the first such result of its kind. This result allows the deployment of large swarms without the requirement to train all its members. In the latter, the breakdown of emergent behaviour is characterised as communications channel congestion increases, leading to a new metric for measuring emergence within such algorithms.

As a complete work, the thesis makes several contributions to advancing the current state of autonomous swarm control through simulated and mathematical analysis. Where practicable, experiments were performed on real systems to further validate results in the real world. A desirable consequence of these contributions is an increase in trusted autonomy for systems that exploit swarm control.

Authorship Declaration

I certify that this work contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint-award of this degree.

The author acknowledges that copyright of published works contained within the thesis resides with the copyright holder(s) of those works.

I also give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

I acknowledge the support I have received for my research through the provision of an Australian Government Research Training Program Scholarship.

Bradley Robert Fraser

27/01/2022

Date

Thanks

I am extremely grateful to the **Defence Science and Technology Group (DSTG)** for both funding my doctoral studies and assisting with funding to present my research around the world. To the long list of people in my line management that have supported my studies over the last five years, even when my absence from the workplace was disruptive: thank-you.

I am forever grateful to my supervisors: Andrew, Claudia and Robert. Andrew, your dedication to your students, even in the face of retirement is truly appreciated. Receiving emails after a ‘couple’ of reds at 11:20pm on a Saturday is a testament to this fact. Your asides into philosophy, language and history have forced me to not take my work so seriously and to not let it get in the way of a good yarn. Your natural talent to frame a problem and write a couple of lines of code to investigate it immediately is amazing (and sometimes annoying for us mere mortals) to watch; I am grateful to have had such a great teacher. Thank-you.

Claudia, you are one of the most dedicated students to the scientific method I have had the pleasure to work with. Your constant reminders to ask myself, what I am solving, why and how, have undoubtedly made me a better researcher; your professionalism and knowledge is something that I aspire to achieve. Your dedication to helping students and genuine interest in their well-being is something that I have taken back to my own workplace. Thank-you.

Rob, I don’t have to say anything but I will anyway. You’re the reason I got into this mess. Having a mate as well as a mentor during this whole process made it just that bit easier. Thanks.

Finally, to Mum and Dad, and my wife Daina, your constant encouragement and support is without a doubt the reason that I get to type these words now. I love you.

List of Thesis Research Problems

1	Problem (Equidistant Ferry Spacing)	35
2	Problem (Connectivity-aware Data Transfer)	35
3	Problem (Buffer-constrained Ferrying)	35
4	Problem (Multi-role Coordination)	36
5	Problem (Breakdown of Emergent Behaviour)	36
6	Problem (Policy Transplantation)	36

List of Thesis Definitions

1	Definition (Policy Transplantation)	127
2	Definition (Swarm-safety)	127
3	Definition (Absolute (Explicit) World-View)	128
4	Definition (Relative (Implicit) World-View)	128

List of Thesis Conjectures

1	Conjecture (Relative agent world-views)	141
---	---	-----

List of Algorithms

1	Equidistant Ferrying Algorithm	54
2	Adaptive Ferry-Relay Algorithm	78
3	Ferry Flocking Algorithm	106
4	Attraction-Repulsion Mesh Algorithm	144

List of Figures

2.1	Classifying data-ferrying systems using ferry characteristics. . . .	13
2.2	The data-ferrying spectrum.	16
2.3	Classifying platform control mechanisms for swarms.	31
3.1	An example of the LNPLM.	41
3.2	Receive probability against packet size using Bernoulli trials. . .	43
3.3	A 100 node $r = 0.1$ RGG instantiation.	44
3.4	An RGG example.	45
3.5	Data-ferry and data-relay counts on a RGG.	45
4.1	Perimeter separation convergence.	57
4.2	The quadratic final data-ferry speeds and user-node service time on the equilateral triangle topology.	57
4.3	The stigmergic algorithm in action.	58
4.4	The discrete time evolution of four initial velocities.	61
4.5	The Verhulst (cobweb) diagram of the dynamical system. . . .	62
4.6	A fictitious landing scenario.	64
4.7	LHD service time over the life of the mission.	66
4.8	The speed responses of two simulated data-ferries servicing four user-nodes on a 6×6 unit square.	68
4.9	The emulated speed response on a 5×5 km square.	70
4.10	The control setup for physical ferry experiments.	71
4.11	Ferry speed response during a physical run.	73
4.12	Snapshots of the physical ferry experiments.	74
5.1	The blackhole data sink problem.	77
5.2	Data-ferry and data-relay examples in the adaptive algorithm. .	79
5.3	Localised behaviour of resource-nodes on a RGG.	79
5.4	TSP tour length with increasing graph connectivity.	81
5.5	Packet latency results over various graph connectivities.	82
5.6	The spatial history of resource-nodes during a simulation run. .	83
5.7	χ^2 histogram distance deviation of resource-node locality. . . .	84
5.8	Aggregate resource-node locality of 10 nodes and a single high data rate pair.	85
5.9	Resource-node history traces.	87
5.10	The conceptual model of the adaptive resource-node algorithm. .	88
5.11	The collisional area for a single particle of radius r travelling at velocity v for time Δt	91
5.12	Comparison of simulated and analytic collision rate.	92
5.13	The collisional area for the length L	92
5.14	A low density case where two resource-node cells overlap. . . .	93
5.15	Analytic and simulated results for the number of hand-offs be- tween resource-nodes.	94
5.16	Analytic and simulated results for the data latency.	95

5.17 RMSE and MAE in the data latency between the model and simulation with increasing range r	96
5.18 Analytic and simulated results for the data latency with user-node routing enabled.	97
5.19 A phase transition in the probability of data latency being less than 50 time-steps.	98
5.20 Three ferrying schemes for the two resource-node ferrying problem.	101
5.21 Throughput for the two resource-node ferrying problem with increasing buffer size.	101
5.22 Delay for the two resource-node ferrying problem with increasing buffer size.	102
5.23 Threshold packet generation.	103
5.24 A flock of 10 nodes with the origin as the group objective. Lines represent valid communications links.	105
5.25 Average latency for a single topology.	107
5.26 The mean latency ratio.	107
5.27 Mean latency as a function of buffer size.	109
6.1 The communications agent scoring the surveillance agents position suggestion.	116
6.2 The multi-role decision architecture.	116
6.3 Mean visit times when role priorities are equal.	118
6.4 Mean visit times when role priorities are unequal.	119
6.5 Impact on performance of failed nodes.	120
6.6 Both the multi-role and single-role approaches quickly react to the dynamically introduced role requirement at time step 500. .	120
6.7 Heterogeneous role results.	121
7.1 A visual representation of the bullseye observation vector model.	131
7.2 Transplantation results with explicit models.	133
7.3 Different world views generated different coordination strategies.	133
7.4 The <i>II</i> model transplantation results.	135
7.5 The F550 flamewheel hexacopter.	136
7.6 The platform Command, Control and Communications (C3) architecture.	137
7.7 The five surveillance target locations nodes needed to cover. . .	137
7.8 Position traces and achieved rewards from a 4-node swarm during a flight test at the WTR.	139
7.9 Node 5 was recalled due to a hardware concern leaving the other three nodes to coordinate on target coverage.	139
7.10 A three node run displaying intelligent coordination.	140
7.11 Node 1 stands-off from the bottom left target due to node 3 drifting between the two left-most targets.	140
7.12 A swarm of four UAS at the WTR.	140
8.1 Lossless experiment results for a particular run.	149
8.2 Lossy experiment results. Error bars indicate one standard deviation.	153

8.3	The correlogram of the stigmergic data-ferry algorithm of Chapter 4.	154
8.4	Decreasing entropy in the stigmergic data-ferry algorithm of Chapter 4.	155

List of Tables

2.1	Reviewed literature in the context of the data-ferrying spectrum.	26
4.1	The achieved mean and 95% CI for the inter-ferry spacing. . . .	56
4.2	Simulated and analytical ferry separation distances.	60
4.3	Algorithm parameters for simulation experiments.	67
4.4	Algorithmic parameters for simulations on a 6×6 unit square. .	67
4.5	Emulation parameters.	68
4.6	Algorithmic parameters for the emulated mission on on a 5×5 km square.	69
4.7	Mean (μ) service times and 95% CI observed at convergence in stigmergic data-ferry experiments.	72
5.1	Summary of statistical testing. Bracketed values record the test-statistic and p – value of each test.	96
6.1	The TOPSIS decision matrix.	114
7.1	Four possible observation vector models.	130
8.1	The basic 4IPP model.	148
8.2	Parameters for the attraction-repulsion ideal baseline experiment.	148
8.3	Comparison of end-of-run metric values for the lossless and two lossy cases	150

Acronyms and Abbreviations

<i>k</i>-PCPN	<i>k</i> Path Cover Problem with Neighborhood 19
<i>k</i>-TSPN	<i>k</i> Traveling Salesman Problem with Neighborhood 19, 26
95% CI	95% Confidence Interval 55, 56, 72, 73, 117, 131
ACF	Autocorrelation Function 6, 143, 146, 150, 153, 155, 156, 160, 161
AD	Anderson-Darling 95
AI	Artificial Intelligence 157
ANN	Artificial Neural Network 46, 47, 138
AO	Area-of-Operations 21, 112, 115, 158, 160
AOFS	Assisted Opportunistic Ferry System 12
APFS	Assisted Planned Ferry System 12, 14
API	Application Programming Interface 8, 136
BDA	Battle Damage Assessment 1
BER	Bit Error Rate 41, 42
BPSK	Binary Phase-Shift Keying 41, 42, 147
BSR	Bit Success Rate 42
C3	Command, Control and Communications xxi, 137, 160
CETC	China Electronics Technology Group Corporation 4
CI	Confidence Interval 56
COTS	Commercial-Off-The-Shelf 69, 110, 160
CSMA	Carrier Sense Multiple Access 146
DL	Deep Learning 5
DMFR	Dynamic Message Ferry Delay 22
DNN	Deep Neural Network 46, 47, 125, 126, 128, 138, 160
DQN	Deep Q Network 46, 47
DRL	Deep Reinforcement Learning 5, 9, 128
DSTG	Defence Science and Technology Group xiii, 8, 9, 69
DTN	Delay-Tolerant Network 11, 14, 15, 147

- E-GAP** Extended Generalised Assignment Problem 31, 35
- FSPL** Free-Space Path Loss 40
- FTP** File Transfer Protocol 147
- GAP** Generalised Assignment Problem 31, 32, 35
- GCS** Ground Control Station 134, 136, 137
- GPS** Global Positioning System 1, 134
- GUI** Graphical User Interface 72
- HADR** Humanitarian-Assistance and Disaster Relief 35, 52
- HAIL** Hardware Abstraction and Interface Layer 8, 68, 70–72, 136
- HITL** Human-In-The-Loop 1
- HOTL** Human-On-The-Loop 72
- HTTP** Hypertext Transfer Protocol 147
- IEEE** Institute of Electrical and Electronics Engineers 41, 42, 147
- IP** Internet Protocol 134, 137, 147
- IPP** Interrupted Poisson Process 147
- ISTAR** Information, Surveillance, Target Acquisition and Reconnaissance 1
- KS** Kolmogorov–Smirnov 95
- LAN** Local Area Network 31, 134, 137
- LDPLM** Log-Distance Path Loss Model 41
- LHD** Landing Helicopter Dock 63–66
- LNPLM** Log-Normal Path Loss Model 40, 41, 147, 151
- LP** Linear Program 126
- LPD** Low Probability of Detection 52
- LPI** Low Probability of Interception 52
- MAC** Medium Access Control 146, 147
- MADDPG** Multi-Agent Deep Deterministic Policy Gradient 46, 48, 129, 132
- MADRL** Multi-Agent Deep Reinforcement Learning 5, 9, 36, 42, 43, 46, 124, 125, 141, 159, 161
- MAE** Mean Absolute Error 95
- MAJ** Major 8, 158

- MANET** Mobile Ad-hoc Network 3, 4, 11, 13–15, 25, 27, 39, 80, 126
- MAS** Multi-Agent System 113, 125, 157, 161
- MASON** Multi-Agent Simulator of Networks 42, 55, 67, 81, 146
- MAVLink** Micro Air Vehicle Link 71, 136, 137
- MaxSNP** Max Strict NP 22
- MCDM** Multi-criteria Decision Making 48, 49
- MDP** Markov Decision Process 21
- MDP** Markov Decision Process 26
- MFR** Message Ferry Route 17, 21, 22
- NIS** Negative Ideal Solution 49
- NP** Non-deterministic, Polynomial time 12, 17, 35
- NPS** Naval Postgraduate School 4, 8, 158
- NUC** New Unit of Computing 134, 136, 137
- OFS** Opportunistic Ferry System 12
- PID** Proportional-Integral-Derivative 68
- PIS** Positive Ideal Solution 49
- PLI** Position-Location Information 77
- PODMP** Partially-observable MDP 23
- POMDP** Partially Observable Markov Decision Process 26
- ProtoBuf** Google Protocol Buffer 137
- PSK** Phase-Shift Keying 42
- QoS** Quality-of-Service 20, 64, 65, 74, 147, 156
- RAAF** Royal Australian Air Force 9, 136
- RAAFAR** RAAF Active Reserve 9
- RF** Radio-Frequency xxix, xxx, 3, 4, 21, 39, 40, 52, 54, 68, 74, 105, 109, 129, 137, 144, 159, 161
- RGG** Random Geometric Graph 9, 43, 44, 76, 79–83, 85, 87, 93, 96, 109, 158
- RIP** Route Inspection Problem 17
- RL** Reinforcement Learning 5, 21, 26, 32, 46, 47, 128, 142
- RMSE** Root Mean Squared Error 95

- ROS** Robot Operating System 70–72
- SLIP** Serial Line Internet Protocol 134, 137
- SNR** Signal-to-Noise ratio xxix, 40–42, 54, 147, 148
- SSH** Secure SHell 72
- sUAS** small Uninhabited Aerial System 134
- SWaP** Size, Weight and Power 28, 52
- Swarm-GAP** Swarm Generalised Assignment Problem 32
- TCP** Transmission Control Protocol 147
- TOPSIS** Technique for Order of Preference by Similarity to Ideal Solution 6, 49, 114–117
- TSP** Travelling Salesman Problem 12, 17–22, 25, 26, 54, 80–82, 87, 88, 95, 109, 158
- TSPN** Traveling Salesman Problem with Neighborhood 18–20, 26
- UAS** Uninhabited Aerial System ix, 1–4, 8, 12, 28–30, 32, 35, 36, 49, 51, 63–67, 69, 74, 105, 109, 111, 112, 115, 122, 124, 129, 134, 136, 138, 140, 142, 157, 160, 161
- UDP** User Datagram Protocol 134, 137, 147
- USB** Universal Serial Bus 72
- USMC** United States Marine Corps 8, 158
- UxV** Unhabited [Aerial, Ground, Surface, Underwater] System 3
- VNC** Virtual Network Computing 72
- WOFF** Warrant Officer 9
- WSN** Wireless Sensor Network 4, 11, 14–17, 25, 35, 126
- WTR** Woomera Test Range 9, 136
- YOLO** You Only Look Once 142

Definitions

- agent** A software agent; an entity with agency to make its own decisions. [xxi](#), [xxx](#), [3](#), [5](#), [6](#), [24](#), [31](#), [32](#), [36](#), [46](#), [48](#), [111–116](#), [121](#), [129–132](#), [134](#), [135](#), [141–144](#)
- autocorrelation** The correlation of a signal with a delayed copy of itself as a function of delay. [151](#), [155](#), [156](#)
- component** A subgraph in which any two [vertices](#) are connected by one or more paths. [xxix](#)
- correlogram** A chart of correlation statistics. [xxii](#), [150–155](#)
- data-ferry** A [resource-node](#) dedicated to providing or assisting with user network communications by physically transferring data between disconnected graph components. [xx](#), [xxii](#), [xxiii](#), [xxix](#), [xxx](#), [11](#), [12](#), [14](#), [15](#), [25](#), [35](#), [45](#), [51–60](#), [63](#), [64](#), [68–70](#), [72](#), [73](#), [75](#), [76](#), [79](#), [83](#), [99](#), [105](#), [109](#), [154](#), [155](#), [161](#)
- data-relay** A [resource-node](#) dedicated to providing or assisting with user network communications by bridging two or more disconnected [graph components](#). [xx](#), [45](#), [75](#), [76](#), [83](#), [109](#)
- degree** The [degree](#) of a [graph vertex](#) is the number of [edges](#) that are incident to a [vertex](#). [xxix](#)
- edge** One of the two basic units from which a [graph](#) is constructed. An edge connects two [vertices](#). [xxix](#), [42](#)
- Euclidean norm** The Euclidean norm, l^2 -norm or 2-norm is a distance function that defines the Pythagorean distance for n -dimensional vector spaces. [39](#), [131](#)
- expected degree** The [expected value](#) of the [graph degree](#). [44](#)
- expected value** The expected value of a random variable **X** is a generalization of the weighted average, and is intuitively the arithmetic mean of a large number of independent realizations of **X**. [xxix](#)
- fading** Variation in [Radio-Frequency \(RF\)](#) signal energy attenuation during propagation. [40](#), [41](#)
- ferry** See [data-ferry](#). [xx](#), [xxiii](#), [12–24](#), [27](#), [30](#), [34](#), [35](#), [44](#), [52–56](#), [58–70](#), [72–78](#), [82](#), [99–102](#), [105–110](#), [153](#), [154](#), [157](#), [158](#)
- graph** A system of [vertices](#) connected in pairs by [edges](#). [xxix](#), [xxx](#), [20](#), [39](#), [42](#), [74](#), [76](#), [80](#), [144](#)
- interference** Unwanted [RF](#) energy that reduces the [Signal-to-Noise ratio \(SNR\)](#). [40](#)

node A physical unit of a network. [xxi](#), [xxx](#), [11–24](#), [39](#), [40](#), [44](#), [52–55](#), [63–65](#), [67–72](#), [74](#), [76](#), [129](#), [136–140](#), [143–148](#), [150](#), [151](#), [155](#), [156](#), [158](#)

path-loss The attenuation of [RF](#) signal energy during propagation. [40](#)

policy Defines an agent's way of behaving in a given state. [xxx](#), [47](#), [48](#), [132](#), [134–137](#), [140–142](#)

policy-transplantation Copying a learned [policy](#) from one [agent](#) into another. [xxx](#), [9](#), [34](#), [36](#), [132](#), [141](#), [142](#)

range The communications radius of a circular (2-dimensions) or spherical (3-dimensions) propagation model of a transmitting device. [39](#), [55](#), [67](#)

resource-node An autonomous [node](#) with the task of performing some service. [xx](#), [xxi](#), [xxix](#), [11](#), [27](#), [39](#), [44](#), [75–95](#), [98](#), [99](#), [101](#), [102](#), [109–123](#), [158](#), [159](#)

service time Time between [data-ferry](#) visits. [xx](#), [xxiii](#), [56](#), [57](#), [64–66](#), [69](#), [72](#), [73](#), [112](#), [158](#), [159](#)

stigmergy Indirect communication through the environment to facilitate swarm intelligent coordination. Often found in social insect species such as ant and termites in the form of pheromones. [52](#), [74](#), [78](#), [112](#)

swarm A collection of individuals that coordinate using the principles of swarm-intelligence, emergent behaviour and/or those of multi-agent systems. [24](#), [27–30](#), [32](#), [35](#), [36](#)

swarm-safe Successful [policy-transplantation](#) where copied [policies](#) perform well in their new host [agent](#). [128](#), [138](#), [160](#)

target-node A target-node of a particular [agent](#). Target-nodes may be another physical node, such as for a communications [agent](#), or simply a location such as for a surveillance [agent](#) [xxx](#), [112–117](#), [121](#), [123](#), [159](#)

user-node A [node](#) a part of the user network. [xx](#), [xxi](#), [3](#), [11](#), [12](#), [14](#), [15](#), [18](#), [24](#), [27](#), [35](#), [39](#), [44](#), [51–60](#), [63–69](#), [71](#), [73–77](#), [79–82](#), [84–88](#), [91–99](#), [104–106](#), [108–110](#), [123](#), [143](#), [147](#), [153](#), [158](#)

vertex One of the two basic units from which a [graph](#) is constructed. [xxix](#)

1

Thesis Introduction

1.1 INTRODUCTION

Goldman Sachs forecasted in 2016 that a \$100 billion market opportunity existed for **Uninhabited Aerial Systems (UASs)** into 2020 [Gol16]. While the reality of this forecast may have been ambitious, this predicted proliferation of drone technology is well underway with the current market value of \$22.5 billion set to nearly double to \$43 billion by 2025 [Res20]. As a consequence, **UAS** technology is becoming ubiquitous in a whole host of applications from **Information, Surveillance, Target Acquisition and Reconnaissance (ISTAR)**, **Battle Damage Assessment (BDA)** and strike [HO20a; Sto+19; Bor20; Kin16] and defensive communications [HH11; Hui+11; HPK17; Usb+14] in the military domain, to precision agriculture [Lot+17; RG+20], remote sensing for structures such as gas and oil pipelines [BS15; Paj15], wild fire monitoring [HS18; JK18b; IG18], disaster relief [Mic+18; Erd+17], search and rescue [ST10; GPM19] and communications [CTL20] in the civil space. Of course, there is also a huge demand for drones for photography and videography, particularly for the everyday consumer market, with recent reports estimating 1.3 million recreational drones are in use in the U.S. alone [Fed20]. **UASs** represent a true disruptive technology, and like the internet and **Global Positioning System (GPS)** before them, have made their way from the military to civil and consumer markets, and in doing so have opened up a whole new domain which had previously been denied by complexity and financial barriers.

While there are still improvements to be made in physical **UAS** hardware, particularly in battery technology for longer endurance times, hardware is mostly fit for purpose and the final frontier for drones, and robotics in general, is their onboard intelligence. **Human-In-The-Loop (HITL)** remotely-piloted missions require a constant communication channel between the drone and human controller. Not only does this introduce a single-point-of-failure and legitimate threat vector [BP18], constant direct control is an obstacle to deployment due to the inherent limitations of being human; fatigue, concentration and the speed of comprehension and decision-making, as well as the need for specialist training make human control a bottleneck to mass and persistent **UAS** deployments. Specialist interfaces incorporating human-factors considerations must also be considered [SHK21]. A meta-analysis of 27 swarm management interfaces suggests that in general “raising

“Πάντων γὰρ ὅσα πλείω μέρη ἔχει καὶ μὴ
ἔστιν οἶον σωρὸς τὸ πᾶν ἄλλ’ ἔστι τι τὸ
ὅλον παρὰ τὰ μέρη.
*For however many things have a plurality of
parts and are not merely a complete
aggregate but instead some kind of a whole
beyond its parts.*”
—Aristotle

the swarm’s level of autonomy to reduce operator cognitive workload are beneficial and improve situation awareness (SA).” [HN17]. Automated or scripted missions on-the-other-hand removes direct control from the human during the mission. However, this approach simply moves the human burden from piloting to mission-planning, and while planning tools do exist, mission-planning can be an arduous task for a single entity, let alone more than one. Furthermore, automated mission planning is fragile to unplanned for scenarios as they have no ability to adapt to changing conditions apart from those the human mission-planner conceived of, and planned for. More importantly, mission-planning a group of entities as isolated singular entities does not produce a ‘team’ and cannot take advantage of the considerable efficiencies that cooperative teamwork can provide; communication and interaction between entities is required. Onboard intelligence in the form of *autonomous* platform control however, can remove the burden on human operators and simultaneously improve mission efficiency and likelihood of success through the deployment of a cooperative team of redundant entities that react to their environment and peers on-the-fly. For this reason, much of the referenced research above pertains to teams or *swarms* of aerial platforms, indicating the trends of current research [Chu+18].

Swarms are often thought of as simply a group of many entities and hence by definition offers redundancy against individual losses. However, the field of *swarm intelligence* as inspired by the natural world offers much more than simple redundancy, describing the collective behaviour of decentralised, self-organising systems [BDT99]. Examples of swarm behaviour, such as the foraging of certain ant species, demonstrate an approach to cooperative coordination that produces a global (and sometimes optimal) solution [DBT00]. Swarm individuals are independent decision makers as well as physically separate entities. The system then is both decentralised and distributed. Moreover, swarm individuals make decisions using only local information, not requiring a global system state; all highly desirable traits for large autonomous systems. Coupled with the concept of emergent behaviour and swarm robotics, research into the application of swarm intelligence for control of UAS deployments has grown significantly in recent years [ZRW20]. Many algorithms are borrowed from, or inspired by, their natural counterparts, with the quintessential example being Reynolds’ rules to mimic the flocking behaviour of birds and fish in computer animation [Rey87a; OS06]. Other swarming-based systems that exploit emergent behaviour can be found throughout the natural world and extensive research has analysed the various emergent properties shown by such systems [BDT99; TB99]. In particular, nature appears quite adept at navigating the exploration vs exploitation dilemma [BT+14] found in optimisation problems where one must balance the exploration of the entire solution space with the exploitation of promising areas that have already been discovered. This has thus given rise to many optimisation algorithms inspired by natural systems including ants [DBS06], bees [KB07] and wolves [MML14] among others [WNMA15; MLY17]. These population-based optimisation algorithms, of course, take place in computer software and sharing information between agents is trivial. Their natural counterparts on-the-other-hand use their senses such as: smell (ant pheromones for example); and sight (the waggle-

dance in bees [Yuc+13]), to communicate data between them. One major difference however, when implementing multi-agent systems as physical robots, is the medium for agent-communication.

Most physical applications of swarming systems in **Unhabited [Aerial, Ground, Surface, Underwater] Systems (UxVs)** rely on traditional **Radio-Frequency (RF)** communications to transfer data from one swarm member to another, and while they provide a wireless mechanism to transfer data, such networks are not infallible and data loss is a common occurrence within them. This data loss is typically hidden from us in our day-to-day experience of these networks through the employment of protocols that ensure data reaches its desired destination. However, swarming systems more closely represent **Mobile Ad-hoc Networks (MANETs)** [Toh02] than traditional managed networks. In these networks, data is transferred directly between network nodes without the need for a network controller. Moreover, nodes are free to move without concern to the underlying network connectivity and thus data loss is often unavoidable when networks fracture due to nodes moving out of range of each other. Like data loss, platform loss is also likely when employing *attritable* platforms [HO20b]. Thus, **UAS** swarming, and the domain of this thesis, exists at the nexus of distributed and swarm intelligent algorithm design, emergent behaviour, **RF** communications and network modelling. In particular, this thesis focusses on two main issues: adaptability and scalability.

Due to the aforementioned attritable platforms, the size of employed swarms can grow or shrink depending on the environment and mission requirements. This variability in the operating environment within a swarming system, will affect its performance. In addition, these variations can often produce unexpected and unpredictable changes in the collective behaviour. Thus, it is of interest to develop algorithms that adapt to change positively. For example, it is shown that it is advantageous for individual units to be synchronised in a particular swarm intelligent communications algorithm (see **Chapter 4**). It is then of interest how the loss (or addition) of an individual unit affects the synchronisation of the collective. A second example, stemming from communications challenges, is that data loss may also affect the underlying user-node network, leaving it in various states of connectivity, which in this instance forms part of the swarm's environment. An algorithm that can adapt to various connectivity levels is therefore advantageous (see **Chapter 5**). Thus, adaptability is investigated through the modelling and explanation of the consequences of variability within particular instantiations of several novel swarm intelligent algorithms.

Another driving factor of this thesis is how scalability is affected by the mass deployment of platforms. One might ask what scalability issues exist within a control framework that is inherently scalable. To answer this question, consider that technology has advanced in recent years to the point where deployment of swarms, at least from a hardware perspective, is viable. From the point-of-view of control, attention is gradually shifting from a single pilot controlling a single platform, to a single pilot monitoring a swarm. As autonomy in this control increases, pilots will become 'operators' and will shift from direct control of the swarm to only exerting influence on it, leaving the swarm itself to fulfil commander intent. This increase in autonomy (and

trust), opens the door for a greater number of individual units to be deployed. For example, in late 2015, the U.S. Office of Naval Research demonstrated a swarm of 30 autonomous UASs performing manoeuvres that simulated an aerial bombardment of ground targets [Sma15]. Only a year later, the Live-Fly program from the U.S. Naval Postgraduate School (NPS), conducted the autonomous launch, flight and landing of 50 fixed-wing UASs [Chu+16]. Later that year, the China Electronics Technology Group Corporation (CETC) demonstrated the take-off and formation flight of 67 fixed-wing UASs. The U.S. Department of Defense's Strategic Capabilities Office has also demonstrated several autonomous missions carried out by 103 micro-UASs after being launched from three fighter aircraft [US 17]. More recently, CETC has released documentary evidence purporting to show a successful test of a 'barrage swarm' that launches 48 attack drones [Ham20]. Moreover, many companies now offer drone-based 'light shows' in lieu of fireworks displays that contains thousands of individual units [BBC18; Int]. While the autonomy and cooperation present within these examples varies drastically, the sheer numbers of individual units operating in close proximity produces a challenging communications environment. As the size of swarms continue to grow, more pressure will be placed on the communications infrastructure operating within the swarm, particularly if nodes generate their own data such as from video surveillance, in addition to what is required for swarm operation. This warrants an investigation into the practical scalability of these systems when utilising RF communications, given that communication loss affects the production of emergent behaviour itself in some algorithms. To perform this investigation, a metric that quantifies the 'amount' of emergent behaviour within a system is required. One contribution of this thesis is the introduction of such a metric.

The second scalability issue to be investigated is from the perspective of algorithm design. Heuristic design of swarm intelligent algorithms is a difficult task as the 'intelligence' is an emergent global property, unknown to the individual agents. It is therefore up to the designer to program the individuals in such a way as to promote the generation of this global intelligence. However, a common perspective of emergent behaviour, known as Weak Emergence [Bed08], specifies that new properties arising in the system are a result of interactions of individuals, but the property itself cannot be explained using reductionist methods. Investigations must rather rely on post-mortem analysis or simulation. This then leaves algorithm design in a rather difficult state of iterative trial-and-error design which is likely why many emergent-based algorithms borrow from systems known to generate useful emergent behaviour. Indeed, there are many algorithms that make use of Reynolds' boids or similar rules for the deployment of communications and sensor nodes in MANETs and Wireless Sensor Networks (WSNs) [KBJ13a; Hun15a; SB15a; Yua+18; Fra18b; Fra18a]. To speed up this iterative design, the trial-and-error process can be wrapped in an optimisation routine where variables that govern individual agent actions are optimised against the fitness of the global property that is produced. Such methods however are typically time intensive; the quintessential example being the evolutionary timescales required to produce the flocking, ant foraging and termite mound building of today's insects generated through

natural selection. From an artificial evolution perspective, these methods are compute intensive and often do not scale well with the number of agents. **Multi-Agent Deep Reinforcement Learning (MADRL)** is one such example of this.

The recent advances in **Deep Learning (DL)** [Sej18] have made great progress in tackling problems that were traditionally intractable; social network analysis, autonomous driving, natural language processing, speech and audio processing, computer vision and biomedicine have all benefited from the technology to name a few [Pou+18]. Fuelled by large amounts of data that have previously been labelled (given meaning by a human for some purpose), the powerful optimisation-based algorithms are able to find patterns and relationships in the data and apply it to new unseen instances. The same approaches were subsequently applied to **Reinforcement Learning (RL)** where, rather than labelled training data, the learning agent gains feedback by interacting with its environment. Such approaches have been shown to produce human- and superhuman-level performance in some problems [Mni+15; Sch+20]. This field of **Deep Reinforcement Learning (DRL)** was then naturally extended to the multi-agent space [NNN18] shifting the designer's burden from programming agent rules, to designing observation vectors and reward functions to influence the global swarm behaviour. As is typical with optimisation, hundreds of thousands and often millions of environmental interactions are required to learn the correct agent policy that produces the desired swarm behaviour. Moreover, each individual agent requires its own optimisation routine. Thus, the second scalability issue concerns itself with how resource-hungry **MADRL** approaches can be scaled to a swarm of many individuals units.

Through the above investigations, several novel swarm algorithms have been developed. Where practicable, these algorithms have been implemented on real systems. The insights and contributions offered through this work are summarised in the following section.

1.2 CONTRIBUTIONS

This thesis makes the following contributions:

► AN ALGORITHM FOR EQUIDISTANT FERRY SPACING

A distributed algorithm is proposed to produce equidistant ferrying spacing around a static path. The algorithm is based upon pheromone influencing and is shown to be extremely resilient to timing perturbations both in simulation and real-world environments. Its resilience comes from capturing an attractor solution of the underlying dynamical system. It is one solution to **Problem 1 (Equidistant Ferry Spacing)**.

► AN ALGORITHM FOR CONNECTIVITY AWARE FERRYING

A new algorithm for next-hop data-ferrying is proposed. Unlike existing similar algorithms, the algorithm efficiently transfers data around an underlying user network regardless of its connectivity level; bridging where possible and ferrying where necessary producing one solution to **Problem 2 (Connectivity-aware Data Transfer)**. This work also characterises the benefits of where

using a sub-swarm as a single ferrying unit is advantageous over completely individual units, giving insight into **Problem 3 (Buffer-constrained Ferrying)**.

► **A MULTI-ROLE SWARM CONTROL ARCHITECTURE**

An extensible swarm control architecture, based on the **Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS)** [TH11], for multi-role missions is proposed. Our experiments show that it outperforms equivalent-sized, dedicated teams. The approach is adaptive to role requirements, losses and commander intent. It is one approach to satisfying **Problem 4 (Multi-role Coordination)**.

► **CONDITIONS FOR SUCCESSFUL POLICY TRANSPLANTATION**

It is conjectured (**Conjecture 1**) that in order for trained policies to be non-unique to the agents on which they are trained, agents must have a relative world view of their environment. This is shown to hold true for the surveillance target selection case study. To the author's knowledge, this is the first known attempt to characterise this condition and although not mathematically proven, provides significant insight into **Problem 6 (Policy Transplantation)**. This is a significant contribution as the condition allows policies to be copied into an arbitrary number of agents, thereby removing the need to train large swarms and thus avoiding the significant compute and time costs associated with such training. In short, by limiting specialisation within the learned solution, scalability can be greatly increased.

► **MEASURING THE BREAKDOWN OF EMERGENT BEHAVIOUR**

Chan [Cha11] introduced a metric to indicate the presence of emergent behaviour in a multi-agent system based on the interaction between individuals. The method requires the collection of interaction statistics of the entire system. This approach is built upon to characterise the 'amount' of emergent behaviour based upon the **Autocorrelation Function (ACF)** of successive states of a single agent. The quality of the solution is shown to correlate with the proposed metric and in doing so provides insight into **Problem 5 (Breakdown of Emergent Behaviour)**. The approach is shown to find non-random state correlations for a modified version of Reynolds' seminal flocking algorithm as well as the algorithm proposed in **Chapter 4**.

1.3 PUBLICATIONS

The following publications were produced during the course of my doctoral candidature:

► **CONFERENCE PUBLICATIONS**

- **Bradley Fraser** and Robert Hunjet. "Data ferrying in tactical networks using swarm intelligence and stigmergic coordination". In: *2016 26th International Telecommunication Networks and Applications Conference (ITNAC)*. Dunedin, New Zealand, Dec. 2016, pp. 1–6, see **Chapter 4**.

- Robert Hunjet, **Bradley Fraser**, Thomas Stevens, Laura Hodges, Karina Mayen, Jan Carlo Barca, Madeleine Cochrane, Ricardo Cannizzaro, and Jennifer L Palmer. “**Data Ferrying with Swarming UAS in Tactical Defence Networks**”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Australia: IEEE, May 2018, pp. 6381–6388, see [Chapter 4](#).
- **Bradley Fraser**, Andrew Coyle, Claudia Szabo, and Robert Hunjet. “**Adaptive Swarm Control for Mobile Resource Placement in Wireless Ad-Hoc Networks**”. In: *2019 IEEE 20th International Symposium on “A World of Wireless, Mobile and Multimedia Networks” (WoWMoM)*. June 2019, pp. 1–6, see [Chapter 5](#).
- **Bradley Fraser**. “**Data Ferry Flocking for Bulk Information Transfer Under Ferry Buffer Constraints**”. In: *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*. Nov. 2018, pp. 1–6, see [Chapter 5](#).
- **Bradley Fraser**, Claudia Szabo, Andrew Coyle, and Robert Hunjet. “**Leveraging emergent specialization in a heterogeneous multi-role swarm control architecture for positional-based UAS missions**”. In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. Canberra, Australia: IEEE, Dec. 2020, see [Chapter 6](#).
- **Bradley Fraser** and Giuseppe Laurito. “**Train Small, Deploy Big: Do Relative World Views Permit Swarm-Safety During Policy Transplantation for Multi-Agent Reinforcement Learning Problems?**” In: *AI 2020: Advances in Artificial Intelligence*. Springer International Publishing, 2020, pp. 269–280, see [Chapter 7](#).
- **Bradley Fraser**, Claudia Szabo, and Robert Hunjet. “**Simulating the effect of degraded wireless communications on emergent behavior**”. In: *2017 Winter Simulation Conference (WSC)*. Dec. 2017, pp. 4081–4092, see [Chapter 8](#).

► JOURNAL PUBLICATIONS

- **Bradley Fraser**, Robert Hunjet, and Andrew Coyle. “**A swarm intelligent approach to data ferrying in sparse disconnected networks**”. In: *Int. J. Parallel Emergent Distrib. Syst.* (Oct. 2017), pp. 1–22, see [Chapter 4](#).
- **Bradley Fraser**, Andrew Coyle, Robert Hunjet, and Claudia Szabo. “**An Analytic Latency Model for a Next-Hop Data-Ferrying Swarm on Random Geometric Graphs**”. In: *IEEE Access* 8 (2020), pp. 48929–48942, see [Chapter 5](#).

The reader is directed to [Appendix A](#) for signed Statements-of-Authorship for each of the publications listed above.

1.4 AUTHORSHIP NOTES

The publications where I am listed as the first author indicate that the idea generation, analysis and manuscript preparation were primarily performed by me with guidance from the remaining authors.

The publication [Hun+18] consisted of two related but disjoint pieces of work, one of which I did not contribute to. Of the work in that publication related to this dissertation (see Section 4.3.4), I solely developed the agent code based on my previously published work and performed the simulation and emulation of the physical experiment. However, the implementation relies heavily on the lower-level control software, *Hardware Abstraction and Interface Layer (HAIL)*, developed by Mr Thomas Stevens *Defence Science and Technology Group (DSTG)*. The implementation is described in the graphic shown in Fig. 4.10 which is attributed to Thomas and taken from [Hun+18]. Other members from *DSTG* provided and maintained the physical aircraft. Physical experimentation, analysis and manuscript generation were performed with assistance from *DSTG* colleagues.

Chapter 4 describes work in which a swarm intelligent approach involving pheromones is used to effect equally-spaced ferry nodes around a given polygon. An extension to this work that introduced counter-rotating ferries was undertaken by Major (MAJ) Andrew Hietpas *United States Marine Corps (USMC)* during his Master's Thesis program at the *NPS*, Monterey, CA. and is published in [Hie18]. During this work, I acted as a subject-matter-expert and provided assistance and an approach to linking pheromones together in order to coordinate multiple ferry teams. I also provided an initial codebase for simulation experimentation. MAJ Hietpas' acknowledges my involvement in this work as an extension to ideas generated for this thesis and credits this in [Hie18]. Section 9.1 provides a brief description of this work. Additionally, during our collaboration, MAJ Hietpas and I, along with my thesis supervisors, co-wrote a draft survey paper on data-ferrying techniques and swarm control. In particular, MAJ Hietpas' contribution was the review and classification of general swarm control techniques which appears in an edited form in Section 2.3.

1.5 ACKNOWLEDGEMENTS

There are several people who provided technical assistance with the various projects of which this thesis is comprised, as well as proofing of the thesis itself. It is with pleasure that I acknowledge them here:

- Regarding work in Section 4.3.4, I wish to thank friend and colleague Mr Mathew Elliot *DSTG* for his assistance using the Harlequin radio emulation environment [ES16]. I also wish to thank friend and colleague Mr Thomas Stevens for his assistance with, and incremental improvement of, the *HAIL Application Programming Interface (API)*. Physical experimentation was conducted with the assistance of several team members from the *UAS* Group of *DSTG*; I thank them also.
- Regarding work in Section 7.2.3, I wish to thank Mr Giuseppe Laurito and Mr Philip Moss *DSTG* for assistance with physical implemen-

tation and experimentation. I also thank **Warrant Officer (WOFF)** Wayne Newbery **RAAF Active Reserve (RAAFAR)** and Headquarters, Air Warfare Centre **Royal Australian Air Force (RAAF)** for facilitating flight approval at the **RAAF Woomera Test Range (WTR)**, and No. 453 Squadron **RAAF** for facilitating flight approval at **DSTG** Edinburgh.

1.6 THESIS ORGANISATION

The succeeding chapters of this thesis are organised as follows: **Chapter 2** presents a literature review on data-ferrying, swarm control, emergent behaviour and policy-transplantation in multi-agent learning. Following this literature review, a gap-analysis is performed leading to the research problem-set. **Chapter 3** provides some background on **DRL**, **Random Geometric Graphs (RGGs)** and modelling of wireless propagation and networks.

Chapter 4 begins the technical chapters of the thesis introducing a novel static-path data-ferrying algorithm. **Chapter 5** moves into swarm coordination for dynamic-path data-ferrying under variable graph connectivity conditions. This approach is then expanded into multi-role missions in **Chapter 6**. **Chapter 7** and **Chapter 8** then look at specific scalability issues related to swarms. In the former, a condition on the agent observation vector associated with **MADRL** is proposed that allows learned policies to be copied into additional agents, thereby removing the need to train large swarms. In the latter, emergent behaviour generation is examined under contested communications conditions. The investigation is facilitated by a novel metric for the quantification of emergent behaviour strength. **Chapter 9** makes final remarks and suggests future work directions before drawing the thesis to a close.

2

Literature Review

- **SYNOPSIS** This chapter provides a literature review of relevant topics to the thesis. In particular, methods of coordination of autonomous resource-nodes for transferring data between user-nodes of a serviced wireless network are studied in depth. Based upon the review, the chapter concludes by identifying open problems that the subsequent technical thesis chapters target.

2.1 DATA-FERRYING AND REMOTE SENSING

Connectivity is fundamental to data internetworking. Traditional wired networks typically only suffer connectivity issues due to hardware failures or misconfiguration, as their intrinsic data connectivity is endowed through physical connectivity. Wireless technology has changed this traditional networking landscape by allowing nodes to communicate via the electromagnetic spectrum. Ad-hoc Networks have added to the flexibility of wireless networks by providing participating nodes with the means to act as routers and forward data to their destination; removing the need for centralised control. Adding a mobile capability to these nodes further enhances flexibility by allowing them to move with their own purpose and without regard to the underlying network topology. These networks, known as **MANETs** [Toh02], are ideal for deployment where infrastructure does not exist or has been destroyed. However, this flexibility comes at the cost of decreased network reliability since such networks can easily break. Networks may fragment due to large physical distances between nodes, harsh transmission environments and node failures. They may even be intentionally deployed in such a state as is the case in sparse **WSNs**. Such networks can however, become *time-connected* through the use of data-ferrying.

Data-ferrying is considered the physical transport of electronic data to user-nodes by one or more intermediary nodes. During the ferrying process, data that are generated by user-nodes that do not have an immediate route to their destination are stored for later transmission, rather than dropped. This has become known as the ‘store-and-forward’ paradigm and is fundamental to the concept of **Delay-Tolerant Networks (DTNs)** [Fal03].

While a seemingly simple concept, the design of routes for data-ferries in **DTNs** faces many difficult challenges; not the least significant of which is that finding a shortest tour that visits all vertices in a graph exactly once, known

as the **Travelling Salesman Problem (TSP)**, is **Non-deterministic, Polynomial time (NP)**-hard in complexity [App06]. Furthermore, depending on the application, route designs may also need to account for:

- User-node movement;
- Data timeliness and storage constraints;
- User-node priority and fairness;
- Data-ferry organisation and cooperation; and
- The electromagnetic transmission environment.

These challenges have led to widely differing approaches to data-ferrying route design. In addition, unprecedented access to unmanned systems, particularly **UASs**, means data-ferrying using autonomous platforms is well within the realms of reality. Furthermore, data-ferrying is useful in both civilian and military domains as the backbone for survivable communications within applications including tactical and disaster relief networks, and remote-sensing and surveillance networks. A simple classification system for data-ferrying networks is used to narrow the focus to **Assisted Planned Ferry Systems (APFSs)**. Within these systems, a data-ferrying spectrum is used to group different data-ferrying approaches according to the network environment.

2.1.1 Ferry System Classification

Data-ferrying systems can be immediately classified into categories based on the description of the ferries themselves. Two important characteristics are whether the ferry movement is pre-planned, and whether the ferry is an additional resource to system.

As is shown in Fig. 2.1, if the ferry movement is not planned then the system is describing an **Opportunistic Ferry System (OFS)**. Such networks are extant in the literature and are known as Opportunistic Mobile Social Networks [VY13]. These networks use the opportunistic interactions between their members to enable data exchange. Members simultaneously participate in the network as well as actively ferry data for others. Since the ferries are not controllable, techniques in this space focus on routing at the data level rather than at the platform trajectory level. Correspondingly, the next-hop decision problem is from the perspective of a routing protocol deciding on which ferry a packet is to be forwarded. This generally takes the form of deciding which ferry is most likely to reach the packet's destination sooner. Common approaches include exploitation of known [LDS04] or learned [Lin+12; KS16] mobility patterns of peers, and packet replication [VB00; Bur+06].

We can further divide **OFS** into two categories based on whether ferries are additional resources. A network without additional ferries can be described as an **OFS** and data is transferred purely by the nodes that are members of the social network. A network with additional nodes acting as ferries, though uncontrollable, can be described as an **Assisted Opportunistic**

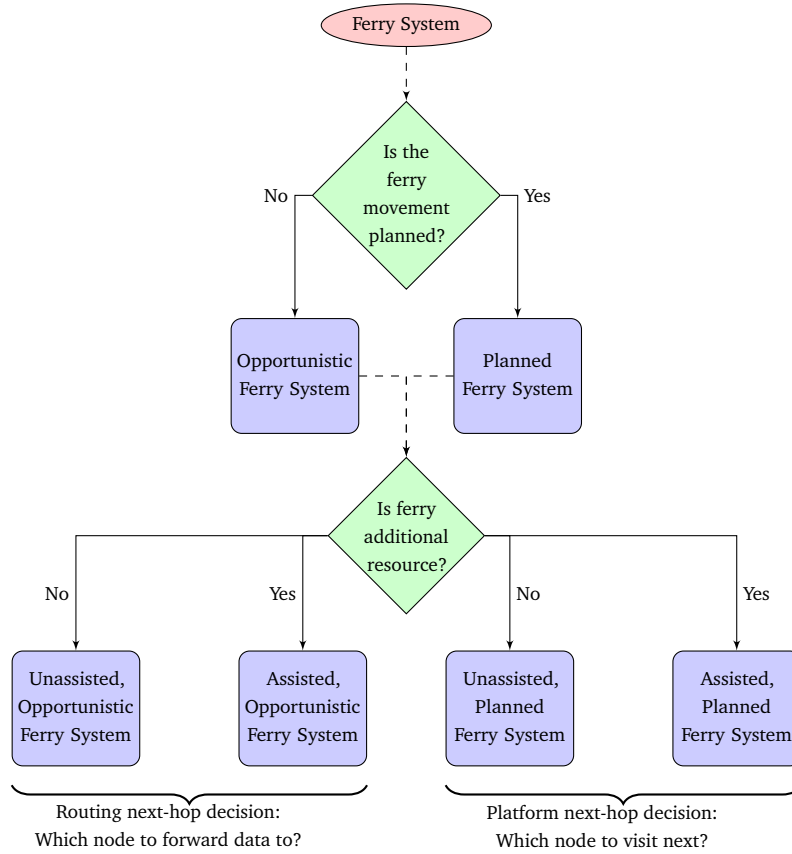


FIGURE 2.1: Classifying data-ferrying systems using ferry characteristics.

Ferry System (AOFS). Both types of opportunistic data-ferrying have been proposed for a variety of applications within the literature, for example:

- Animal monitoring where node-equipped zebras generate their own tracking data and communicate it back to a sink via their own social interaction [Jua+02], and whales communicating their own tracking data between themselves to sinks, but also opportunistically using node-equipped seabirds and buoys as intermediate communication relays [SH03].
- Data dissemination in MANETs where highly mobile nodes carry data between different network segments [DFL01; RG12], and the use of existing network infrastructure to store-and-forward messages for nodes without an immediate route [RTO08].
- Traffic monitoring and the sharing of vehicular data with [YK09; ZC08; AOS08; Bon+16] and without [DNP04; Shi+06; LW08] the assistance of relaying entities such as buses and networked traffic lights.

For sparse networks due to large geographic distances, the assisted, opportunistic approach is favoured and has been proposed for rural connectivity using buses and cars [Guo+06; GK07; Mar+11; Liu+15] as well as health care [PFH04] using boats. This is to be expected when the social network is not dense enough to cover vast areas.

There is a large body of literature covering opportunistic networks and routing within DTNs, and surveys on these topics can be found in [PPC06; KKZ11]. There are also many studies on APFSs but relatively few proposals for the unassisted version. This is not unexpected as it represents a disruption to a member node(s) activity while it performs the exclusive role of a data-ferry. Some works in this category include:

- Identifying nodes in a network that can act as ferries based upon their movement patterns [Pol+11].
- Ferrying data for other nodes using the smallest detour (least inconvenience) [LR00].
- Selecting a ferry successor from regular nodes in the event of a failure [Yan+05].
- Recruiting regular nodes to assist an overwhelmed ferry [XLP12].

The most relevant works to this thesis are those where the ferries are both controllable and are additional resources. The corresponding next-hop decision problem is from the perspective of a ferry deciding which user-node to visit next. There is a substantial amount of research in this area for both WSN (see the survey papers [DFDA11; Aln+15]) and MANET contexts; and the range of target problems and their proposed solutions can be very different.

2.1.2 Data Ferrying

There are several terms used to refer to an entity whose purpose is to carry, collect and deliver data on behalf of other nodes in a network. One of the earliest of these is *MULE* or *Mobile Ubiquitous LAN Extension* [Sha+03]. It was introduced in the context of WSNs and is still in usage there, commonly in the lowercase, noun form *mule*. In this seminal work, data-ferrying feasibility using mules was explored with ferries moving according to a random walk. Around the same time, the term *message ferry*, or simply *ferry*, was introduced in the context of MANETs [ZA03]. Less common terms include Data Carrier [Bon+16], Flying Witness Unit [San+17], Hawker [Mar+11], Mobile Collector [HN08], Mobile Host [LR00], Mobile Sink [Gao+16], Routing ferry [LJ11], Store-Carry-Forward Node [Faw+17], Pigeon [GLQ08] and UAV [Che+07; ST10].

Reference Terminology

Given the various definitions for data-ferrying found in the literature, it is convenient to provide concrete definitions for further use within this thesis. Note that some terms have been appropriated from the literature. The following terms are henceforth used with the stated definitions:

- User-node: An entity which is participating within the network and may produce data bound for another node where there is not an immediate route. Nodes may or may not be mobile and include sources, sinks and sensors from WSNs and nodes from MANETs.

- Data-ferry (or ferry)[:] Is an entity with the physical capability and (primary or secondary) mission to transport data between disconnected user-nodes.
- Data-ferrying: The physical transport of data from one node to another. This definition is used to refer to the general physical transport of data regardless of the underlying purpose of the data. Data-ferrying can be broken into two main types:
 1. Data Gathering (or Collecting): A data-ferrying behaviour where data transport is predominantly all-to-one, such as is found in **WSNs** with a mobile collector and sink.
 2. Message ferrying: A data-ferrying behaviour where data transport is predominantly one-to-one, -many, or -all such as is found in traditional networking unicast, multi-cast or broadcast traffic.

In the surveyed literature, it is generally assumed an appropriate **DTN** protocol is implemented at the networking layer to support the store-and-forward routing of data and the definitions of **DTN**-related terms remain unchanged. An example of such a protocol is presented in [NRG14].

The Data Ferrying Spectrum

The above definitions propose that different terminology should be used based on the data-ferrying application. However, these applications and their proposed solutions are not discrete, and instead lie on a continuous spectrum of problem definitions. Figure 2.2 presents a data-ferrying *spectrum* moving between the two major applications found in the literature: data gathering on the left and message ferrying on the right. This continuous view of data-ferrying unifies the seemingly disjoint and independent research that has evolved in the field of route design in **WSNs** and **MANETs**.

An important characteristic when deciding upon where particular works lie on the spectrum is examining the conflicting objectives a ferry faces when deciding on the next hop. This can be governed, for example, by:

- Traffic Flow: On the left, traffic can generally be described as all-to-one. This problem is conducive to static, pre-planned ferry routes since all data has the same destination and is generally not time sensitive. Data latency can then be minimized by minimizing ferry travel time, leading to shortest-tour problems. On the right however, there may be multiple traffic flows and an optimal route for one flow may not be for another.
- Visit Priority: In message ferrying traffic flows, every message in the ferry buffer contributes to a particular destination's priority. The ferry must somehow resolve which destination to visit next given it may have messages to deliver to many other nodes. The decision is made even harder when individual messages have expiry times. There are cases of visit priority in data gathering as well. In particular, sensor nodes may generate data at different rates or have varying buffer sizes leading to unequal node importance.

Data Ferrying Spectrum			
		Data Gathering	Message Ferrying
Context	WSN	MANET	
Time Sensitivity	Low	High	
Traffic	All-to-one	One-to-one One-to-many	
Flows	One	Many	
Next Hop	Static	Dynamic	
Planning	Pre-planned	On-the-fly	
Control	Centralised	Distributed	
Techniques	(approx) TSP, TSPN, k-TSPN, k-PCPN Optimisation Graph Theoretic Scheduling	Prize-Collecting TSP Online Learning	Data Inspection (PO)MDP Self-organisation
Designed For	Shortest Path Minimum Energy	Minimum Latency	

FIGURE 2.2: The data-ferrying spectrum moving from data gathering to message ferrying.

This generalized data-ferrying spectrum was developed through observations of the literature and outliers may be found. Nevertheless, we shall use this continuous view of data-ferrying to guide our discussions on surveyed problems and their proposed solutions. Hence, the following sections group data-ferrying research into their respective positions on the spectrum.

2.1.3 Route Planning to the Left of the Spectrum

To the left of the data-ferrying spectrum, solutions are predominantly pre-planned, static and generally solve the data gathering problem. If the ferry is constrained to simple uniform paths, then multi-hop forward schemes become important. If full ferry control is allowed, then problems generally aim to find shortest tours for ferries to maximize overall system efficiency. However, individual message latency is not critical since data is generally not time-sensitive.

Uniform Paths

The simplest data-ferrying paths are those where the path is *uniform* in the sense that the ferry moves in some simple trajectory, such as a straight line or curve. Such a path is also considered to be *static* because it does not change from one ferry *round* to another. These static, uniform paths are generally combined with multi-hop data forwarding schemes since the ferry path may not intersect the communication region of all nodes. For example, the works in [JSS05; GZD09; PG16; SKS16] all propose algorithms where ferries move along a straight line path through a WSN field to collect sensor data. Various issues are explored including:

- Load balancing between two available ferries based on hop-count [JSS05].
- Throughput improvement by multi-hopping data in such a fashion that more data is forwarded to single-hop neighbours of the ferry that

will have more contact time [GZD09].

- The ‘hotspot’ problem where sensor nodes that are closer to a sink deplete their energy reserve faster because they spend more time transmitting. To address this problem, clusterhead-based forwarding to a single ferry routed through the center of the field is proposed in [PG16]. Likewise, [SKS16] investigates a clusterhead approach where data is routed to a ferry on the left side of the WSN field on even rounds and to a ferry on the right during odd rounds.

When all nodes lie on a line, such as in linear WSNs, the ferry route is also that line. However, speed control is still available to the ferry. The effect of different speed policies for data gathering in such linear WSNs is analyzed in [Jaw+13]. Rather than using straight lines, [AO+15] proposes to use ferries that move in concentric circles in a WSN field. Ferries opportunistically forward data between themselves and WSN sinks as they travel on the circular, uniform path.

If the ferry route is known in advance and nodes are capable of movement, they may augment their own position to intercept the ferry during its regular route. This has been proposed in the seminal Node-Initiated Message ferrying algorithm [ZAZ04]. Alternatively, in the ferry-Initiated version, out-of-band communication may be used to ask the ferry to deviate from its uniform path to intercept the node in order to exchange data.

The proposal in [LSS13] allows deviation from fixed paths by introducing a weighted cost function. It is used to balance the importance of a minimal distance ferry path and minimal transmission energy usage. In this problem, nodes are laid out on a grid, each sensor can communicate with its nearest neighbours and multi-hopping information is allowed. If minimizing the ferry route is weighted heavily, then the ferry moves from its starting position to its end position in a straight line and all nodes will multi-hop their data to the ferry. Conversely, if minimal transmission is weighted heavily, the ferry will visit each individual node, removing the need for any multi-hop transmissions at all but increasing ferry movement. Although the ferry movement is increased in this extreme case, the optimization still rewards minimal ferry movement, leading to a shortest-path problem. We use this as a jumping off point to TSP-based problems in the next section.

Travelling Salesman-based Formulations

The TSP is a particularly useful formulation for data gathering for obvious reasons but other graph-theoretic problems have also been used in this space. For example, the Route Inspection Problem (RIP), where a shortest-tour is desired that covers a (sub-) set of edges is used in [Wan09]. There, the RIP is formulated for a ferry to visit all nodes while being constrained to roads on a street map. Solutions to these graph-theoretic formulations, given that they are derived from the NP-hard TSP, are also, in general, NP-hard. This is shown in [ZA03] where the original Message Ferry Route (MFR) delay problem is equated to the TSP in complexity. In this seminal work, the proposed solution addresses message delay by generating a shortest tour TSP route that is locally optimized with the well-known 2-opt and 2H-opt

heuristic techniques [Ben92] to reduce average message delay. The authors then consider increasing the bandwidth of the system by increasing the amount of time a ferry is in bidirectional communication with nodes. They extend their work in [ZAZ05] where four algorithms for static route design are proposed:

1. The Single Route Algorithm uses one ferry and the route is generated as above.
2. The Multiple Route Algorithm uses a greedy heuristic to initially assign nodes to ferries and the single route method is then used to generate independent ferry routes.
3. The Node Relaying Algorithm federates different routes using nodes that are common to more than one ferry route to act as a relay between them. Other schemes using regular nodes as relays between disparate ferrying routes can also be found in [ZF07; Li+11; MEWP13].
4. The ferry Relaying Algorithm relies on the synchronization between ferries on different routes such that ferries will come into range of one another to exchange data.

In both articles, communication ranges play an important role in the ferry route design.

THE TRAVELLING SALESMAN PROBLEM WITH COMMUNICATION RANGES The traditional **TSP** involves computing the shortest tour between a number of points, where each point is visited exactly once. A modification to the traditional **TSP** considers including a neighbourhood around each point and is known as the **Traveling Salesman Problem with Neighborhood (TSPN)** [DM03]. In the context of this article, the neighbourhood can be considered as a communications region, usually modelled as circular disc (in two dimensions) with range (radius) r . Nodes are considered to be able to communicate when the distance d between them satisfies $d \leq r$. This is exploited in [LWC13] where the next ferry location is chosen such that the ferry can communicate with as many nodes as possible while still being the shortest tour. Regions with more intersecting nodes are hence more likely to be chosen. The associated **TSP** can then be solved to find the ferry route.

The **TSPN** problem formulation has been shown to work in practice in [Isa+12]. There, a ferry gathers data from disconnected acoustic sensors for centralised event triangulation. Node neighbourhoods are also used in [SG11] where a complete graph (a graph where every pair of vertices is connected by a unique edge) representation is created. Edge labels indicate which user-nodes a ferry would be able to communicate with at some point during the edge traversal. For example, if an edge intersected the communications disc of nodes $\{1, 2\}$ moving from vertex S to vertex 2 , then edge $e(S, 2)$ would receive the label $L(e(S, 2)) = (1, 2)$. The *Label-covering Tour Problem* then involves finding the minimum distance tour such that each label is included in at least one edge traversed by the tour. This formulation adds tour flexibility by being able to remove edges that are not physically traversable, or by adding points, allowing a vehicle to turn at places that are not nodes.

As well as communications range, [BTAZ06] assumes knowledge of a node's mobility model to construct individual ferry waypoints for each node with a given contact probability. A TSP tour is then generated between waypoints. The scheme adds dwell times at each waypoint to improve contact probability which ultimately becomes unity as the dwell time tends to infinity. Another mobility prediction-based scheme is presented in [LJ11] where a graph composed of temporo-spatial meeting points is used to generate ferry routes.

The schemes so far have assumed that communication ranges are fixed. However, the proposal in [Kim+16] assumes that power control of nodes is also available to the route designer. By increasing a node's transmission power, a ferry need not get so close to a node and thus the tour length is decreased. By increasing the power enough, the route design may make use of the intersecting communications range approaches above. However, in this instance the optimization also rewards minimal energy usage by nodes to increase network longevity.

Rather than individual nodes, the proposal in [HA14] explores the federation of disconnected clusters. In this proposal, clusterhead nodes act as a gateway for traffic destined for delivery outside of its originating cluster. Once all clusters have been discovered, a TSP tour is calculated between the clusterheads. The proposal in [VKK16] explores a similar problem. They enhance the aforementioned approach by the addition of 'check point' nodes. Check point nodes act as the cluster-ferry gateway and satisfy the condition of being on the periphery of the cluster. This removes the need for a ferry to travel too far into a cluster before exchanging data and thus decreases the tour length.

The federation of disjoint network clusters using data-ferrying is also considered in [KY13]. It is assumed that each 'segment' has a dedicated ferry working for it and imposes the constraint that the number of ferries, n , is less than the number of original disconnected clusters. This requires allocating all clusters into n segments. Segments are constructed such that segments with more than one cluster form a TSP tour between cluster boundaries. Considering each segment as a graph node, the algorithm then extends certain clusters using a Minimum Spanning Tree approach. Each segment then has one or more clusters that also belongs to another different segment, allowing inter-segment communication. Ferries then follow a TSP tour between clusters in their segment to produce a fully, time-connected network.

When considering multiple ferries, the TSPN becomes the k Traveling Salesman Problem with Neighborhood (k -TSPN) where k is the number of ferries. If the constraint to return to the starting (sink) node is relaxed, for example the ferry itself may be connected to a reach-back network rather than needing to ferry data to a specific node with this capability, then we are dealing with the k Path Cover Problem with Neighborhood (k -PCPN). Both problems have been extensively analysed and solved in the literature with approximation algorithms [Xue+14; Kim+14]. The k -TSPN is also examined in [BTI11] but the data transfer time between a ferry and node is taken into account. To demonstrate the importance of the extra cost, consider the case with two clusters of disconnected nodes that are equidistant from

a base station. A 2-TSPN solution would task one ferry to each cluster to minimize travel distance and hence delay. However, now consider the case where one cluster has many nodes while the other cluster has only one. The ferry servicing the larger cluster would experience a larger data transfer time delay; deploying both ferries to the larger cluster may well outperform the naive 2-TSPN solution in this instance in terms of delay, especially if low data rate radios are being used. As well as data transfer time, ferry buffer capacity may also become an issue in these cases.

THE TRAVELLING SALESMAN PROBLEM WITH BUFFER CONSTRAINTS AND PRIORITIZATION In practice, remotely deployed nodes have a limited buffer space and cannot store data indefinitely. This restriction adds another constraint to the route design process. The single route case, [Cha+11; Cha+14] proposes starting k ferries P/k distance apart around a TSP tour of nodes where P is the length of that tour. By having the ferries move in the same direction and at the same speed, nodes are visited in a uniform time. For multiple routes, [Gao+16] proposes several constant-factor algorithms for cooperative and non-cooperative cases for disjoint route design where the maximum time a node must wait to be visited by a ferry is guaranteed. Hence, buffer constraints can be designed around it.

Sensor nodes may have heterogeneous sampling rates meaning that buffers fill unevenly, leading to visit prioritization. Several iterative, path splitting algorithms have been proposed to deal with this situation [Cha+11; Cha+14; CJ12; MEP12]. The schemes in [Cha+11; Cha+14; CJ12] see the original TSP tour split into loops where each loop is rooted at a prioritized node. This node is then visited more often during one ferry round. The loops can also be rooted at the sink node [MEP12]. These approaches all involve choosing a break edge where this edge is removed and the adjacent vertices are then connected to the priority node or sink.

Two heuristic approaches to finding a feasible route for this problem have been proposed [SRS04]. The first being earliest-deadline-first where the sensor whose deadline is soonest will be visited first. This idea can be extended with ‘ k -lookahead’ by visiting k least deadline nodes on the way to the earliest. The second heuristic proposes a weighted sum where a control variable balances the importance between the amount of time left to visit a node and the cost of getting there. If the time left is weighted heavily, then the algorithm becomes the earliest-deadline-first.

Similar to the constraints imposed by buffer overflow, [VLC05] forces arbitrary Quality-of-Service (QoS) constraints on a TSP route in the form of high priority sender-receiver pairs. In this case, a high-priority receiver must be visited shortly after its high-priority sender. To work with these constraints, the high priority pairs are replaced with virtual nodes, the corresponding TSP problem is solved and the virtual nodes are then expanded back into the high-priority pairs. The high-priority node pairs can be considered as competing traffic flows and to address this in a static, pre-planned way, it is argued that a closed-walk on a graph is likely to outperform a TSP tour in terms of message delay [WL11]. This is because nodes can be visited more than once per round. For example, if a large amount of traffic exists along a particular edge, a closed walk solution that services this edge multiple

times in a single tour will result in a lower average delay. The general **MFR** delay problem is defined for this case and it is shown to be MAX-SNP-hard. A heuristic algorithm is proposed where a route is generated by concatenating a full **TSP** tour with a partial **TSP** tour where one or more nodes are skipped based on traffic requirements. These examples move us along the spectrum as it demonstrates the increasing importance of (competing) traffic patterns as we move from data gathering to message ferrying.

2.1.4 Route Planning for the Middle of the Spectrum

A common characteristic in this regime sees a ferry follow a predetermined path that is recalculated after each round. Learning methods in particular, exhibit this approach. Ferry routes can be planned in advance and improved upon during deployment as the environment is observed. This is the approach taken in [CWF14] where a Gaussian learning process is used to iteratively adapt an initial **RF** model. Using the (adapted) model, a trajectory is plotted that maximizes data throughput at each node. The model is then updated per ferry round. Alternatively, the **RF** model can be removed completely and have the ferry generate its own path based on **RL**. This is demonstrated in [Pea10] for two nodes and multiple nodes in [PB10] where the ferry route is altered per round based on newly acquired knowledge. The reward function can also be modified to incorporate other objectives, for example, to preserve sensor node energy use [PB12].

Data Gathering

In the previous data gathering approaches, routes remain fixed once they have been planned and node priority was addressed by incorporating multiple visits to priority nodes in a single ferry round. A more flexible approach is to alter the ferry route after each round based on some metric. These problems are related to the Prize Collecting **TSP** where the objective is to find a tour that visits a subset of vertices such that the length of the tour plus the sum of penalties associated with vertices not in the tour is as small as possible [Bie+93]. For example, [LES14; AC15] both use a value-of-information type metric to decide upon the next node subset that the ferry should visit. Rather than focusing on the amount of data collected, this approach also takes the data quality (for example age, position in **Area-of-Operations** (AO)) into account. Signal processing techniques have also been used in this space to direct ferries to sensors with more significant data [Zho+17].

A similar dynamic-per-round approach [HB08] adopts a **Markov Decision Process** (MDP) and **RL** strategy to minimize overall latency for a ferry when gathering data. After each round, the ferry decides which, and in what order, nodes should be visited by using (known or predicted) knowledge of node buffer states.

Message Ferrying

Rather than building a **TSP** tour around a value-of-information metric or node buffer state heuristic, in message ferrying, the visit subset can be determined by the outgoing traffic. That is, a data gathering ferry generally

doesn't have data to deliver, so visit order and priority is determined by the node rather than the ferry. In message ferrying however, a ferry will have data to deliver so visit order and priority is determined by the messages already in the ferry buffer, that is, outgoing traffic. For example, the Homing-Pigeon message ferrying scheme HoP [GLQ08] considers the problem where each disconnected node has a dedicated ferry (pigeon) to deliver messages. When tasked, a ferry visits nodes for which it has data using a TSP approach. Similarly in [HAG14], ferries deliver data between clusterheads in disjoint clusters of a network. But rather than naively travelling between clusters on a shortest-tour, it first analyses outgoing traffic statistics of the cluster provided by the clusterheads. These statistics are then used to create a minimum delay, closed walk for a set number of subsequent ferry rounds. This change in perspective from visiting a node to receive information to visiting a node to send information moves us further along the spectrum to approaches where only the next-hop is planned.

2.1.5 Route Planning to the Right of the Spectrum

Route planning to the right of the spectrum is characterized by dynamic next-hop algorithms that try to minimize data latency. They generally solve the message ferrying problem where all nodes are sources and destinations of data packets. In the seminal work [WL10], the authors argue that the original MFR problem ([ZA03] from Section 2.1.3) is not appropriate for cases where traffic is unpredictable. Instead, they define the **Dynamic Message Ferry Delay (DMFR)** problem where message sizes are considered in the latency minimization problem. It is observed that: this problem is **Max Strict NP (MaxSNP)**-Hard, a static route cannot be an optimal solution, and the key to finding a suitable route is to determine the next hop. The DMFR problem is equated to the Job Scheduling Problem and the shortest-process-time-first heuristic is proposed as an approximate solution. The heuristic chooses the destination node according to the ratio of total message size to the time to reach the destination. The dynamic inspection of traffic within the ferry buffer is a common technique found on the right of the spectrum, and we continue this discussion below.

Traffic Inspection

On the left of the spectrum, buffer constraints of nodes were used to inform a TSP tour such that buffers would not overflow. From this, node priority emerged when buffers filled unevenly. As we move to the middle of the spectrum and ferry routes became more dynamic, node buffers are again used to inform a TSP tour per ferry round to avoid overflows. Here, on the right of the spectrum, ferries inspect their own buffer of packets on-the-fly to inform the next-hop destination decision. For example:

- The function

$$V_i = \frac{(W_m M_i) \cdot (W_t T_i)}{W_d D_i} \quad (2.1)$$

from [PLX12a] is used to determine the value of visiting node i next, where M_i , T_i and D_i are the normalized number of packets destined

for node i , the normalized waiting time for packets destined to node i and the normalized distance to node i respectively. W_m , W_i and W_d are the corresponding weights such that $W_m + W_i + W_d = 1$. The node with the largest value is then selected as the next destination. The ferry thus decides on the next destination by taking into account the amount of traffic for a particular node, the cumulative waiting time and the travel distance.

- The weighted traffic inspection function in [SMT13] is comprised of the number of waiting messages for a particular destination, the number of times a destination node has been visited and the distance to it. The inclusion of the destination node count counteracts ‘node starvation’ where a node is never visited if it is not a recipient of any traffic, even though it may have traffic to send. In this work, an optimization routine was performed to find the best performing weights for various traffic profiles.
- The heuristic approaches such as ‘nearest destination first’ or ‘maximum message count destination first’ from [Wu+12] use waiting messages to determine the value on visiting the next destination. In this scheme however, only a subset of waiting messages are used in the value calculation with the subset chosen based on a message’s time-to-live.
- The approach in [CY05] simply selects the destination with the earliest deadline first as having the most value. We saw a similar scheme on the left side of the spectrum where it was used to calculate a feasible ferry route in a pre-planned, centralised way. Here, the calculation is performed greedily at each hop. The pre-planned approach also proposed a lookahead method where non-critical nodes would be visited along the way to the earliest deadline node. In the dynamic approach here, the authors propose to use a geographical elliptical zone from which to select possible lookahead nodes. In this scheme, a ferry constructs an ellipse using its current position and the next earliest node position as foci with the length of the semi-major axis equivalent to the distance the ferry can travel before the next deadline expires. Other nodes within the defined ellipse are then checked to see if a detour to one of these nodes is feasible before the critical node expires. In later work, it is acknowledged that this ‘Elliptical Zone Forwarding’ method does not account for critical messages that are awaiting to be picked up and a modified scheme is devised [CY07].
- The Partially-observable MDP (PODMP) model approach in [Ma+11] examines the ferry buffer for the number of waiting messages, their residual lifetimes and destination distances to assist in decision-making when transitioning from one cluster to another. The authors also provide PODMP ferry control policies that maximize throughput for cases where: node locations are unknown [Liu+10] and with multiple-cooperating ferries [HSL11].

- The Deficit Round Robin approach from [MAZ11] uses a technique borrowed from flow scheduling in routers where each traffic flow accumulates credits in each discretized round. When a flow queue is visited, its credits are reduced according to the amount of data that was sent. Analogously, if each flow is a node waiting to be visited, then a ferry should only traverse the required distance once enough credit has been earned.

In many of these dynamic next-hop schemes, ferries are making decisions locally without knowledge of other ferries that may be in the system, thus coordination between them is difficult. However, global emergent behaviour properties and self-organization can be observed in systems with many locally-acting individuals.

Self-organization

When deployed, ferries may come into contact with each other and there is an opportunity for ferries to influence their peers' behaviour. Given particular interaction rules, self-organisation between peers may arise as a global emergent property. For instance, the swarm coordination algorithm in [ST10] uses multiple simple rules such as 'steer towards where I haven't been before', 'steer away from other agents' and 'if I have data, steer towards a base-station' to calculate the next trajectory (rather than next hop). When agents do come into contact, they can opportunistically exchange information about discovered user-nodes and recent visit history. These simple rules combine to give a macro-level search and message ferry behaviour for search and rescue scenarios.

Stigmergy is a common characteristic of swarm intelligent algorithms when agents cannot directly communicate. Stigmergy is considered the indirect communication between agents through the environment [DBT00]. That is, it is possible to influence the future behaviour of an agent by changing the environment in some way. It is noted in [SMT13] that when multiple ferries follow the proposed weighted traffic inspection function, unexpected localized load-balancing in the network is achieved. The simple act of a ferry collecting messages from a node will influence the decision of another ferry at a later time since there may be less messages waiting in the node buffer. On the other hand, [HMT17] uses stigmergy directly by purposely having ferries leave their visit history at user-nodes for other ferries to consume at a later time. This information can then be combined with other rules to decide on the next hop. The authors expand on their work in [HMT18a] such that ferries also signal their next-hop intention and buffer status with neighbouring ferries. This allows intelligent multiple-hopping of data to neighbouring ferries that may reach data packet destinations earlier. A non-flooding message replication scheme is then added to the ferry data exchange protocol to further reduce packet latency when user-nodes are dynamic [HMT18b].

2.1.6 Discussion

Table 2.1 summarizes the reviewed literature by placing it on the spectrum and highlighting the main features of the proposed solution. Moving down the table (or from left to right on the spectrum) we can see the proposed solutions adapting to the increasing irregularity of the data-ferry route; from uniform and static **TSP**-based paths at the top, to methods of deciding on node subsets for a **TSP** tour per round in the middle, and finally methods to select the best next hop at the bottom. This progression represents movement from **WSN**-based, single flow, all-to-one traffic with low time-sensitivity to **MANET**-based, multiple flow, many-to-many traffic with high time-sensitivity.

TABLE 2.1: Reviewed literature in the context of the data-ferrying spectrum. Spectrum Position can take a value of L, M or R indicating the article sits either to the Left, Middle or Right respectively.

Article	Spectrum Position	Features
[JSS05; GZD09] [PG16; SKS16]	L	uniform path, multi-hop
[LSS13]	L	uniform path, multi-hop, TSP
[Wan09; ZA03] [ZAZ05; ZF07] [Li+11; MEWP13]	L	TSP
[LWC13; Isa+12] [SG11]	L	TSPN
[BTAZ06; Li+11]	L	TSPN, mobility models
[Kim+16]	L	TSPN, power control
[HA14; VKK16] [KY13]	L	TSPN, clusters
[Xue+14; Kim+14] [BTI11]	L	k -TSPN
[BTI11]	L	k -TSPN, data transfer time
[Cha+11; Cha+14] [CJ12; MEP12]	L	TSP, buffer constraints, prioritization equal spacing, path splitting
[VLC05]	L	TSP, prioritization, multiple flows
[WL11]	L	shortest closed walk
[LES14; AC15]	M	per round TSP, value-of-information
[HB08]	M	per round TSP, MDP, RL
[CWF14; Pea10] [PB10; PB11] [PB12]	M	per round TSP, RL for comms
[GLQ08; HAG14]	M	per round TSP, traffic inspection
[WL10; PLX12a] [SMT13; Wu+12] [CY05; Ma+11] [MAZ11]	R	traffic inspection for next hop
[Liu+10; HSL11]	R	POMDP for next hop
[ST10; Hun+17] [SMT13; HMT17]	R	self-organization for next-hop

2.2 DATA-RELAYING

Data-ferrying is suitable in cases where end-to-end connectivity is not achievable. On the other hand however, data-relaying can be used to permanently reconnect a disconnected graph component by acting as a (relative) stationary bridge between two nodes. While this thesis is mostly concerned with disconnected environments, it is acknowledged that connectivity can vary from region to region within the same network and a deployed solution should consider a range of connectivities, rather than one or the other. To that end, data-relaying is briefly reviewed in this section.

A survivability-oriented network management approach is taken in [Hui+11] where data-relaying, also known as range extension, is used. A distributed network algorithm draws a resource-node to a particular location in the network based on graph connectivity metrics. It addresses the problem of making a network harder to break by creating redundant paths. In the distributed approach, raw network state information about a node and its neighbours within a specified hop-count is broadcast to neighbours. Each node then builds up a picture of its local topology which resource-nodes can use to geographically place themselves using optimisation of graph metrics. The approach however does not consider disconnected networks.

Both [KBJ13b] and [Hun15b] consider using swarm-based control to place multiple mobile resource-nodes such that end-to-end connectivity is maintained in a MANET. They use attractive and repulsive type rules based on user and peer resource-node locations to place resources. Similarly to the above, there is no mechanism to effect data transfer once if disjoint clusters cannot be spanned.

A more recent range extension algorithm uses network capacity modelling to generate force-based velocity control of individual resources to appropriately deploy an aerial network above a congested ground network [Par+18]. The work uses resource-node altitude to control the ground coverage and hence the number of connected users to optimise capacity. [Coy18] examines the effect of self-organising resource-nodes on tactical networks utilising current MANET military protocols. Resource-nodes position themselves in the network to increase route capacity between user-nodes. These works are viable approaches to (multi) resource management to improve communication networks where end-to-end connectivity is achievable, but again do not consider disconnected networks.

2.3 CONTROL MECHANISMS FOR SWARM SYSTEMS

Movement and coordination of multiple ferry nodes in a swarm, like other areas of communications, can be divided into centralised and distributed approaches [Bar64]. In this context, these categories define where control decisions take place. In centralised swarms for example, an optimizing agent determines an explicit path for each ferry. Conversely, distributed swarms use distributed techniques to effect global change in the swarm. The intelligence of a centralised system resides in the optimizing node(s), but the intelligence of distributed systems resides in each agent and their

interactions [Sau+09]. The following subsections detail research on the different forms of swarm control.

Centralised Control

Centralised control relies on central planners which use collected data to develop a global solution [BAS13]. Centralised swarms are simpler to manage since each node is under direct control, which in turn makes predicting overall system behaviour simpler. These two qualities make centralised control useful for situations where the swarm behaviour must be tightly regulated. However, this control type also has negative qualities:

- Direct control of each node leads to a fixed solution which is difficult to adjust as the environment changes [Leg+08];
- The central planners require global knowledge of all assets in the swarm, severely affecting scalability;
- The effectiveness of the swarm is directly tied to the operation of the planner;
- The planner represents a central point of failure that reduces the robustness of the entire swarm; and
- A data link from the planning node to the other swarm nodes is required if an update to the planning is needed.

Centralised control can be further subdivided into external centralised control and internal centralised control.

EXTERNAL CENTRALISED CONTROL External centralised systems have the central planner as a system component external to the swarm agents. This approach is popular when **Size, Weight and Power (SWaP)** metrics becomes important such as in platform miniaturization. For example, the micro **UAS** quadcopters in [Kus+13] was reduced to a diameter of 21cm and mass of 50g. To achieve this, processing was done offboard by control nodes and was communicated to the platforms via radio links. In this case, these control nodes served as the swarm's central planners.

This approach is both external since it utilized non-swarm nodes, and centralised since global positioning information about the swarm was available to the planners where processing was done centrally. It enabled the swarm to execute a wide variety of complex formations. However, the approach relies on strict bandwidth, processing and latency requirements; because of this, each central controller was responsible for only 5 aircraft and hence may not be a scalable solution.

INTERNAL CENTRALISED CONTROL Internal centralised control places the centralised control nodes within the swarm. An example of this control type can be found in [Lau15] where a single master **UAS** generated flight paths for slave **UASs**. While this type of control can reduce the distance a control data-link must span, it places the processing burden on the platform which affects **SWaP** requirements. Additionally, it was found that as the swarm size increased, the high latency and bandwidth requirements of centralised

control prevented effective swarm operation. To overcome this, researchers subdivided the swarm into subswarms; each with their own central planners creating a hierarchical, *decentralised* control structure.

Distributed Control

The other broad category of control is distributed control where each UAS independently receives some information and takes an action [Gau+03]. With their distributed communication and control methods, distributed control solutions are scalable because the computation capability is not tied to a single platform. They are also resilient to node losses and reduce the possibility of communication delays. For example, distributed control has recently been used to control a swarm of 50 aircraft where each individual makes decisions based on local and multi-hopped information [Chu+16; Dav+16]. In the following, we discuss two biologically-inspired techniques used to achieve distributed swarm control: flocking and stigmergy [DBT00].

FLOCKING-BASED SWARM CONTROL Flocking behaviour is a form of self-organization and can be seen in some animal species. Flocking of the collective is achieved through the local interactions of individuals and was first modelled in 1987 [Rey87b]. A comprehensive mathematical framework for flocking is presented in [OS06] and has been suggested as a type of swarm control for a variety of applications [KBJ13b; Hun15b; SB15b; Fra18a]. It has also been implemented in [Vás+14] to control a swarm of 10 aircraft. True to biological flocking, each UAS uses only information about nodes that are within communication range to make its next heading decision. While scalable, flocking methods still require regular information updates to maintain the emergent behaviour. This was investigated in [FSH17] where the degradation of the flocking behaviour was observed as protocol information was lost or delayed. To overcome this requirement, stigmergic methods may be used.

STIGMERGY-BASED SWARM CONTROL As with flocking, stigmergy is found in the natural world and is considered to be indirect communication through the environment. In particular, some species of termites and ants demonstrate the use of stigmergy by depositing pheromones to alter the behaviour of their peers [DBT00]. In the case of Argentine ants, shortest-paths are found between nests and food sources since pheromone deposits along a shorter path are renewed more often. This attracts more ants, which in turn, increases the attractiveness of that path [Gos+89]. This process is the basis of the well-known Ant Colony Optimization [DBS06].

Digital pheromone approaches have been investigated for path planning where digital analogies for pheromone depositing, propagation and evaporation are defined [Par03]. It has also been implemented as a platform controller for surveillance applications [Sau+09] where it was found that pheromone control required minimal bandwidth resources. This makes the use of digital pheromones an appealing choice for swarm control. Similarly to centralised control, we can divide pheromone-based approaches into internal and external categories.

Internal pheromones are updated on a pheromone map stored in each asset, that is, they are internal to an individual platform, and are updated as a swarm element observes its environment. This updated information is then relayed to the other swarm members when communications are available. This approach was taken in the aforementioned surveillance application where a heterogeneous swarm of two **UASs** and four unmanned ground vehicles could coordinate among themselves, react to the environment, and accomplish overarching objectives [Sau+09]. A similar approach is also taken in [ST10]. A key advantage of an internal pheromone swarm is that the controller is able to make changes to the map's pheromones to affect the overall behaviour of the swarm [Wal+06]. This capability provides the controller with control similar to centralised swarms if necessary, but the effective implementation of this method relies upon end-to-end connectivity of the swarm.

Thus far, all the swarming techniques discussed require inter-vehicle communication. Whether they rely on global or local knowledge, these connections update each **UAS** about their surroundings. In certain conditions, connectivity between swarm members cannot be guaranteed. The external pheromone method of distributed control, which utilizes indirect communication between swarm members, fills this gap.

External pheromones store information in the environment rather than in the swarm element itself. A direct artificial implementation of this has been implemented with robots containing chemical sensors [Rus99]. A digital approach to this technique is through the use of unattended ground sensors that could be deployed by swarm members (or external nodes) to mark their environment [Par03]. Pheromones may also be stored in other non-swarm network nodes [SMT13].

2.3.1 Discussion

From the reviewed literature, it was found that control mechanisms for physical implementation of autonomous robots could be classified based on where decision processing was done and how much data is available to the platform. Figure 2.3 provides a simple organization for ferry control mechanisms. In general, centralised control systems allow complex behaviour at the expense of heavy processing and bandwidth requirements. Centralised processing can be moved to within the swarm but the physical platform must account for the increased processing load. On the other hand, distributed control techniques such as those inspired by biology (flocking and stigmergy) are likely to be robust and scalable.

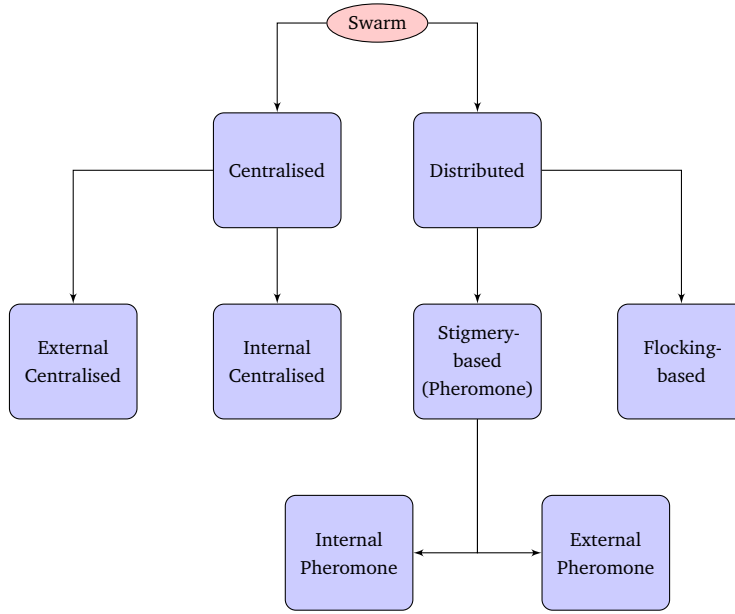


FIGURE 2.3: Classifying platform control mechanisms for swarms.

2.4 SWARMING-BASED ROLE ALLOCATION

One of the earliest formulations of the role allocation problem is known as the **Generalised Assignment Problem (GAP)** [ST93]. As originally posed, it involves scheduling parallel machines to undertake multiple jobs. Each job must be serviced by a single machine where each machine has a certain cost and requires a certain time to process the job. However, an important aspect of performance in multi-agent systems is the capability to handle multiple roles during the coordination of intelligent agents [Abr05]. Thus, the **Extended Generalised Assignment Problem (E-GAP)** was proposed which allows agents to take on multiple roles and switch between them [See+04; Sce+04].

Multi-agent role allocation methods can be broadly categorized into negotiation, threshold-based, broadcast-based, token-based and non-communicative methods [CW11]. In negotiation methods, agents discuss and come to an agreement on the assignment of roles with market-based methods, such as auctions [Dia+06]. In this approach, agents are contracted by a manager agent to fulfil the role based on their expected utility of completing it. Negotiation methods are considered inappropriate for the disconnected environment considered within this thesis as communications cannot be ensured. Much like the very early token-ring **Local Area Network (LAN)** topologies that regulated digital transmission by the possession of a token [IEE89], token-based methods e. g. [Far+05], sees agents only performing a role when it is in possession of a token. While effective at preventing multiple agents from performing the same role, token-based methods are similarly considered unsuitable due to the assumption of regular agent contact.

Threshold-based methods (e. g. [AM02]) and broadcast-based methods (e. g. [MCA05]) are more suitable for the considered environment due to their internal decision-making nature. Threshold-based methods keep track

of variables representing each role in the system and adjust it in some way. Once a particular threshold is met, that role is chosen. Parker’s ALLIANCE architecture [Par98] anthropomorphises this process by referring to the variables as ‘motivation’. Motivation is: increased by sensors that inform the agent a role needs fulfilling; decreased by performing its current role; decreased on being informed that other agents are performing the role; but increased again if they perform it slowly.

Broadcast-based methods are similar to threshold-based methods in that agents independently make role allocation decisions with raw agent state information being continually broadcast (requiring no explicit acknowledgement). Agents can then make their decisions based on their current state information and that of their peers. Role preferences and agent intentions can also be broadcast to prevent multiple agents performing the same role [ACM05].

Non-communicative methods rely on agents estimating peer utility based purely on their own sensed information. RL has been leveraged in this space to decide on when to switch roles due to the difficulty in hand-crafting switching rules [MA03] given large complex state information.

To directly address the GAP described above, [FBB07; Fer+10] propose the **Swarm Generalised Assignment Problem (Swarm-GAP)** algorithm that divides resources according to theoretical models of division of labour in social insect colonies. In the threshold-based method, an agent’s probabilistic response to a task is regulated by its tendency; a metric composed of task specific stimuli and the agent’s capacity to perform it. Specialised agents are therefore attracted to low-stimulus tasks. Communication between agents is based on tokens and is considered reliable. Recently, [Sch+18] proposes improvements to **Swarm-GAP** for **UAS** teams, again assuming perfect communications.

2.5 CHARACTERISATION OF EMERGENT BEHAVIOUR IN UAS SWARMS

First introduced in the context of cellular robotics, swarm-intelligence has been described as non-trivial intelligent behaviour exhibited in unpredictable systems capable of producing order [BW93]. This ‘robot system intelligence’ could be produced by a collection of autonomous, non-synchronized, non-intelligent robots; ultimately achieving a global task. This process is now known as emergent behaviour, and while the definition of this term has been hotly debated [Bos+05; Joh06], it is used here simply to refer to the process that produces the global property that we are interested in exploiting.

One definition of emergence, known as ‘Weak Emergence’, considers the macro-level behaviour of a system as a result of the interaction between micro-level components [Bed08]. Unfortunately, this behaviour cannot be reduced to the behaviour of individual system components in common complex systems [SB17b], or as Bedau describes, the emergent property is a “property of assemblage that could not be predicted by examining the components individually” [Bed08]. Thus, the prediction, identification, measurement and validation of emergent behaviour within complex systems are difficult tasks.

As noted above, the interaction between micro-level components is of particular importance to emergent behaviour generation. Indeed, [Cha11] used agent-based simulation to gather statistics about these interactions noting that, no emergence can arise in a swarm without interaction from its parts. Characteristics about these statistics were examined in complex systems that were known to generate emergent behaviour, as well as ones that do not. This resulted in the discovery that non-random interaction statistics were associated with those examples that generated emergent behaviour. Thus, the method of interaction statistics was a reasonable candidate to identify if emergent behaviour existed in those systems. However, the metric is only useful in a binary sense, and does not indicate the ‘amount’ of emergence present. In order to examine the effect of poor communications on emergent behaviour through the disruption of micro-level interactions, a continuous metric is required. The work presented in Chapter 8 extends the work of [Cha11] into a continuous domain.

There are few works that characterise team or swarm performance with variable communications quality; one of these however can be found in [ZSS17]. It describes an investigation of a multi-robot surveillance task with controlled packet-dropping. In this example, a non-linear performance decrease was found with respect to increasing packet loss. However, this work only measures the algorithm performance, something that Chapter 8 refers to as a *purpose-driven* metric. It does not attempt to measure inherent emergence within the team and analyse the performance with respect to it. This may be an unfair statement for this particular example, since the multi-robot algorithm, which is not described, may not be a swarm intelligent one. In fact, no works could be found that examine swarm performance from both practical and emergent perspectives.

2.6 POLICY TRANSPLANTATION IN MULTI-AGENT LEARNING

Given the static-length nature of neural network input layers, compact data representations of dynamic information must be developed for use in swarm systems due to their dynamic nature. For example, agents may be physically added or removed from the swarm. In the former, an observation vector that explicitly inputs data about agents would not be able to observe new peers. In the later, zero-padding and indicator neurons would need to be added to the observation vector to indicate the presence or absence of peers, and therefore whether or not to ignore their associated data. Even if physical platform numbers do not change, communications are limited in the real-world which leads the same problem when nodes move out of range of each other. To address this issue, the work in [HŠN18] proposes a compact observation model for swarm agents by allocating peers into regions of space surrounding the agent. This model forms the basis of the one used in this article. An earlier example of this approach can be found in [Mor+04] where ultrasonic sensor readings of a mobile robot are divided into groups such as ‘near’ and ‘far’. The work in [Hsu+20] similarly represents other agents by allocating them into a set of categories. However, rather than regions of space, observations consist of a set of belief states. Similar to the space-based formulation, the input dimension is no longer a function of the number of

agents, and therefore an arbitrary number of agents can be represented.

With regards to policy-transplantation and swarm-safety, there are no works that explicitly consider them. However, discussed below are some related work where it has (unknowingly) been used. The authors of [Răd+19] consider the problem of how to transfer knowledge from a single trained agent to multiple agents. In their study, they consider autonomous agents that must navigate safely down a highway. While their main contribution shows that a multi-agent system learns faster and better when seeded with a single-agent policy than a multi-agent system learning from scratch, we focus on another result; the training of one agent and the transplantation into many. They show that transplantation is successful but not unexpectedly, the performance decreases as the number of autonomous agents increase. The performance issue here is from the fact that a *single* agent was originally trained and transplanted into a swarm, not because the policy itself was unsuitable for the new host agents. Performance could be improved by training several, rather than one, agents and then transplanting one of those policies into many. Importantly, they describe a ‘car-centric’ observation vector model which is an implicit world-view that is relative to each car agent.

The second example of policy transplantation occurs in [Li+17]. The authors consider how a multi-agent swarm can learn how to solve a rendezvous problem with limited communications, not from reinforcement, but rather learning from a centralized known solution. Again, policy transplantation in this work is tangential to their main results. They train a swarm on 10 agents and deploy the policy on 100. Similarly to the above, the observation by which agents view their world is a relative one where peer locations are represented in a way relative to an agent’s own position. We take inspiration from this work in our own relative representation.

Both works above employ a relative world-view in an effort to produce a static-length input vector required by neural networks but that also accounts for the agent dynamics involved in swarm and multi-agent systems. While this is true in our case, we also have other static information known in advance in the form of surveillance target locations. With regards to this data, we have the option to represent them in either an absolute or relative fashion. As is shown below, while both options solve the task, only one is swarm-safe during policy transplantation.

2.7 OPEN CHALLENGES

Based upon the review above, this section identifies and describes open problems that are addressed in the technical parts of this thesis.

2.7.1 *Swarming Algorithms for UAS Operations*

As shown above, data-ferrying can be roughly grouped based on whether the ferry path is static or dynamic. In static-path ferrying, all ferries follow the same route and must move at the same speed and direction, and with a specific separation to maximise efficiency. The current and most obvious, solution to this problem is by equally spacing ferries at start-up, which has

been proposed in [Cha+11; Cha+14] (see Section 2.1.3). However, this solution is unsatisfactory should any of the ferries fail. Even if a controlling user-node is introduced to identify and re-orient the ferries, all ferries must pass by it before any separation adjustment can be done. This leads to our first open problem:

Problem 1 (Equidistant Ferry Spacing). *Develop a distributed, equidistant data-ferry spacing scheme on a static-path ferrying route.*

This is a challenging problem given its disconnected nature but is highly relevant to some Humanitarian-Assistance and Disaster Relief (HADR) and defence-based missions where intermittent communications are to be expected.

Dynamic, or next-hop, data-ferrying has ferries dynamically calculate their next destination to visit (i. e. the next-hop) and can be more efficient since they can choose any path and thus be reactive to data traffics. From the surveyed literature on data-ferrying and data-relaying however, there is an underlying assumption of a static user-node network connectivity. No algorithms in Section 2.1 or Section 2.2 attempted to perform both functions or investigated connectivity as a variable, thus our second open problem is to:

Problem 2 (Connectivity-aware Data Transfer). *Develop a graph connectivity-aware, data transfer algorithm that dynamically performs data-ferrying and/or data-relaying as required by the underlying network connectivity.*

Again this is particularly relevant to military-oriented networks that stretch large geographical distances and are therefore likely to experience heterogeneous connectivity states due to localised propagation environments.

Another consideration during data-ferrying is that of the buffer state. This problem is typically considered in the context of WSNs (see Section 2.1.3) where it is the sensors that are of concern, not the ferries. While data storage is now cheap, the adoption of high-resolution sensors and the possibility of being disconnected from other network nodes for an extended period of time, means that data can easily overwhelm ferry buffers. Thus, the following problem exists:

Problem 3 (Buffer-constrained Ferrying). *Develop a buffer-constrained next-hop data-ferrying scheme.*

Until intelligent, edge-processing becomes common-place, raw sensor data will still be required to be transferred back to sink nodes that are capable of processing and acting upon it; making constrained-buffer data-ferrying a relevant and challenging problem.

Some UAS platforms are capable of multiple, possibly heterogeneous, roles. This adds another dimension to the existing spatio-temporal UAS coordination problem and further complicates task scheduling and allocation between resources. The problem is well defined in the literature through formalism of the GAP and E-GAP problems. However, E-GAP, and the GAP on which it is based, are both known to be NP-complete [ST93]. This has led to the heuristic, swarm -intelligent algorithms described in the review. However,

many of them require reliable communications and very few consider how commander intent can be translated into actual behaviour. Thus, this thesis addresses the following problem:

Problem 4 (Multi-role Coordination). *Develop an extensible, multi-role swarm control architecture for autonomous, heterogeneously-capable UAS assets with a mechanism for commander intent with unreliable communications.*

2.7.2 Scalability Issues in Deployment and Design

In the context of control, scalability of swarm algorithms is typically taken for granted; they are by definition, scalable. However, there are other issues associated with the deployment and design of swarming algorithms that affect their scalability. In the deployment space, the sheer number of platforms competing for a shared communications channel may produce packet loss and hence the loss of data required to sustain any desired emergent behaviour. While this is an obvious outcome, the characterisation of how emergent behaviour breaks down in such conditions is currently unknown, leading to the following problem:

Problem 5 (Breakdown of Emergent Behaviour). *Characterise the effect of wireless channel congestion on an emergent algorithm using an appropriate metric.*

Defining and measuring the quality of an emergent algorithm when challenged with protocol disruption will provide insights into the requirements, traffic prioritization for example, for such deployments.

In the design space, the heuristic engineering of emergent behaviour is inherently difficult because the local rules associated with swarm individuals do not map to the collective emergent property in an explainable or predictable way [BDT99]. The alternative to hand-designing these algorithms is to evolve behaviour using an optimisation routine that produces more desirable behaviour over successive iterations; the quintessential example being evolution via natural selection. The recent deep learning revolution has seen the development and success of multi-agent reinforcement learning architectures in this space (for a comprehensive review, see [NNN18]).

Given that time and compute resources typically scale unfavourably with the number of agents, training numerous agents becomes intractable. To avoid this situation, it is possible to use a single trained policy in multiple agents via policy-transplantation i.e. copying of the policy. The first and most obvious issue is that the transplanted policy, having been trained in a smaller swarm, may not perform well in its new environment of a larger swarm. While one of the works described in Section 2.6 has considered this problem already, the conditions leading to *successful* policy-transplantation are currently unknown, leading to the following problem:

Problem 6 (Policy Transplantation). *Determine the requirement for successful agent policy-transplantation within MADRL problems.*

This problem differs from the performance issue described in the review above in that it is possible that a trained agent policy can only be deployed on

the agent on which it was trained; any attempt at transplantation will result in abject failure, not just poor performance. Determining this requirement has the potential to significantly improve training scalability.

With the literature review and gap analysis complete, [Chapter 3](#) provides a brief background on topics used within the thesis.

3

Background and Formulations

- **SYNOPSIS** This chapter provides introductory backgrounds to several topics of importance to this thesis, namely: [Section 3.1 \(Radio-Frequency, Communications Network and Agent Modelling\)](#), [Section 3.2 \(Random Geometric Graphs\)](#), [Section 3.3 \(Multi-Agent Deep Reinforcement Learning\)](#) and [Section 3.4 \(Multi-criteria Decision Making\)](#).

3.1 RADIO-FREQUENCY, COMMUNICATIONS NETWORK AND AGENT MODELLING

In this section, the common network architecture, wireless propagation model and routing model used throughout the thesis, is defined.

3.1.1 Network Model

Let $\mathcal{G}(t) = (\mathcal{N}, \mathcal{E}(t))$ denote a **MANET graph** with node set $\mathcal{N} = \mathcal{U} \cup \mathcal{R}$, which comprises a user-node set $\mathcal{U}$ and a resource-node set $\mathcal{R}$, and an edge set $\mathcal{E}(t)$ at discrete time t defined as

$$\mathcal{E}(t) = \{(i, j) : i, j \in \mathcal{N}(t), i \neq j, f(i, j) = 1\} \quad (3.1)$$

where $f : X \rightarrow B$ maps the arbitrary set X to boolean domain $B = \{1, 0\}$.

3.1.2 Communications Model

Communications between nodes can be modelled at varying levels of fidelity depending on the needs of the investigation. This thesis makes use of two models: a fixed-range step-function model and a **RF**-propagation model.

Fixed-radius Propagation Model

A fixed-radius communications range model consists of a circular (in two dimensions) or spherical (in three dimensions) region with radius r , and origin at the **node's** location. Any other receiving **node** whose position is within this region implies that it can receive traffic transmitted by the sender. That is,

$$f(i, j) := d(i, j) \leq r_i \quad (3.2)$$

where $d(i, j) = \|\mathbf{x}_i(t) - \mathbf{x}_j(t)\|$, $\|\cdot\|$ is the Euclidean norm and $\mathbf{x}_i(t)$ is the Cartesian position of node i at time t . To ensure bidirectional communication

between nodes

$$d(i, j) \leq \min(r_i, r_j). \quad (3.3)$$

Radio-Frequency Propagation Model

RF-propagation is affected by various processes that govern if an RF-signal can be decoded by a receiver. These processes include:

1. Fading: occurs due to variation in the attenuation of a radio signal. During transmission, environmental clutter causes multiple copies of the signal, each with their own attenuation, phase and delay, to be emitted. The superposition of one or more of these copies at the receiver can cause constructive or deconstructive interference and a variation in received signal strength.
2. Path-loss: occurs due to attenuation of radio energy through the propagating medium. Path-loss is caused by diffraction and absorption of radio energy by obstructions in the propagation path but also occurs due to free-space loss. As the range increases, the surface area of an imaginary sphere with origin at the emitter increases and the power decreases proportionally. This drop in power can be conceptually understood as a decrease in flux and is known as the inverse-square law.
3. RF interference: is the unwanted RF energy that is not part of the RF signal. To decode a RF signal, a sufficient Signal-to-Noise ratio (SNR) is required. RF interference contributes to the RF noise reducing the SNR.

To reduce modelling complexity and capture all these processes in a single model, the statistical Log-Normal Path Loss Model (LNPLM) model was developed [Rap02].

For antennas with no directivity, the Free-Space Path Loss (FSPL) is given by:

$$PL_{FS} = \left(\frac{4\pi d}{\lambda} \right)^2 = \left(\frac{4\pi df}{c} \right)^2 \quad (3.4)$$

$$PL_{FS}(dB) = 20 \log_{10} \left(\frac{4\pi df}{c} \right) \quad (3.5)$$

where d is the distance between transceivers, λ is the wavelength of the radio signal (m), f is frequency of the radio signal (Hz) and c is the speed-of-light (ms^{-1}). Note that $d \gg \lambda$ to ensure transceivers are in far-field propagation.

To include attenuation due to clutter in the environment, another term is added to further reduce the signal strength. Assuming transmit and receive gains of 0dB, the received signal strength power P_r is

$$P_{r \text{ LDPL}}(dB) = P_t + 20 \log_{10} \left(\frac{\lambda}{4\pi d_0} \right) - 10\gamma \log_{10} \frac{d}{d_0}, \quad (3.6)$$

where P_r is receive power in decibels referenced to one milliwatt (dBm), P_t is transmit power (dBm), λ is wavelength (m), d_0 is the reference (far-field)

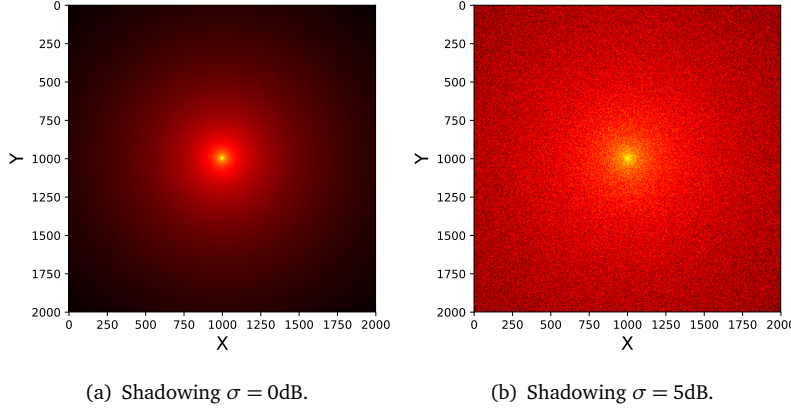


FIGURE 3.1: The signal power over a $2000 \times 2000\text{m}$ field from a point source located at the origin modelled using the LNPLM. With clutter (right), the signal gradient is non-smooth.

distance (m), γ is the unit-less path loss exponent. The signal attenuates with the logarithm of distance between transceivers and is known as the **Log-Distance Path Loss Model (LDPLM)**. By appropriately choosing γ , the equation can model the loss experienced caused by various types of environments. However, since fading is a random process, the **LDPLM** can be improved with the addition of a zero-mean Gaussian random variable giving the **LNPLM**

$$P_{r \text{ LNPL}}(\text{dB}) = P_{r \text{ LDPL}} + \chi_{\sigma}, \quad (3.7)$$

where χ_{σ} is a random variable such that $\chi \sim \mathcal{N}(0, \sigma^2)$ with a standard deviation of σ dB. Along with γ , the selection of σ determines the type of propagation environment. As an example of this model, see Fig. 3.1. Empirical values for various environments and frequencies have been derived [Rap02].

Now, given a model for the received signal strength power, the packet reception function becomes

$$f(i, j) := \text{SNR}(i, j) \geq \text{SNR}_{\text{threshold}} \quad (3.8)$$

where the **SNR** $= \frac{P_r}{n_r} = \frac{P_r}{n_f}$ where n_r is the received noise (dB) and n_f is the noise floor (dB). To account for additional noise and interference $n_r = n_f + \sum_{i=1}^k P_i$, where n_r is the total noise at the receiver and P_i is the power at the receiver from noise transmitter i for k simultaneous transmitters. Note that this calculation is done in linear units. For an in-depth introduction into radio-propagation modelling, the reader is directed to [Rap02].

Bit-Error Rate Modelling

The above level of modelling may be sufficient for many applications, however the model treats all packets equally. To include the effect of varying packet sizes, the **SNR** packet reception model must be translated to **Bit Error Rate (BER)**. To facilitate this translation, a specific modulation scheme must be chosen. Given the prevalence of the **Institute of Electrical and Electronics Engineers (IEEE)** 802.11 Wi-Fi standard, and the readily available closed-form solution to translate **SNR** to **BER**, the **Binary Phase-Shift Keying**

(BPSK) modulation was chosen. BPSK is the simplest form of Phase-Shift Keying (PSK); a method of phase modulation of the carrier wave. BPSK uses two phases and therefore can only modulate a single bit per symbol. As a consequence, it is the slowest modulation found in the IEEE 802.11 standard, but also requires the lowest SNR to demodulate. The closed-form solution of SNR to BER for BPSK is given by [SP07]:

$$\text{BER}_{\text{BPSK}} = (0.5) \text{erfc}(10^{\frac{\text{SNR}}{10}}) \quad (3.9)$$

where erfc is the Complementary Error Function [Wei].

Given a BER, we use Bernoulli trials to decide if a packet is in error. The probability of k successes in n trials with probability p is given by [CD01]:

$$P(k) = \binom{n}{k} p^k (1-p)^{n-k}, \quad (3.10)$$

then assuming we are only interested in the case where all trials are successful, i. e. $n = k$, then

$$P(k) = p^k,$$

and so the probability of packet success is given by

$$P(\text{BSR}, b) = \text{BSR}^b \quad (3.11)$$

$$P(\text{BER}, b) = (1 - \text{BER})^b \quad (3.12)$$

where b is the packet size in bits and BSR is the Bit Success Rate (BSR). Note that P is maximised when there is zero error or zero packet size. Also note that this model assumes that bit errors are independent. Other more complex models that simulate burst error exist, for example see [Gil60]. However, for simplicity, no form of error correction or redundancy is assumed, and packets are dropped with one or more bit errors, making a burst model unnecessary. Figure 3.2 graphs the receive probability against packet size for various bit success rates. As can be seen, the probability of all bits being correct rapidly drops with increasing error rate and packet size.

3.1.3 Routing

Some simulation within this thesis requires the hopping of information across multiple nodes. Many routing protocols for wireless networks exist [TK10]. To approximate best-case routing behaviour, data is instantaneously transferred from sender to recipient if a graph path exists. At each time step, a graph is built using the above models to determine edge connections. The routing path taken is calculated using Dijkstra's Shortest Path [DO59], facilitated by the NetworkX Python package [HSS19].

3.1.4 Discrete-event, Agent-based Modelling

This thesis employs discrete-event, agent-based modelling tools. These tools include Multi-Agent Simulator of Networks (MASON) [Luk+05], custom Python environments similar to Mesa [MK15; KMC20] (a Python analogue of MASON) and OpenAI Gym [Bro+16] compliant MADRL environments. In all cases, the environments are discrete in that time advances in discrete steps, where each step represents some time delta that is scenario-dependent. The

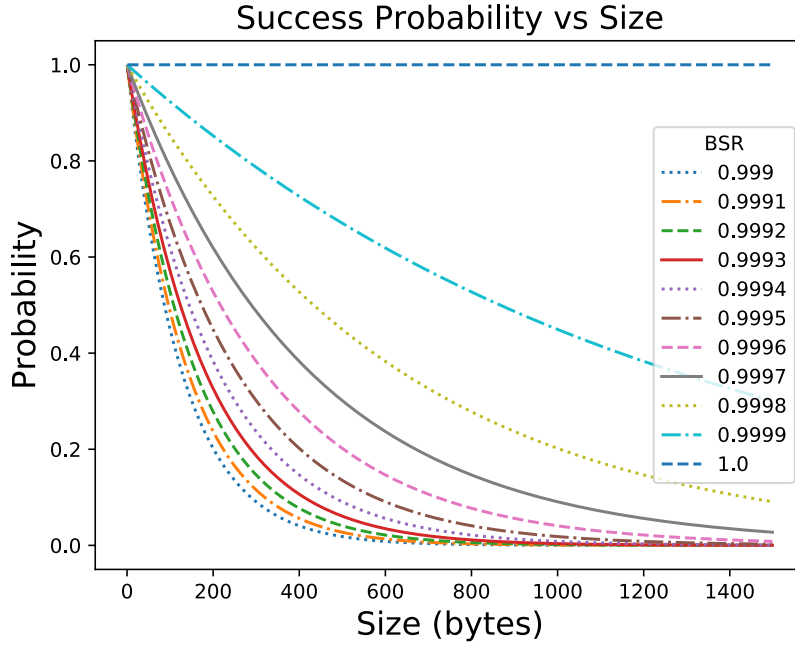


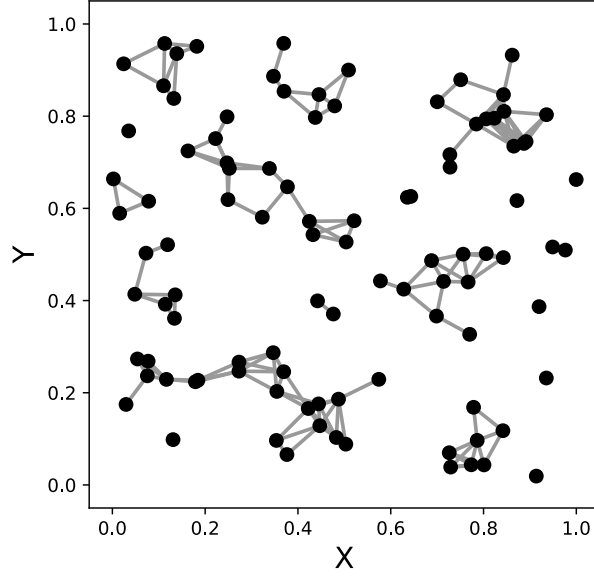
FIGURE 3.2: Receive probability against packet size using Bernoulli trials.

modelling environments are also agent-based. Agent-based modelling simulates simultaneous execution of, and interactions between, individual agents with the aim of measuring the resulting system behaviour and outcomes over time [CH12]. In the case of MADRL, the discrete-event, agent-based environment provides a straightforward approach for learning where the system is initially in some state, the agents then take some action, which in turn moves the system to another state.

3.2 RANDOM GEOMETRIC GRAPHS

As originally described by Gilbert, vertices of a RGG are generated according to a Poisson point process with vertex pairs being connected via an edge if they fall within a fixed range r [Gil61]. As such, RGGs are a useful tool for the investigation of wireless ad-hoc networks and are utilised in this thesis for that purpose, see Chapter 5. RGGs represent networks with the communication range of nodes being approximated by a circle of radius r but RGGs with generalized connection functions have also been developed to allow for other communication models [DG16]. However, for simplicity, we assume the standard disc model. Further to their usefulness in modelling wireless ad-hoc networks, RGGs have well understood connectivity properties [Pen03]. These mirror some of the properties of tactical military and deployed emergency services networks, and so are also useful in better understanding some properties of these networks. In particular, these networks can take on various connectivity states making RGG a useful modelling tool and are formally introduced below.

Let $\mathcal{G}(n, r)$ represent a graph with a sequence $\mathcal{X} = \{x_1 \dots x_n\}$ of independent and uniformly distributed points on $[0, 1]^d$ where d is the space dimension, then given a fixed $r > 0$, connect two points if their $\mathcal{L}_p$ -distance

FIGURE 3.3: A 100 node $r = 0.1$ RGG instantiation.

is at most r . In this thesis, we are primarily concerned with two-dimensional results ($d = 2$) and the distance measure is the $\mathcal{L}_2$ (Euclidean) norm. For graph connectivity analysis, it is convenient to represent the radius r as a function of n , that is, $r = r(n)$ and if D is a random variable representing the degree in $\mathcal{G}$, then the **expected degree** is given by [Pen03]

$$\mathbb{E}[D] = \pi(n-1)r^2 \sim \pi n r^2. \quad (3.13)$$

A connectivity threshold occurs at $\mathbb{E}[D] = \ln(n)$ which gives a threshold radius of

$$r_t \sim \sqrt{\frac{\ln(n)}{\pi n}} \quad (3.14)$$

and holds *almost surely*. That is to say that the probability of this result occurring tends to one as $n \rightarrow \infty$. Figure 3.3 plots an RGG example with $r = 0.1$ and $n = 100$. The **expected degree** ($\mathbb{E}[D] \approx 3.14$) is below the connectivity threshold of $\ln(100) \approx 4.61$ ($r_t \approx 0.12$). On inspecting the distance between components, it is clear that some components could be bridged with a resource-node, while others cannot and require data-ferrying. Of course, any gap can be bridged given enough resources. However, this comes at the expense of other components if resources are limited.

Figure 3.4 plots an RGG example with $r = 0.1$ and $n = 10$. Using a simple, centralised greedy algorithm, resource-nodes were explicitly given the role as a relay or ferry, and service a single, disconnected user-node pair. Resource were continually added to the graph until it became fully-connected. Due to the small radius and number of nodes, and hence expected degree, the number of ferries dominates the number of relays. With an unbounded number of resources, Fig. 3.5 suggests that certain strategies are more appropriate at different connectivity levels.

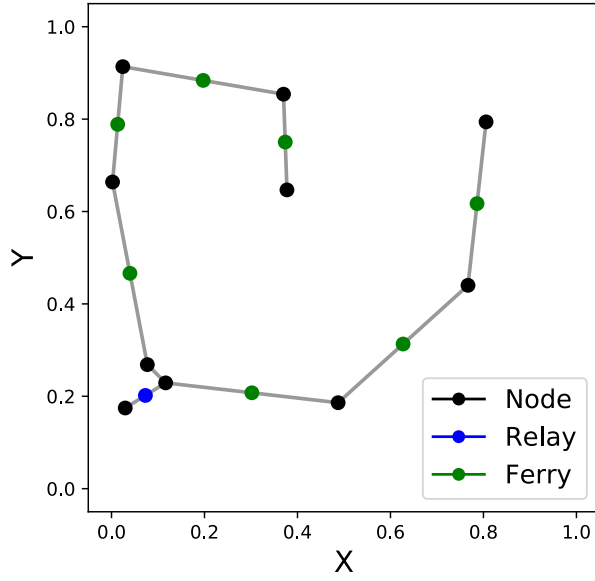


FIGURE 3.4: A 10 node (black), $r = 0.1$ RGG instantiation. Data-ferries (green) dominate data-relays (blue) due to low connectivity.

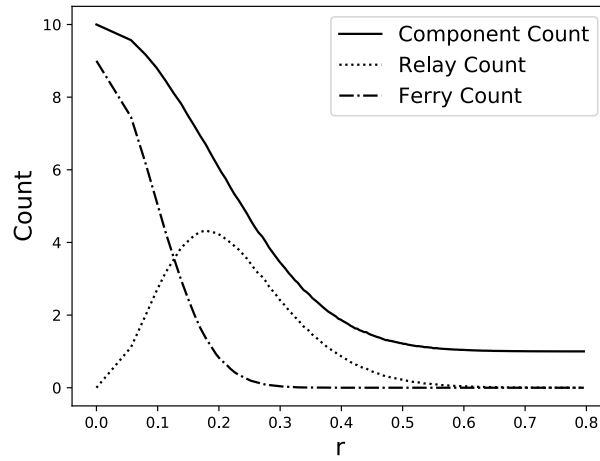


FIGURE 3.5: The number of extra resources required to produce full connectivity on a unit, $n = 10$, RGG for $0.1 < \mathbb{E}[D] < 20$ (converted to radius r by Eqn. (3.13)). Data points are the mean of 10000 trials and CI error bars (not shown) are smaller than the point markers.

3.3 MULTI-AGENT DEEP REINFORCEMENT LEARNING

To investigate a scalable method of deploying many agents with a learned control policy (see Chapter 7), multi-agent reinforcement learning is used. This section provides a background on MADRL and in particular, the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) architecture.

3.3.1 Reinforcement Learning

RL is a framework where agents maximise some notion of reward by continually interacting with their environment [SB17a]. Concretely, an agent observes the environment state $s_t \in \mathcal{S}$ at discrete time t and chooses an action $a_t \in \mathcal{A}$. The environment ‘responds’ to this action by moving to a new state $s_{t+1} \in \mathcal{S}$ and provides a reward signal $r_t \in \mathbb{R}$ to the agent. Using the knowledge that the pair (a, s) produced the reward r , the agent must try to maximise the total reward received from the environment by choosing appropriate actions according to some policy π . Rewards are often discounted such that the agent tries to maximise $r = \sum_{t=0}^T \gamma^t r_t$ over time-horizon T where the discount factor $\gamma \in (0, 1]$ weights the importance of future, rather than immediate, rewards. By continually interacting with the environment in this manner, the agent can learn an optimal policy π^* [SB17a].

Probably the most famous RL algorithm is that of Q-learning which defines an action-value function $Q^\pi(a, s)$ that describes the expected cumulative reward when taking action a in state s using the decision policy π [WD92]. That is,

$$Q(a, s) = \mathbb{E}[r_t | s_t = s, a_t = a, \pi]. \quad (3.15)$$

Q-learning directly approximates the unknown, optimal action-value function $Q^*(s, a)$ via a value iteration update to incorporate new knowledge with existing knowledge. Using the Bellman-optimality equation, the iterative update is given by [SB17a]:

$$Q(s_t, a_t) \leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha \left(r_t + \gamma \max_a Q(s_{t+1}, a) \right) \quad (3.16)$$

where the learning rate $\alpha \in (0, 1]$ regulates how quickly new information is incorporated. When writing the update equation in the form

$$\begin{aligned} Q(s_t, a_t) &\leftarrow Q(s_t, a_t) \\ &+ \alpha \left([r_t + \gamma \max_{a'} Q(s_{t+1}, a')] - Q(s_t, a_t) \right), \end{aligned}$$

the second term can be considered the error between the current and predicted Q-value where a' is the next action. When this term is sufficiently small, Q becomes the optimal action-value function Q^* and the agent can act greedily with respect to the Q-function to extract the optimal policy π^* .

3.3.2 Deep Q Networks

Recently, Deep Neural Networks (DNNs), have been used as function approximators in the Q-learning algorithm to produce Deep Q Networks (DQNs). This approach has seen great success in developing super-human performance in playing Atari video games [Mni+15]. Artificial Neural Networks

(ANNs) are a learning framework inspired by the biological brain and consist of artificial (digital) neurons. Like their biological counterparts, artificial neurons are connected by synapses of varying strengths known as weights, and only ‘activate’ when sufficiently stimulated by an input signal. Activation of a neuron is controlled by an activation function which determines the magnitude of the output signal. DNNs simply refer to an organisation of a number of neurons into many ‘layers’; each DNN contains an input and output layer, and a number of intermediate layers. By appropriately adjusting the synapse weights between neurons, the DNN can ‘learn’ to match the input signal to a known output given enough training examples. ANN research has existed since the 1940s [MP43] with roots in Hebbian Learning [Heb49]; the biological theory of how brain neurons adapt during learning. Given the large number of neurons and synapses in neural networks, training of such networks is difficult and it wasn’t until the invention of the ‘backpropagation’ algorithm [Rum+95] that training of multi-layer (or deep) neural network became feasible. Since that time, a great many neural network architectures have emerged with research examining every facet of the networks themselves as well as their application to a multitude of problems. For an overview of deep neural networks, the reader is directed to [Liu+17].

Similarly to the traditional RL formulation, DQNs learn the optimal action-value function $Q_\theta^*(s, a)$ corresponding to π^* by adjusting neural network weights θ to minimising the loss

$$\mathcal{L}(\theta) = \mathbb{E}_{s_t, a_t, r_{t+1}, s_{t+1}} [(Q_\theta^*(s_t, a_t) - y)^2] \quad (3.17)$$

with

$$y = r_{t+1} + \gamma \max_{a'} (\hat{Q}_\theta^*(s_{t+1}, a')) \quad (3.18)$$

where $\hat{Q}$ is a target Q-function represented by another quasi-static neural network whose weights are periodically updated by copying those from the original network. In addition to the target network formulation, *experience replay* assists in stabilising learning. In experience replay, experience in the form of (s, a, r, s') tuples are stored in a memory buffer $\mathcal{M}$. Batches of experience are then sampled from this memory buffer during training to prevent the network from forgetting past experiences. Weights in the original network can be updated according to

$$\theta \leftarrow \theta + \alpha [(y - Q_\theta(s_t, a_t))] \nabla_\theta Q_\theta(s_t, a_t) \quad (3.19)$$

where α as the learning rate and ∇ is the gradient operator.

3.3.3 (Multi-Agent) Deep Deterministic Policy Gradient

Policy gradient algorithms directly adjust the parameters of the policy (ANN weights θ) to maximise the objective function

$$J(\theta) = \mathbb{E}_{s \sim p^\pi, a \sim \pi_\theta} [r], \quad (3.20)$$

given a state distribution p^π and stochastic policy π_θ , by moving in the direction of $\nabla_\theta J(\theta)$. The gradient of the policy is given by [SB17a]

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [\nabla_\theta \log \pi_\theta(\tau) r(\tau)] \quad (3.21)$$

where τ is the trajectory through the states and actions an agent takes through a given sample, so $\log \pi_\theta(\tau) = \sum_{t=1}^T \log \pi_\theta(a_t|s_t)$ and $r(\tau) = \sum_{t=1}^T r(s_t, a_t)$ for T timesteps. By sampling many trajectories N from the current policy and averaging, we can estimate the expectation of the gradient to be

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^N \nabla_\theta \log \pi_\theta(\tau) r(\tau). \quad (3.22)$$

However, this approach results in a high gradient variance as each trajectory can take very different paths through the state and action spaces, causing successive weight updates to diverge. To reduce this gradient variance and allow learning after each successive step rather than at the end of each episode, the actor-critic approach replaces the reward value required in the gradient calculation with a value derived from a learned Q-function. That is,

$$\nabla_\theta J(\theta) = \mathbb{E}_{s \sim p^\pi, a \sim \pi_\theta} [\nabla_\theta \log \pi_\theta(a|s) Q_\phi(s, a)]. \quad (3.23)$$

In this framework, the *critic* $Q_\phi(s, a)$ estimates the reward, while an *actor* updates the policy parameters in the direction calculated by the critic

$$\theta \leftarrow \theta + \alpha \nabla_\theta (\log \pi_\theta(a|s)) Q_\phi(s_t, a_t). \quad (3.24)$$

Multi-agent settings exacerbate the high variance gradient estimates of policy gradient algorithms since agent rewards often depend on other agents in the system as well, not just the agent in question; without knowledge of their actions, gradients can vary widely. In fact, the probability of moving in the direction of a correct gradient has been shown to decrease exponentially with the number of agents [Low+17]. To overcome this problem, the **MADDPG** has been proposed [Low+17]. In this formulation, the Q-function for each agent contains all agent actions and observations. That is, given N agents with policies $\pi = \{\pi_1, \dots, \pi_N\}$ parametrized by $\theta = \{\theta_1, \dots, \theta_N\}$, the gradient of the expected return for agent i , $J(\theta_i) = \mathbb{E}[r_i]$, is

$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{s \sim p^\mu, a_i \sim \pi_{\theta_i}} [\nabla_{\theta_i} \log \pi_i(a_i|o_i) Q_i^\pi(\mathbf{x}, a_1, \dots, a_N)]. \quad (3.25)$$

Notice that the centralised Q-function for agent i contains all agent actions $a_1, \dots, a_N$ and state information $\mathbf{x}$ (e.g. $\mathbf{x} = (o_1, \dots, o_N)$) but the agent policy is dependent only the local state observation o_i . This formulation allows global knowledge to be available during training but retains distributed execution.

A final characteristic to note about the **MADDPG** algorithm is that it is, as the name suggests, *deterministic*, i.e. it utilises deterministic policies rather than stochastic ones. With deterministic policies μ , the gradient of the expected return relies on the gradient of the critic Q-function, i.e., $\nabla_a Q^\mu(s, a)$, requiring that the action space be continuous so that it is differentiable.

3.4 MULTI-CRITERIA DECISION MAKING

Multi-criteria Decision Making (MCDM) is long-studied discipline of Operation Research which aims to make optimal decisions that involve multiple,

and often conflicting, criteria [ALV13]. The seminal example in the application of MCDM is that of purchasing a vehicle from several available options. Criteria may include cost and comfort, which can easily conflict. MCDM thus provides a structured approach to make the decision. Business decisions such as location planning [ACG11; BE16] and service or supplier selection [MK21] are the typical domain for MCDM. However, it is easy to see how such a framework can be applied to multi-agent problems where multiple agents exist on a single UAS platform and wish to control its position. Indeed, the MCDM framework TOPSIS [TH11] is used for this purpose within this thesis.

TOPSIS is a compensatory algorithm and so allows trade-offs between criteria. Additionally, the algorithm accepts a set of weights for the evaluation criteria as input and so provides a convenient way to influence the priority of criteria. As the name suggests, TOPSIS compares all possible alternatives to the Positive Ideal Solution (PIS) and Negative Ideal Solution (NIS). Alternatives are then ranked such that the chosen alternative has the shortest geometric distance to the PIS and longest geometric distance from the NIS. TOPSIS is easily understood and straight-forward to implement, and as such was chosen as the MCDM tool used in Chapter 6.

4

On the Equidistant Ferry Spacing for Static Path Data Ferry Routes

- **SYNOPSIS** Beginning the investigation into operating in challenged communications environments, a distributed, stigmergic-based coordination of data-ferries for equidistant perimeter spacing is presented. It is observed that a swarm of data-ferries can be modelled and detailed analysis is undertaken using a dynamical systems formulation. The algorithm's ability to adapt to velocity disturbances due to real-world conditions and varying perimeter lengths caused by dynamic user-nodes is also examined. This chapter is based on the papers:

Bradley Fraser and Robert Hunjet. “**Data ferrying in tactical networks using swarm intelligence and stigmergic coordination**”. In: *2016 26th International Telecommunication Networks and Applications Conference (ITNAC)*. Dunedin, New Zealand, Dec. 2016, pp. 1–6;

Bradley Fraser, Robert Hunjet, and Andrew Coyle. “**A swarm intelligent approach to data ferrying in sparse disconnected networks**”. In: *Int. J. Parallel Emergent Distrib. Syst.* (Oct. 2017), pp. 1–22; and

Robert Hunjet, **Bradley Fraser**, Thomas Stevens, Laura Hodges, Karina Mayen, Jan Carlo Barca, Madeleine Cochrane, Ricardo Cannizzaro, and Jennifer L Palmer. “**Data Ferrying with Swarming UAS in Tactical Defence Networks**”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Australia: IEEE, May 2018, pp. 6381–6388.

The reader is directed to [Appendix A](#) for signed Statements-of-Authorship for each of the publications listed above.

4.1 INTRODUCTION

Data-ferrying is an approach to data transfer in disconnected wireless networks where extra network resources, such as a [UAS](#) equipped with a radio transceiver, physically moves between user-nodes to overcome the disconnected environment, to facilitate data exchange. Causes of disconnected

network topologies are irrelevant to the data-ferry approach but may be due to:

- *A large geographical distance between nodes or difficult terrain:* Military and **HADR** operations may require nodes to move such that the network becomes disconnected, or operate in environments where terrain is unfriendly to radio propagation such as in hills and valleys. This can occur in traditional military operations but also in civilian search and rescue scenarios.
- *The use of low-power emitters:* **Low Probability of Detection (LPD)** and **Low Probability of Interception (LPI)** networks [Cam+18] attempt to reduce detection and interception by reducing their emission footprint. Low-power emission may also be a property of the physical platform as radios have different **SWaP** requirements.
- *Interference reducing ‘typically’ expected ranges:* **RF** jamming may reduce the reception range of conventional nodes, facilitating the need for ferries to move closer to user-nodes to effect data exchange.

Static-path ferrying has data-ferries follow a static, pre-defined route along which disconnected user-nodes are located. During traversal of the path, data is exchanged between user-nodes via the ferry. When multiple ferries are used, they must be spaced evenly around the perimeter, travel at the same speed, and in the same direction, to minimise packet latency. This is easily understood by considering the packet latency in the two extremal cases:

1. Equally-spaced ferries : traversal packet latency in this case is $\frac{P}{ns}$ where P is the perimeter length, n is the number of ferries and s is ferry speed.
2. Co-located ferries : traversal packet latency in this case is $\frac{P}{s}$, equivalent to a single ferry.

The case of co-located ferries is identical to a single ferry and existence of multiple ferries does not improve data transfer¹. However, equidistant ferry spacing reduces latency considerably.

A simple solution to this problem is to start the ferries in the correct location on the path, or by starting them sequentially with an appropriate delay, if starting from the same location. This strategy however will not adapt should one of the ferries fail. Similarly, additional ferries cannot be added to the swarm in this case. A centralised controller could facilitate corrections to the ferry spacing but this requires all ferries to pass by the controller before they are re-synchronised. Additionally, we are interested in avoiding centralised approaches and single-points-of-failure. To that end, we look for inspiration from the natural world to solve this problem and find it in the collective behaviour of social insects; in particular, that of ants and termites.

Stigmergy describes the indirect coordination of agents via communication through the environment [DBT00]. By sensing and altering the environment in some way, an agent can affect the behaviour of itself or other agents at a future time. For example, ants utilise stigmergy by laying down

¹This may not be strictly true if the amount of data received at user-nodes exceeds the ferry buffer capacity. However, here we consider ferries to have adequate buffer space. **Section 5.4.1** examines ferry buffer space limitations and introduces a flocking approach to overcome it.

pheromone trails as they forage. Upon sensing these pheromones, other ants are more likely to move in the direction of the strongest signal, thereby creating a positive feedback loop; popular paths become more popular, while unpopular ones decay. This response logic and the pheromone behaviour is encoded mathematically in the next the section.

4.1.1 Algorithm

In the developed algorithm, user-nodes act as the environment for the pheromone; i. e., they are able to be ‘marked’ with, and decay, a pheromone. Data-ferries use simple logic to in response to the pheromone level:

1. If the pheromone level is high, then another data-ferry has visited this user-node recently and it should slow down to *widen* the gap between itself and the previous ferry ; and
2. If the pheromone level is low, then this data-ferry has not been visited recently, to *reduce* the gap between itself and the previous ferry.

This logic is encoded mathematically as follows, at each time step t , the value of the pheromone at the user-node $j \in \mathcal{U}$ is updated as:

$$p_j(t+1) = p_j(t) - \delta \quad (4.1)$$

where p_j is the pheromone level between $[0, 1]$ and δ is the decay value. Upon being serviced by a data-ferry node , the user-node’s pheromone level is reset to 1. Before this occurs, however, the ferry $i \in \mathcal{R}$ adjusts its speed according to:

$$s_i(p) = (s_{\max} - s_{\min})(1 - p) + s_{\min}, \quad (4.2)$$

where s_i is the data-ferry speed on the interval $[s_{\min}, s_{\max}]$ and p is the value of the pheromone of the user-node with which the ferry is communicating. Once the user-node-ferry interaction is complete, the ferry continues to the next user-node in the predefined route. From an implementation perspective, ignoring protocol overhead, the user-node need only transmit the pheromone (represented as a single double-precision float), while the ferry need only transmit an acknowledgement and evaluate the speed function; limiting the communication and processing requirements.

Algorithm 1 provides pseudocode for the user-nodes and data-ferries for the stigmergic data ferrying algorithm. Data-ferries are initialised with the $s_{\min}, s_{\max}$ and the polygon path, while user-nodes only require the decay constant δ . There is no termination condition.

Given the round-robin nature of the algorithm (i. e. traversal of the user-node’s polygon perimeter), a decision must be made by a ferry as to when it is appropriate to move onto the next user-node (vertex in the polygon). Three possible operational modes have been identified and are detailed below.

1. **Algorithm-centric:** With an algorithm-centric approach, a data-ferry will decide that the current user-node has been serviced once the

Algorithm 1 Equidistant Ferrying Algorithm

```

1 procedure USER( $\delta$ )
2    $p \leftarrow 1$ 
3   while true do
4      $p \leftarrow \max(p - \delta, 0)$ 
5     if User in contact with Ferry then
6        $p \leftarrow 1$ 
7 procedure FERRY( $s_{\min}, s_{\max}, s_{\text{init}}$ )
8    $s \leftarrow s_{\text{init}}$ 
9    $u \leftarrow \text{nextUser}()$ 
10  while true do
11     $\text{moveTo}(u)$ 
12    if Ferry in contact with User then
13       $s \leftarrow (s_{\max} - s_{\min})(1 - p) + s_{\min}$ 
14       $u \leftarrow \text{nextUser}()$ 

```

protocol data between the user-node and itself has been successfully exchanged. Whilst enabling the algorithm, this mode of operation limits the amount of user data that can be transferred between the nodes .

2. **Data-centric:** A data-centric approach seeks to ensure that the required data is transferred from the ferry to the user-node. The authors of [HB06] argue the best strategy for a data-ferry to take when transferring data between two user-nodes is to head towards one node until its send buffer is half empty, then switch towards the other. Given the collinear nature of that scenario, the send buffer will then be empty upon losing contact with the first user-node. This approach fails to consider that the user-node itself may be a substantial source of data and requires time to upload to the ferry . It also assumes that the propagation environment is identical when heading towards and away from the user-node.
3. **Position-centric:** A position-centric approach forces the data-ferry to arrive at the location of the user-node before heading to its next user-node target. From a survivability perspective, this scheme offers the greatest resilience against possible RF-jamming. The ferry will spend a substantial amount of time in an area of high SNR, maximising data transfer. Unlike the other two modes, moving to the exact locations of user-nodes also assists with algorithmic convergence since the polygon perimeter length does not change. In the experiments and analysis that follows, a position-centric approach is adopted.

- **ROUTE CALCULATION** can be done in a number of ways depending on the overall communications strategy of the network. For example, if static-path ferrying is utilised in isolation, then the path can be calculated as a solution to the TSP [App06]. A more complex scheme however may see the path being calculated as the convex hull [And79] to provide an ‘information highway’

around the network perimeter, with another data-ferrying approach working internally to transfer traffic to the nearest convex hull user-node vertex, or data-ferry.

4.2 EXPERIMENTAL ANALYSIS

In this section, we describe the initial experiments used to validate the stigmergic data-ferrying algorithm described above. A simulation environment was created with the Java-based **MASON** [Luk+05]. **MASON** specialises in the fast simulation of networks and is an ideal environment for this environment. Three user-nodes are placed on the vertices of an equilateral triangle with a side length of $L = 433$ units and n data-ferrying nodes are randomly placed on the perimeter of the triangle. The ferries are given a random initial speed on the interval $[s_{\min} = 1, s_{\max} = 2]$, and the decay rate enforced by user-nodes is $1/L$ where L is the side-length of the polygon formed by the user-nodes.

Parameter selection pertaining to the decay rate, minimum and maximum allowable speeds greatly affect the timeliness of convergence and the final speeds of the ferries themselves. Here, the parameter selection has been conducted to show ferry convergence irrespective of timeliness constraints. For the number of ferries, n , being between 2 and 10, 100 simulation runs each of 20000 time steps have been undertaken. During each simulation, the distance between successive data-ferries around the perimeter of the polygon was recorded and compared to the optimal value of P/n , where P is the perimeter length.

The simulation uses a fixed-range communications model (see [Section 3.1.2](#)) with a range $r = 1$ unit for all nodes, simulating a very disconnected state. This also allows analytic calculations to be more easily calculated since the perimeter length is only marginally affected by the communications range.

4.2.1 Results

Table 4.1 shows the mean, **95% Confidence Interval (95% CI)** and optimal values (calculated analytically) over the 100 simulation runs for the node spacing around the triangular topology. These results show a near-optimal data-ferry spacing around the user-node network indicating that the swarm intelligent coordination is closely approximating a highly-coordinated solution. Figure 4.1 shows the data-ferry spacing convergence over time for multiple data-ferries for a specific run. This shows that as the number of ferries increase, the time taken for convergence also increases. A similar separation behaviour is exhibited for the different number of ferries but with a decreasing amplitude with an increasing number of ferries. In particular, as the number of ferries increases, the stable solution distance between ferries also decreases, which in turn, reduces the distance between separation (or equivalently speed) updates. This can be understood more clearly by examining Fig. 4.5; the structure of the plot remains the same but with a decreasing step size as ferry numbers increase.

TABLE 4.1: The achieved mean and 95% CI for the inter-ferry spacing of data-ferries on equilateral triangle topology over 100 simulation runs compared to the optimal result.

No. Data Ferries	Mean	95% CI	Optimal
2	650.26	2.23	649.50
3	434.00	1.20	433.00
4	325.90	0.33	324.75
5	260.01	0.79	259.80
6	218.46	0.80	216.50
7	183.79	1.64	185.57
8	162.12	2.56	162.38
9	143.04	3.77	144.33
10	129.04	4.82	129.90

The **Confidence Interval (CI)** of the collective runs is high in some cases, particularly as the number of ferries increases. This is partly due to the algorithm not being provided the necessary time to converge. In rare cases, convergence to a non-equally spaced solution was observed. The reason for convergence to such a state is discussed in **Section 4.3**.

Figure 4.2 depicts the service time, (the average time between data-ferry visits), for user-nodes on the triangular network topology as a function of the number of data-ferries. The service time decreases as the number of ferries increases; however, it is not a linear relationship, indicating that the converged speed of the data-ferry in a system, differs with the number of ferries. This can also be seen in Fig. 4.2 and is discussed in **Section 4.3**.

Figure 4.3 shows a simulation run with four ferries at 0, 2000, 4000 and 6000 time steps. At 6000 steps, the ferries (triangles) have equally spaced themselves around the perimeter of the user-nodes (squares). Examining subfigure (d), the perimeter distance between successive ferries is approximately equally, indicating a converged state.

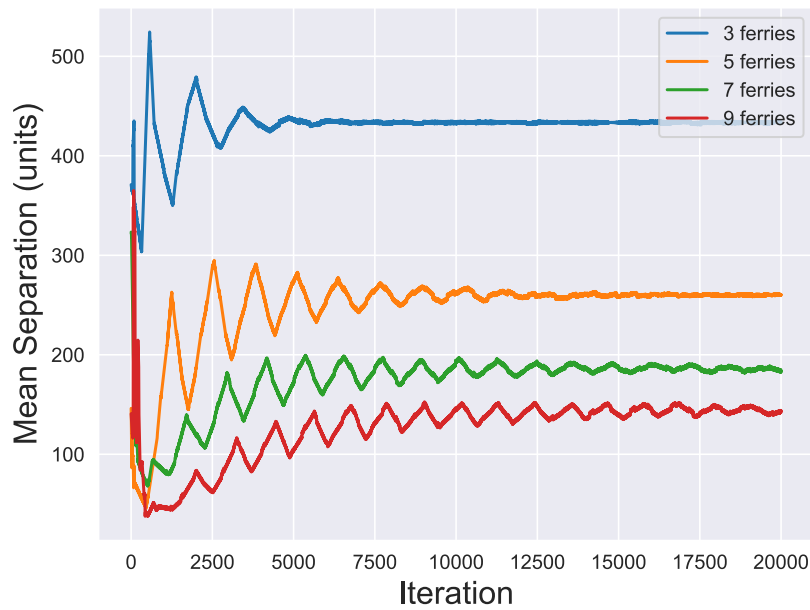


FIGURE 4.1: Convergence of perimeter separation distances for a various number of data-ferries on the equilateral triangle topology. Final distances correspond to those in Tab. 4.1

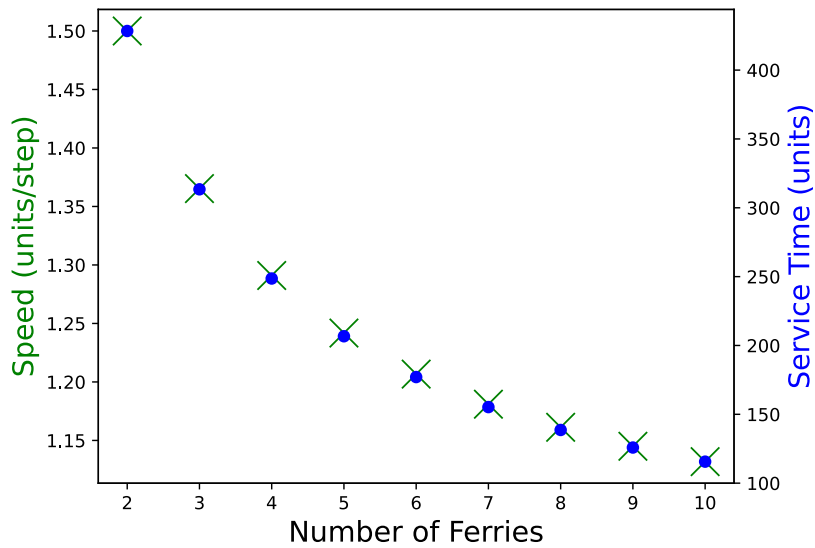


FIGURE 4.2: The quadratic final data-ferry speeds and user-node service time on the equilateral triangle topology.

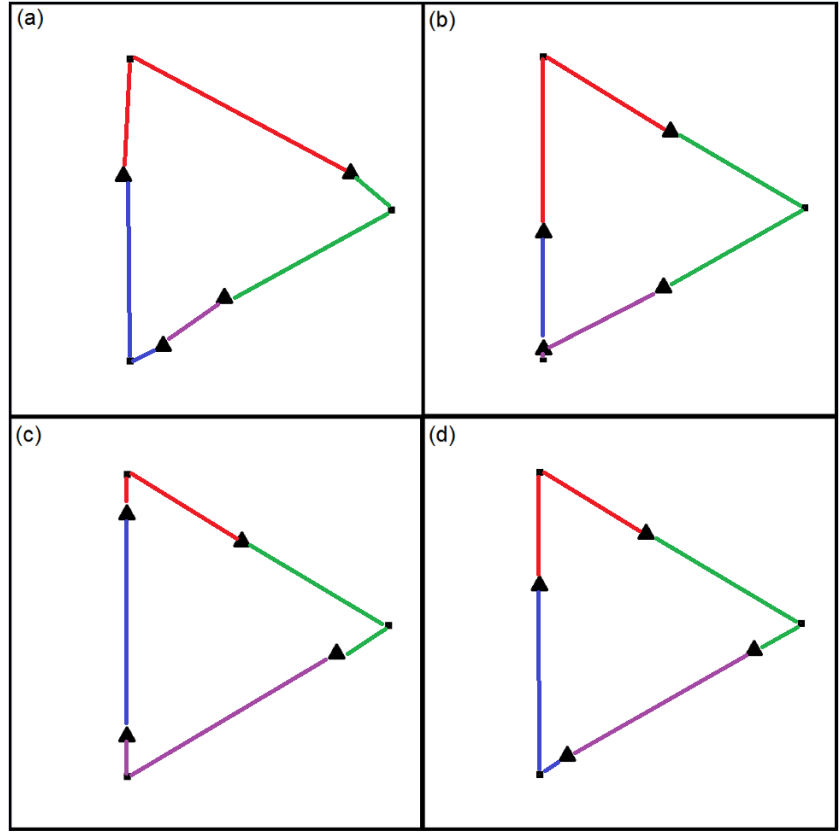


FIGURE 4.3: The stigmergic algorithm at (a) 0, (b) 2000, (c) 4000 and (d) 6000 time steps. Squares represent the static user-nodes situated on the vertices of an equilateral triangle, while triangles represent the dynamic data-ferryes. At time step 6000, the system has converged with the triangle perimeter being separated into equal (coloured) segments by the ferryes.

4.3 MATHEMATICAL MODEL

In this section we present a mathematical analysis of the algorithm. In keeping with Bedau's definition of Weak Emergence [Bed08], this analysis was performed after-the-fact based on observations of the simulated algorithm. The analysis describes bounding cases and idiosyncrasies of the swarm behaviour, and allows us to estimate convergence times and speeds without needing to simulate the algorithm with specific parameters.

We begin by exploring an equally-spaced solution to the simple single user-node, single data-ferry system and examine the system stability and convergence. We then extend this analysis to more complex systems.

4.3.1 Single User Node, Single Data-Ferry Solution

In the simulations presented, the pheromone function was chosen to decay linearly with time t which can be expressed as

$$p(t) = 1 - \frac{t}{T}, \quad (4.3)$$

where $p(t)$ is the pheromone value on the interval $[0, 1]$ at time t and the constant T is the time taken for the pheromone to decay to zero. The data-ferryes adjust their speed s linearly with the pheromone according to Eqn.

(4.2). Substituting Eqn. (4.3) into Eqn. (4.2) yields

$$s(t) = (s_{\max} - s_{\min}) \left(\frac{t}{T} \right) + s_{\min}. \quad (4.4)$$

Assuming a solution of

$$s^* = \frac{d}{t^*}, \quad (4.5)$$

where d is the distance travelled by the ferry before returning to the user-node. Then, substituting into Eqn. (4.4) gives

$$s^* = (s_{\max} - s_{\min}) \left(\frac{d}{Ts^*} \right) + s_{\min}, \quad (4.6)$$

and upon rearranging yields the quadratic equation

$$s^{*2} - s_{\min}s^* - \frac{d}{T}(s_{\max} - s_{\min}) = 0. \quad (4.7)$$

The two solutions for s are therefore

$$s_{1,2}^* = \frac{s_{\min} \pm \sqrt{s_{\min}^2 + 4\frac{d}{T}(s_{\max} - s_{\min})}}{2}, \quad (4.8)$$

with the only feasible solution being

$$s^* = \frac{s_{\min} + \sqrt{s_{\min}^2 + 4\frac{d}{T}(s_{\max} - s_{\min})}}{2}. \quad (4.9)$$

The ratio d/T in Eqn. 4.9 affects the final solution in interesting ways. For example, we can derive conditions where $s^* > s_{\max}$, that is,

$$\begin{aligned} \frac{s_{\min} + \sqrt{s_{\min}^2 + 4\frac{d}{T}(s_{\max} - s_{\min})}}{2} &> s_{\max} \\ \frac{d}{T} &> \frac{(2s_{\max} - s_{\min})^2 - s_{\min}^2}{4(s_{\max} - s_{\min})} \\ \frac{d}{T} &> s_{\max}. \end{aligned} \quad (4.10)$$

Similarly, $s^* < s_{\min}$, requires

$$\frac{s_{\min}^2}{4(s_{\min} - s_{\max})} \leq \frac{d}{T} < 0, \quad (4.11)$$

however this cannot occur since $d/T > 0$.

Table 4.2 compares the analytical value of the ferry separation distances and final velocities for the single ferry, single user-node system to the corresponding separation distances and final speeds for the simulated system with three user-nodes and 2 to 10 ferries. The mathematical results for distance (variables d and s) closely align with the simulated ones. This occurs since, when converged, the time between data-ferry -user-node interactions is uniform. The shape of the polygon and number of users is inconsequential as the distance to the next user-node (be it the same user as in the single user-node, single ferry case, or different user-node, as in the case of multiple ferries and multiple user-nodes) is always the same. More precisely, the assumed solution of equally-spaced ferries satisfies both systems.

TABLE 4.2: Data-ferry separation distances d , analytic speed solutions s_1^* and s_2^* , their absolute derivatives $|f'(s_1^*)|$ and $|f'(s_2^*)|$ for the single user-node, single data-ferry system compared to the simulated mean data-ferry separation distances final speed s for the system with three user-nodes and two to ten ferries.

Analytical (1 User)					Simulated (3 Users)		
d	s_1^*	s_2^*	$ f'(s_1^*) $	$ f'(s_2^*) $	Ferries	d	s
649.50	1.50	-0.50	0.33	3.00	2	650.26	1.50
433.00	1.37	-0.37	0.27	3.73	3	434.00	1.37
324.75	1.29	-0.29	0.23	4.44	4	325.90	1.29
259.80	1.24	-0.24	0.19	5.14	5	260.01	1.24
216.50	1.21	-0.21	0.17	5.83	6	218.46	1.21
185.57	1.18	-0.18	0.15	6.51	7	183.79	1.18
162.38	1.16	-0.16	0.14	7.19	8	162.12	1.16
144.33	1.15	-0.15	0.13	7.87	9	143.04	1.15
129.90	1.13	-0.13	0.12	8.54	10	129.04	1.13

Note that when $d = 0$ in Eqn. (4.8), implying an infinite number of ferries, $s_1^* = s_{\min}$ and that the second velocity solution, s_2^* , is zero. This (infeasible) second solution is an unstable fixed-point of the system. Consider again Eqn. (4.6) (dropping the $*$ notation),

$$s = (s_{\max} - s_{\min}) \left(\frac{d}{Ts} \right) + s_{\min}.$$

This describes a discrete, one-dimensional dynamical system through the mapping $s \mapsto f(s)$, that is,

$$s \mapsto (s_{\max} - s_{\min}) \left(\frac{d}{Ts} \right) + s_{\min} := f(s). \quad (4.12)$$

This discrete-time map iteration of the speed function is relevant since in the single data-ferry, single user-node case, the ferry's next speed is dependent only upon its current speed. It is however communicated through the pheromone at the user-node.

We now wish to investigate the stability of the dynamical system $s_{n+1} = f(s_n)$, $n = 0, 1, 2, \dots$ representing the ferry speed with fixed point $\bar{s}$ such that $f(\bar{s}) = \bar{s}$. Stability analysis of such equations can be found in [Bar11; Ela96; HSD04] and gives the following result:

Theorem 1 (Fixed-point Stability). *If the function $f : \mathbb{R} \rightarrow \mathbb{R}$ is continuously differentiable with a fixed point $\bar{s}$ such that $f(\bar{s}) = \bar{s}$, then the fixed point is stable if $|f'(\bar{s})| < 1$, and is unstable if $|f'(\bar{s})| > 1$ where f' represents the first derivative of f with respect to s . If $|f'(\bar{s})| = 1$ then further analysis is required.*

The derivative in this case is given by

$$f'(s) = (s_{\max} - s_{\min}) \left(-\frac{d}{Ts^2} \right), \quad (4.13)$$

and therefore the solution converges if

$$f'(s) = \left| (s_{\max} - s_{\min}) \left(-\frac{d}{Ts^2} \right) \right| < 1. \quad (4.14)$$

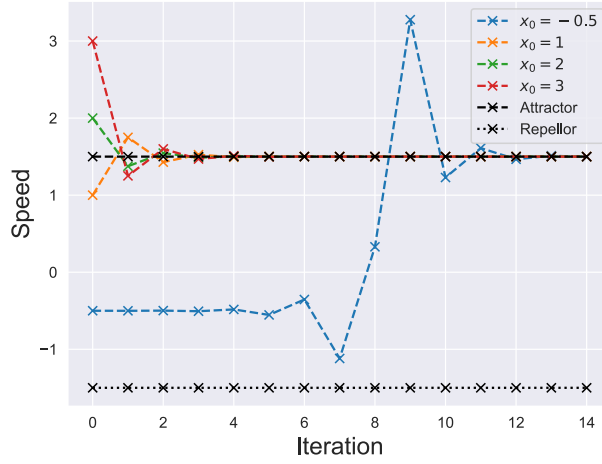


FIGURE 4.4: The discrete time evolution four initial velocities. Three initial velocities begin near a fixed-point attractor of the system and converge quickly to it. A negative initial velocity also starts very near an unstable fixed-point repeller which converges to the attractor fixed-point over several iterations.

If the pheromone does not decay to zero we have

$$\left(\frac{d}{Ts}\right) \leq 1. \quad (4.15)$$

In other words, the minimum speed is still quick enough to ensure that the pheromone does not expire completely during one cycle of the ferry around the network so that

$$\left(\frac{d}{T}\right) \leq s_{\min}. \quad (4.16)$$

If this is true then from Eqns. (4.14) and (4.16), a sufficient condition to ensure convergence is

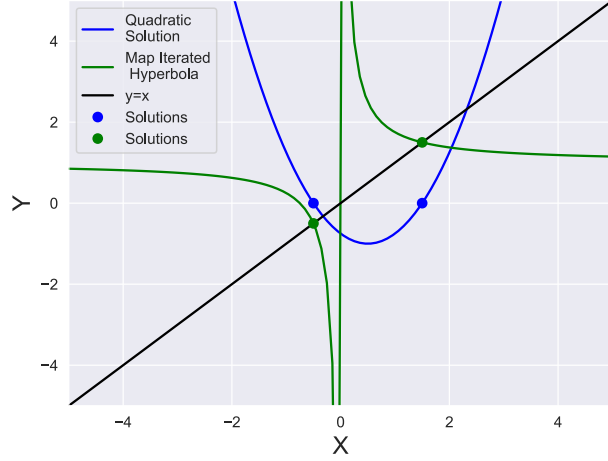
$$(s_{\max} - s_{\min}) \leq s^*. \quad (4.17)$$

This shows two conditions which are sufficient for the algorithm to converge to the optimal solution. The parameters can be chosen for different scenarios to make convergence more likely. So for example choosing $s_{\max} = 4$ and $s_{\min} = 2$ ensures Eqn. (4.17) is true, and convergence of the solution is ensured.

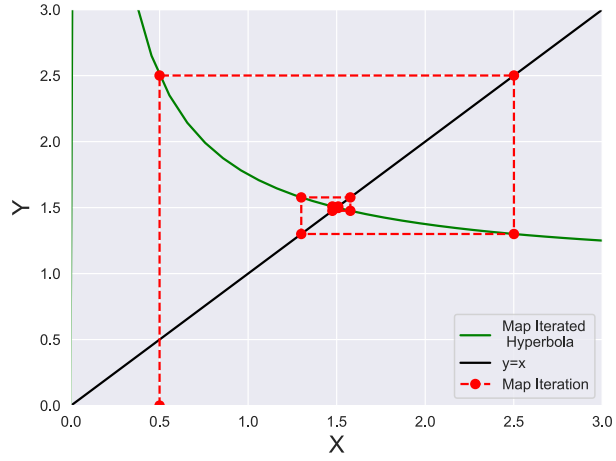
In Tab. 4.2, all s_1^* solutions are stable fixed-points (attractors) while s_2^* solutions are unstable fixed-points (repellers). This is confirmed by their corresponding absolute derivative. Figure 4.4 shows the discrete time evolution for four different initial speeds generated by map iteration (Eqn. (4.12)). Notice that the (infeasible) negative initial speed case, which starts near the unstable fixed point s_2^* , still converges to the stable solution s_1^* . The other initial speeds converge after only a few iterations.

Figure 4.5 depicts the system plot and the *cobweb* plot, or Verhulst diagram², for the system using parameters from line one of Table 4.2. The solutions -0.5 and 1.5 can be seen as x-axis intercepts of the parabola that corresponds to Eqn. (4.7); and also as fixed-points as the intercepts of the hyperbola, that corresponds to the iterated map Eqn. (4.12), and the line $y = x$. Figure 4.5 also shows the solution beginning with an initial value

²Named for Pierre Verhulst who originally published works on population growth using the continuous Logisitic Equation [Ver45], of which, the Logisitic Map is a discrete-time variant and closely related to the model we present here.



(a) Solutions appear as the roots of Eqn. (4.7) and the intersection of Eqn. (4.12) and $y = x$.



(b) Map iteration converges to the attractor fixed-point.

FIGURE 4.5: The Verhulst (cobweb) diagram of the dynamical system.

of 0.5. Similarly to other initial values seen in Fig. 4.4, it converges after several iterations.

4.3.2 General Solution

For the general case in which there are n ferries and m users, we can see that a solution to the pheromone algorithm is that every ferry is equally-spaced around the perimeter and that they all travel at the same constant speed s^* . Let the perimeter distance be D and

$$\hat{d} = \frac{D}{n}, \quad (4.18)$$

then the solution to the algorithm solves the two equations

$$t^* = \frac{\hat{d}}{s^*} \quad (4.19)$$

and

$$s^* = (s_{\max} - s_{\min}) \left(\frac{t^*}{T} \right) + s_{\min} \quad (4.20)$$

which can be solved as before so that we can get one solution being

$$s^* = \frac{s_{\min} + \sqrt{s_{\min}^2 + 4 \frac{\hat{d}}{T} (s_{\max} - s_{\min})}}{2}. \quad (4.21)$$

Similarly to before, this can be shown to be a stable solution when both the pheromone does not expire and

$$(s_{\max} - s_{\min}) \leq s^*. \quad (4.22)$$

However, in this general case there could feasibly be more solutions to these equations, and it is difficult to prove that the described solution is unique. What this does show is that the solution, in which all the ferries behave identically, is stable. Additional experimentation will now show that in realistic scenarios the ferries do in fact converge to this solution and can overcome disturbances to user-node movement.

4.3.3 Case Study: A Simulated Amphibious Landing

Some military missions, such as amphibious landings, may require nodes to move without regard to the state of the communications topology. In particular, littoral insertion operations require communications from tens of kilometres offshore to possibly hundreds of kilometres inland. Communications in such an environment are difficult. This section describes the simulated application of the algorithm to such a scenario. In particular, it examines how the algorithm copes with:

- A dynamic perimeter length;
- Non-equilateral polygons; and
- Dynamic addition and removal of ferries.

Consider the fictitious scenario shown in Fig. 4.6; a **Landing Helicopter Dock (LHD)** command has launched 6 amphibious craft, initially travelling as two groups towards separate beaches. Once there, two craft remain stationary while the four others move inland. During this insertion two of the four craft cease moving while the final two craft move further inland and converge on an objective. The final network is left in a completely disconnected state. Note that any number of these craft may represent a gateway to a cluster of individual nodes representing for example, soldier radios.

This scenario was simulated with the **LHD** being able to create new data-ferries to perform the data-ferry function. Conceptually, the **LHD** node is launching a **UAS** with the last known user-node locations as a list of way-points. With this conceptual view, the following assumptions were used:

- The ranges $R_{\text{amphib}} < R_{\text{LHD}} < R_{\text{UAS}}$.

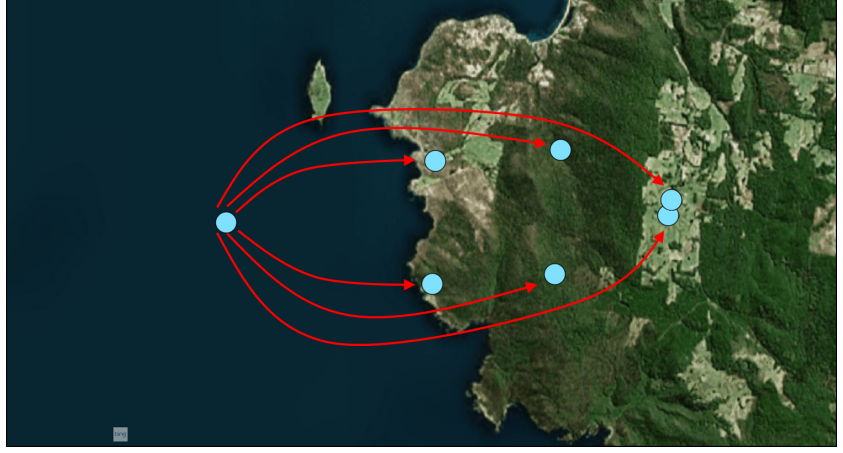


FIGURE 4.6: A seven user-node, fictitious landing scenario generated with the military communications planning tool Harlequin [ES16]. The leftmost node is the LHD command ship which launches amphibious craft that travel inland along the indicated paths.

- The speeds $s_{\text{amphib}} \ll s_{\text{UAS}}$ and $s_{\text{LHD}} = 0$.
- The **LHD** has prior mission knowledge and so can provide the **UAS** with the expected positions of the amphibious craft. It is also assumed that the **LHD** dictates the rotation direction and visitation order.
- In this simple scenario, amphibious craft always create the expected convex polygon, that is, the next-hop decision is always correct.
- A fixed range communication model is applicable since communication to and from the **UAS** is predominantly line-of-sight (assuming the **UAS** are operating at an altitude much greater than the average local terrain); and jamming is not considered in this scenario.
- All antennas patterns are isotropic.

The simulation was allowed to run for 20000 time steps with $R_{\text{amphib}} = 40, R_{\text{LHD}} = 60, R_{\text{UAS}} = 75$ units placed in an environment of dimensions 600×600 units. The **LHD** can launch a **UAS** data-ferry once initial communication between an amphibious craft is lost and also whenever the service time visits exceeds a specified **QoS** constraint threshold of 150 time-steps. Upon launching a **UAS**, the **LHD** is locked from launching another for 1000 steps to allow the previous launch to take effect. The decay constant was $1/L$ where L in this case is the averaged side length of the irregular polygon formed by the user-nodes. User-nodes use the most up-to-date information of other user-node locations passed on by the ferries. Note that in this case user-nodes are using different decay constants, since global information is not available. The user-nodes move at 10% of the slowest speed of the **UAS** along specific paths to replicate the mission shown in Fig. 4.6. **UASs** were able to choose their speed from the interval $[1, 4]$ units-per-step. During the simulation, the number of time steps between successive data ferry visits at the **LHD** is recorded. To demonstrate the ability of the algorithm to recover from failures, a data-ferrying node is removed from the simulation, replicating a catastrophic failure at time step 15000.

Figure 4.7 shows the service time for the LHD over the life of the mission. There are several distinct stages in the simulation:

1. Step 0: Represents the time when initial contact between the LHD and the amphibious nodes is lost.
2. Steps 1 $\rightarrow$ 6000: As the amphibious nodes travel further from the LHD, the service time increases. During this time the single UAS is gradually increasing its speed in response to the increasing distance between amphibious nodes. This results in a constant gradient in the LHD service time.
3. Step $\approx$ 6000: The service time becomes unacceptable and a new launch is triggered.
4. Steps 6000 $\rightarrow$ 8000: The amphibious nodes are still moving, resulting in an increasing service time. The two ferries work to service the LHD uniformly in time, again shown by a constant gradient. A shallower gradient is seen since there are two UAS. Note that in order for a constant gradient to be seen, the UAS must share the same speed.
5. Steps 8000 $\rightarrow$ 12500: Some amphibious nodes change their trajectory resulting in a different rate of change of the polygon perimeter. The ferries continually adapt to the change resulting in a different gradient.
6. Steps 12500 $\rightarrow$ 14500: The amphibious nodes reach their final positions and the LHD service time has zero gradient with a value just below the QoS threshold.
7. Step $\approx$ 14500: One of the UAS suffers a catastrophic failure.
8. Steps 14500 $\rightarrow$ 15500: The LHD service time breaches QoS constraints for a second time and another UAS is launched. The UAS work to achieve uniformity in LHD service time.
9. Steps 15500 $\rightarrow$ 20000: The two ferries have converged to uniformity once again.

Also shown in Fig. 4.7 is the analytic value of the LHD wait time calculated at each time step. This calculation has knowledge of all node locations, and therefore has up-to-date information of the polygon perimeter and hence the pheromone decay rate. The traces show good agreement between simulation and analytical results. Minor mismatches can be attributed to non-LHD nodes using heterogeneous and time-lagged decay values. Another source of error comes from the analytical results using the actual user-node locations to calculate the polygon perimeter, rather than where a ferry would come into contact with a user-node. Although this offset has been compensated for, minor mismatches still occur due to the geometry of the polygon.

One feature of the Fig. 4.7 is the time lag between the traces. This can be explained by the delay in data moving around the network. For example, in the simulation implementation, the LHD must wait for a ferry to arrive in order to decide if a QoS breach has occurred whereas the all-knowing analytic calculation is immediately aware. Moreover, the analytic calculation

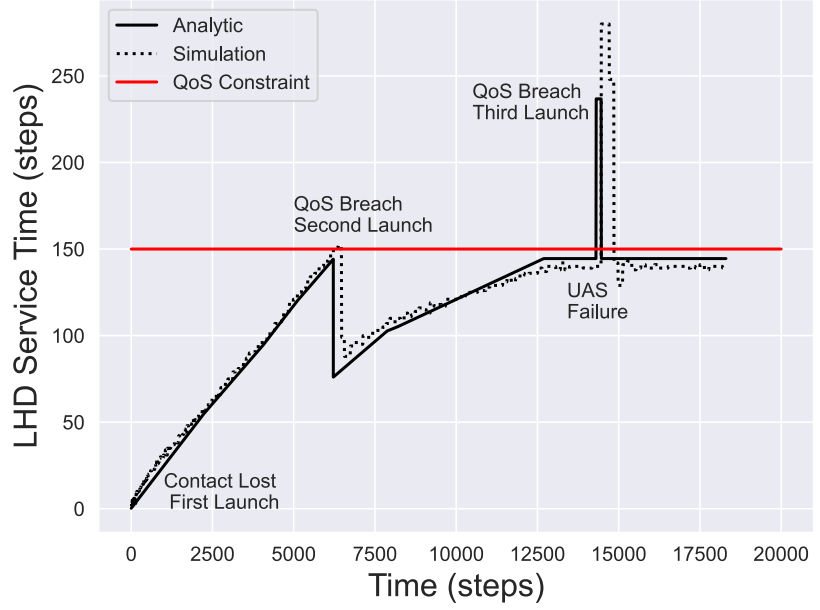


FIGURE 4.7: Analytic and simulated LHD service time over the life of the mission. The service time QoS threshold is 150 time-steps, shown as the red solid line.

does not need time for convergence and so we see only a momentary spike when a ferry fails, the width of which is the time it takes for the **LHD** in the simulation to launch another **UAS**. This can be seen in Fig. 4.7 as the overlapping of the traces at approximately step 14500.

4.3.4 Physical Implementation

This section explores the physical implementation of the stigmergic data-ferrying algorithm on small **UAS** for data dissemination. Results from the physical experimentation are compared with those generated from simulation and the mathematical models of previous sections.

Simulation

Consider the case in which user-nodes are positioned on the vertices of a square with perimeter $P = 4L$, where $L = 6$ units, with the parameters for the algorithm listed in Tab. 4.3. These values closely match the physical experiment described later in this section, ensuring:

1. Timely convergence of ferry speed to a steady state;
2. A sufficiently large variation in ferry speed so that the emergent behaviour of the system was observable; and
3. Safe operation of the experiment (the slower speeds allowed safety pilots time to prevent collisions).

Once picked up by a ferry, the transfer time for data packets to reach all recipients is $\frac{3L}{s^*}$, and on average, a user-node will wait $\frac{P/2n}{s^*}$ before being serviced. On convergence, the expected dissemination latency is then:

$$\frac{3L + \frac{1}{2}(P/n)}{s^*} \quad (4.23)$$

TABLE 4.3: Algorithm parameters for simulation experiments.

$n(\text{ferries})$	$s_{\min}$ (units/step)	$s_{\max}$ (units/step)	T (steps)
1	0.1	0.5	24
2	0.1	0.5	45

TABLE 4.4: Algorithmic parameters and the mean (μ), standard deviation (σ), and calculated results for stigmergic data-ferrying simulations conducted with four user-nodes on a 6×6 unit square for n ferries.

n	Separation (units)			Speed (units/step)			Diss. Time (steps)		
	μ	σ	P/N	μ	σ	Eqn. (4.24)	μ	σ	Eqn. (4.23)
1	N/A	N/A	N/A	0.50	0.00	0.50	60.49	13.81	60.00
2	12.00	0.01	12.00	0.38	0.01	0.38	64.08	9.30	63.09

where s^* is the converged speed and n is the number of ferries.

Simulations were run for 900 time steps (corresponding to 15min in the physical experiment) in the **MASON** simulation environment using a fixed-range communications model (see Section 3.1.2) with a range $r = 1$ unit for all nodes. The final distance separating ferries and their final speed(s) recorded. During simulation, the number of time steps taken to distribute data packets (generated by each user at every time step) to all other user-nodes, the *dissemination time*, was recorded. The mean dissemination time during the last 10% of the simulation was computed to capture its value at (or near) convergence.

Taking the feasible solution from Eqn. (4.8), the final speed of all ferries is given by:

$$s_f^* = \frac{s_{\min} + \sqrt{s_{\min}^2 + 4P(s_{\max} - s_{\min})/nT}}{2} \quad (4.24)$$

for $\frac{P}{nT} \leq s_{\max}$ and $s_f^* = s_{\max}$ otherwise.

Table 4.4 shows results for two series of 1000-run simulations, conducted for single and dual-ferry cases; and Fig. 4.8 shows the speed of the ferries during a run of the dual-ferry case. The results display several interesting features, described below.

1. The single-ferry scenario is a case in which $(P/N)/T > s_{\max}$, which would yield a final ferry speed of 0.68 units/step, if the computed value was not limited to $s_{\max}$. The standard deviation of the dissemination times obtained in the single-ferry simulations is largely due to users having to queue packets for greatly varying amounts of time before they are retrieved.
2. The final ferry speed and dissemination time for the dual-ferry case are in close agreement with the computed values, with convergence to a value of s_f that is less than $s_{\max}$ within 300 time steps.
3. The dual-ferry case achieved a slower dissemination time than the single-ferry case, due to the lower converged speed. This highlights the trade-off between using a single, faster-moving ferry, at the expense of **UAS** battery life, or two slower-moving ferries, at the expense of service delivery.

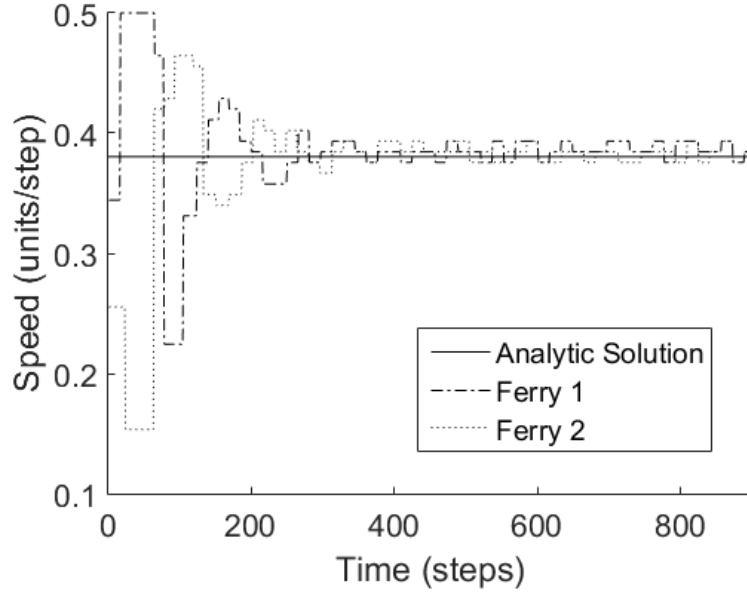


FIGURE 4.8: The speed responses of two simulated data-ferries servicing four user-nodes on a 6×6 unit square.

TABLE 4.5: Emulation parameters.

n	$s_{\min} \text{ ms}^{-1}$	$s_{\max} \text{ ms}^{-1}$	$T \text{ (s)}$
2	57.7	97.5	130
3	47.3	91.5	271

Emulation

Software emulation facilitated testing with a set of conditions scaled-up from those used in simulation. The emulation environment, described in [ES16], makes use of Persistent Systems' Wave Relay radios connected to an RF switch box in which the attenuation between ports is dynamically controlled according to the RF-propagation model described in Section 3.1.2. The user-nodes and ferries ran independently on a Linux machine attached to the radios and both devices were enclosed in an RF-isolation box. In this manner, realistic propagation path loss is observed between communicating nodes. The advantage of the emulation is that the real world effects upon the algorithms induced by the communication systems can be observed. The rotorcraft were modelled with general Proportional-Integral-Derivative (PID) controllers and low-fidelity physics models. Control and environmental sensing were implemented through software known as the HAIL (also described in [ES16]). This software enabled re-use of the algorithm executables when transitioning to physical testing. The geometric shape for the emulated test was four user-nodes arranged in a 5×5 km square. The algorithmic parameters are shown in Tab. 4.5 and were selected for fast convergence.

Ferries began in random initial positions near one of the user-nodes, with their speeds set randomly between $s_{\min}$ and $s_{\max}$. RF-propagation values were configured so that communication was readily available with up to 50m separation, fading rapidly beyond this. For each scenario, five emulations

TABLE 4.6: Algorithmic parameters and the mean (μ), standard deviation (σ), and calculated results for stigmergic data-ferrying emulation conducted with four user-nodes on a 5×5 km square.

n	Speed (ms^{-1})			Service Time (s)			Diss. Time (s)		
	μ	σ	Eqn. (4.24)	μ	σ	$P/N/s^*$	μ	σ	Eqn. (4.23)
2	92.9	0.8	91.3	115.5	3.5	109.6	233.3	34.7	219.2
3	65.0	0.3	64.2	108.0	1.9	103.8	301.3	42.4	285.4

were run for 30 minutes each; and the final service time, which in this case is given by $P/N/s_f^*$ at convergence, was evaluated. Table 4.6 shows the mean results for five runs of each case. Similar to the simulated results, the emulated results are in close agreement with the analytical results. The differences are attributable to delays imposed by the aircraft model (e. g. finite acceleration) and radio connection times.

The time series in Fig. 4.9 show the speed of the ferries during a two-ferry experiment with 5×5 km square of user-nodes. The red and orange traces show the desired speeds of the ferries as demanded by the algorithm, while the blue and green traces show their actual speeds, as reported by the emulated aircraft controller. Abrupt drops in a ferry's speed signify occasions when the rotorcraft reached a user-node and changed direction. Over time, speeds converge to steady values that are identical for both ferries.

During this time series, ferry 1 was first to interact with the nearest user-node. Because user-nodes began with a pheromone level of zero, ferry 1 attained the highest possible speed. It maintained this speed for nine node visits, just over two circuits of the square. During this time, the pheromone set by the trailing ferry (ferry 2) had decayed to zero before ferry 1's next interaction. On the tenth visit, the pheromone set by ferry 2 had not yet decayed to zero. Data-ferry 1 thus began to slow. Similarly, ferry 2 gradually increased its speed because it encountered progressively lower pheromone levels at each user-node. These results demonstrate the resilience of the algorithm to timing perturbations resulting from software delays, such as algorithm timers, and from delays caused by physical modelling of the aircraft which imposed variation in acceleration when approaching and leaving user-nodes.

Physical Deployment

Real-world effects, e. g. aircraft acceleration and communications instability, can affect algorithmic performance. This section examines these effects through tests conducted with real UAS; specifically **Commercial-Off-The-Shelf (COTS)** quadcopters.

PLATFORM, TESTING ENVIRONMENT AND CONTROL Testing was conducted in **DSTG's** Indoor Flight Laboratory, which is fitted with an Optitrack motion-capture system. The system consists of eight Prime 17W and forty Prime 13W cameras managed with OptiTrack's Motive software. It tracks infrared LEDs arranged in distinctive patterns on each quadcopter and from this,

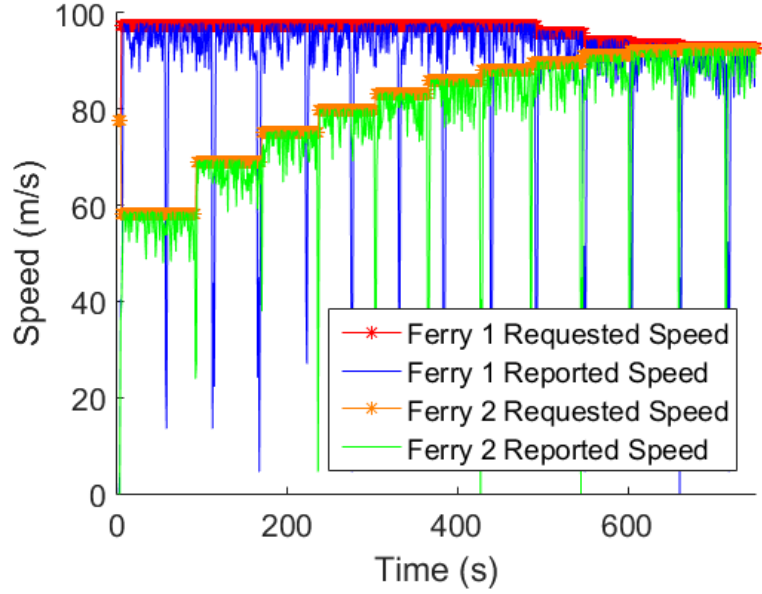


FIGURE 4.9: The speed response of the rotorcraft ferries during an emulated experiment in which two ferries serviced four ground nodes arranged on a 5×5 km square.

calculates a local position. The tracking data for each quadcopter is then streamed via Wi-Fi[] to an on-board Odroid XU4 computer.

Each quadcopter weighed ~ 1.5 kg and was 400 mm in diameter, with propellers in an X4 configuration. Each was equipped with a Pixhawk flight controller operating with PX4 V1.3.1 firmware. High-level control was managed by the on-board Odroid, which operated with the **Robot Operating System (ROS)** [Qui+09] running on Ubuntu 14.04.5, a light-weight version of Ubuntu. Our **ROS**-based control software managed the high-level operation and tasking of each quadcopter and accepted algorithmic input via **HAIL**. A 5200 mAh battery provided a maximum flight time of ~ 10 minutes.

HAIL provided an interface between the **ROS**-based software and the stigmergic data-ferry algorithm, relaying quadcopter status information from the **ROS**-based software to the algorithm. A **HAIL** mobility agent translated navigation directives of the algorithm (waypoint and velocity settings) into commands issued to the **ROS**-based software. A **HAIL-ROS** interface provided translation between Java and the **ROS** messaging bus and node -registration system. The interface, depicted in Fig. 4.10, was built utilising the ROSjava implementation.

HAIL ensured compliance with velocity and positional bounds and confirmed aircraft readiness before messages were translated and sent to the **ROS**-based software. The interface also provided quadcopter state information to the algorithm. Command-acceptance functions enabled reliable command delivery and notifications of command rejection, whilst also providing a ‘safety-gate’, only allowing the forwarding of algorithmic commands if “Autonomous Mode” was selected. This was enabled only once the quadcopter was airborne and ready to commence autonomous navigation.

HAIL also provided a multi-cast transport service for the algorithmic software, enabling the flow of pheromone and node location peer messages

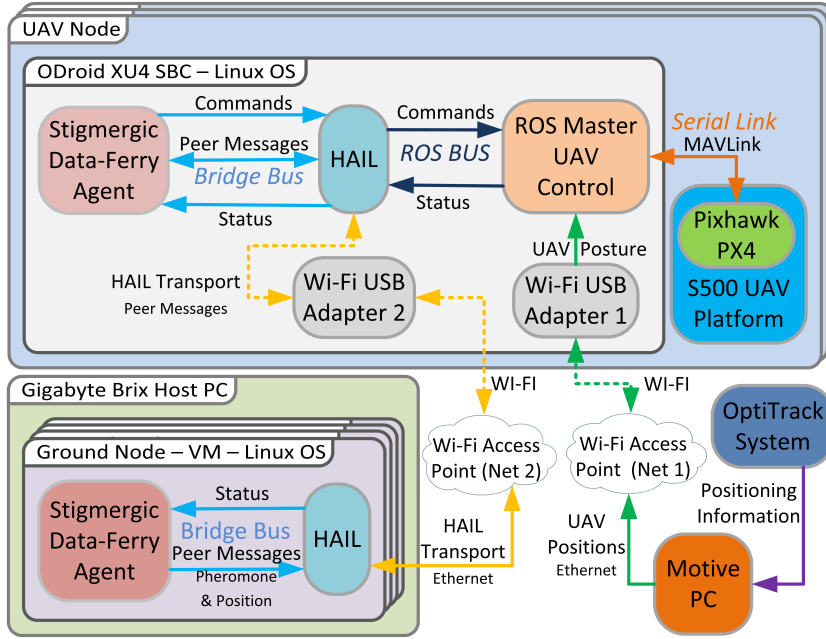


FIGURE 4.10: Experimental setup utilising HAIL, ROS, and OptiTrack.

between quadcopters and ground nodes. This transport service utilised the quadcopters' second Wi-Fi adapter and the physical network adapter of a single PC that was in turn bridged to the host ethernet adapter of each ground user-node virtual machine residing within the physical node. Using **HAIL** to provide the hardware interface for the algorithm allowed a seamless transition from emulation to physical implementation, as the interface remained the same, necessitating only configuration changes to the **HAIL** software.

The **ROS**-based software provided a control interface for the quadcopters. This software ingests commands from a variety of sources in a variety of forms. In this case, input was taken from a command-line interface (as take-off, altitude-locking, and landing commands) and the **HAIL** interface (way-point commands). The software included a pseudo-geo-fence, to enforce safe operating limits, and a state machine, to ensure sensible acceptance and rejection of commands during a flight. It also included conversion protocols, for example, between the geo-referenced coordinate system used by **HAIL** and its own local coordinate system. The software applied safety and bounds checking to all commands and parameters, ensuring they were within the operating ranges. It then provided feedback to the external tasking software as to the acceptance of the request or the reason for rejection, optionally followed by periodic updates on its progress towards them. Finally, the software sent the final output commands to the Pixhawk, via **Micro Air Vehicle Link (MAVLink)** messages over a serial link.

During the tests, quadcopter speed was limited to 0.5ms^{-1} to ensure safe operation. A pseudo-geo-fence bounded the flight space and prevented the quadcopters moving outside the coverage of the motion-capture system or colliding with the walls of the laboratory. When multiple quadcopters were flown, they were separated in altitude by 2m. A take-off sequence, designed to minimise the effects of downwash on the lower quadcopter, was

TABLE 4.7: Mean (μ) service times and 95% CI observed at convergence in stigmergic data-ferry experiments.

Node ID	Service Time (s)			
	Single-Ferry		Dual-Ferry	
	μ	95% CI	μ	95% CI
1	61.0	2.3	35.8	1.5
2	61.0	1.5	35.5	2.5
3	61.2	1.4	36.2	1.7
4	60.8	2.0	35.0	1.8

implemented as follows: the higher quadcopter was positioned closer to the first node and launched first via the **ROS**-based software. As it ascended to its flight altitude, the second ferry was launched. Once both were locked into their flight altitudes, **HAIL** was switched to Autonomous Mode and began controlling the ferries, commanding the first, then the second, to move towards the first waypoint. This sequence ensured a staggered arrival at the first waypoint and minimised interference between quadcopters.

During flight, the accuracy of each quadcopter's position was monitored with the Motive software. Each quadcopter also had an instance of QGround-Control to monitor Pixhawk telemetry data, and a remote pilot on standby, ready to take manual control in the event of unexpected behaviour.

The **HAIL Graphical User Interface (GUI)** showed the position, velocity, current waypoint and status of each quadcopter. The **HAIL GUI** also provided the ability to issue manual waypoint, velocity, and take-off commands in addition to changing the **HAIL** Autonomous Mode, controlling the flow of commands from the algorithm to the quadcopter. The **ROS** command-line interface provided feedback on the high-level operation of the quadcopters, including state information and the processing of data received from **HAIL**. These interfaces, and the remote pilots, provided monitoring and allowed **Human-On-The-Loop (HOTL)** control.

The ground nodes consisted of four Ubuntu Linux virtual machines running on a Gigabyte Brix PC. These were bridged to a single Ethernet network interface, attached to a Wi-Fi access point. Pheromone readings were accessed by the quadcopters through **HAIL**'s multi-cast service using a second **Universal Serial Bus (USB)** Wi-Fi communication payload, with the networks configured on non-conflicting channels. The ground nodes were placed on the vertices of a virtual rectangle $6.0\text{m} \times 5.4\text{m}$ in size; and the communication range was artificially limited to $r = 0.2\text{m}$ in the horizontal plane using the fixed-range communications model (Section 3.1.2), to mimic a sparse, disconnected network. Finally, the stigmergic control algorithm recorded system variables such as requested speeds and pheromone levels and could also be accessed via **Secure SHell (SSH)** or **Virtual Network Computing (VNC)** applications.

The stigmergic data-ferry algorithm was tested with a single quadcopter and with dual quadcopters using parameters comparable to those listed in Tab. 4.4 (i. e. $s_{\min} = 0.1\text{ms}^{-1}$ and $s_{\max} = 0.5\text{ms}^{-1}$, with $T = 24$ and 45s , respectively, for the single and dual-ferry cases). The converged service time of each ground node was measured over five and six runs for the single and

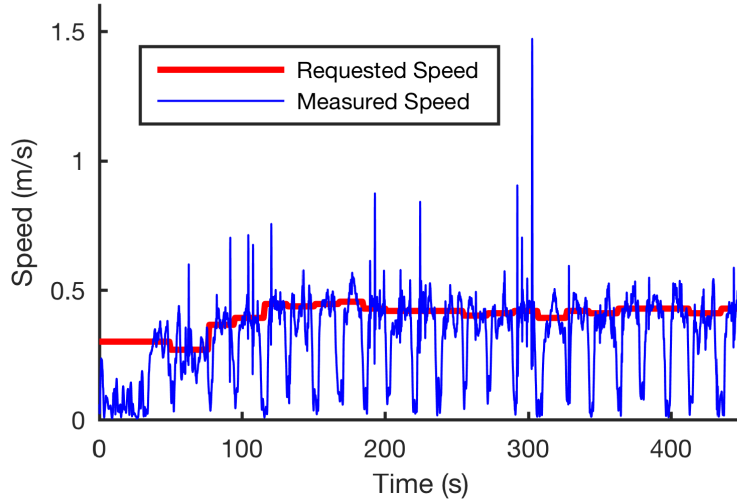


FIGURE 4.11: The speed response of a quadcopter during a stigmergy-based data-ferrying experiment with two data-ferry servicing four ground user-nodes at the vertices of a $6.0 \times 5.4\text{m}$ rectangle.

dual-ferry cases, respectively. Experiments were conducted for the maximum flight time or until the algorithm converged (i. e. the service time for a given ground user-node changed by no more than 1s over three successive ferry laps).

RESULTS Table 4.7 shows the results for measured service times at convergence. The mean for each user-node and the 95% CI around the mean are provided. The service time calculated for the single and dual-ferry experiments are 45.6s and 30.6s, respectively. It can be seen that the measured values are larger than these predictions. This is due to the dynamics of real flight: the perimeter traversal was predicted to be $(2 \cdot 6\text{m} + 2 \cdot 5.4\text{m}) / 0.5\text{ms}^{-1} = 45.6\text{s}$. Additional time taken in the experiment is attributable to time lost as the quadcopter decelerated on approach to each ground user-node and re-accelerated to the desired speed after an interaction.

Speed perturbations can be seen in Fig. 4.11, which shows the speed commanded by the algorithm and the actual speed of one of the quadcopters during a dual-ferry experiment. Despite some perturbations, the algorithm converged, as evidenced by very similar service times on the ground user-nodes in Tab. 4.7. Similar algorithmic resilience was noted in the emulation results. Figure 4.12 shows overhead views (captured with a ceiling-mounted GoPro camera) of (a) the beginning of the experiment, when both quadcopters are first on the perimeter of the rectangular flight area, and (b) the positions of the quadcopters near the end of the experiment, after the algorithm had reached convergence. These results correspond well with the simulation and emulation experiments.

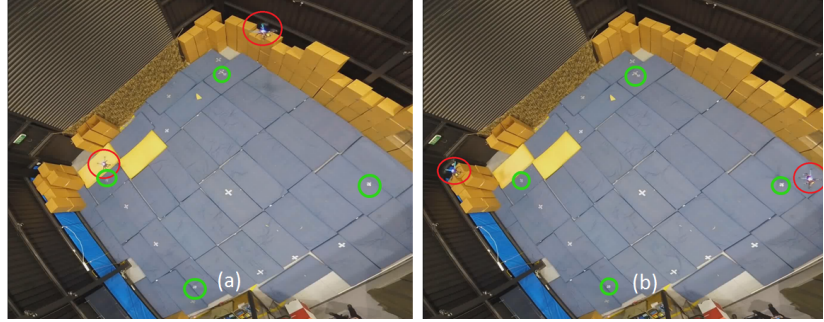


FIGURE 4.12: Snapshots of a stigmergy-based dual-ferry experiment acquired (a) just after its start and (b) after convergence. The red circles highlight the data-ferrying quadcopters, and the green circles show the positions of the virtual ground user-nodes .

4.4 SUMMARY

This chapter introduced a swarm intelligent, bio-inspired algorithm based on the mechanism of stigmergy for the distributed control of data ferries. The purpose of the algorithm is to ensure that ferries are equally-spaced around a static, pre-defined data-ferrying route, to maximise data transfer efficiency and minimise packet latency. The approach avoids centralised control and single-point-of-failure issues and thus is suitable for harsh physical and **RF** environments. The algorithm was modelled as a discrete-time dynamical system and constraints on algorithm convergence were investigated. The chapter also describes the physical implementation of the algorithm on small **UASs** where it was found to be resilient to timing perturbations and thus providing an extra level of validation of the algorithm in the real-world. Additional simulation experimentation examined a case study of moving nodes that involved the dynamic insertion and removal of ferries in response to **QoS** constraints. Given the extra difficulty of a continually changing ferry route length, the algorithm still managed to provide equidistant spacing, owing to the extreme stability of the attractor fixed-point solution of the underlying dynamical system. Finally, a description of collaborative work on an algorithm extension to coordinate teams of ferries was given.

The contributions of this chapter are primarily focused on static-path data-ferrying. Static-path data-ferrying was chosen specifically in response to the state of Trusted Autonomy [**ASR18**] at the time of algorithm development, and the uptake of autonomy in the Defence and Civil communities. The algorithm provides a good balance between autonomy and self-organisation as it is predictable in terms of individual and collective behaviour. Given the fixed-route nature, it is also easier to implement from an airspace authority point-of-view. As trust in autonomy continues to grow, and the enabling technology becomes cheaper and easier to implement, ferries will be given more freedom in determining their own locations. This opens the problem up to dynamic-path (next-hop) data-ferrying where routes are not fixed. We turn our attention to this expanded problem in the next chapter (**Chapter 5**). While many dynamic-path data-ferrying schemes exist, we specifically examine the problem of variable graph connectivities and agent response in terms of data-ferrying and data-relaying.

5

Adaptable Data Transfer under Variable Graph Connectivity Conditions

- **SYNOPSIS** While the previous chapter focussed exclusively on data-ferrying, a challenged network is likely to be comprised of variable graph connectivity. This connectivity of the underlying user-node graph affects the required response of resource-nodes which can be data-ferrying or data-relaying. To handle this more realistic case, a swarm intelligent algorithm based on pheromone influencing that is capable of displaying both these behaviours, without global knowledge of the graph, is developed. It is observed that resource-nodes operate in a spatially local way, leading to a globally emergent behaviour of resource-node coordination that facilitates both data-ferry and data-relay functions. A detailed latency model of the graph is then developed based on post-mortem observation of this emergent behaviour. Additional experimentation introduces the constraint of a finite ferry buffer and examines the utility of flocking emergent behaviour in creating a collective ferry buffer to handle excess data traffic. This chapter is based on the papers:

Bradley Fraser, Andrew Coyle, Claudia Szabo, and Robert Hunjet. “**Adaptive Swarm Control for Mobile Resource Placement in Wireless Ad-Hoc Networks**”. In: *2019 IEEE 20th International Symposium on “A World of Wireless, Mobile and Multimedia Networks” (WoWMoM)*. June 2019, pp. 1–6;

Bradley Fraser, Andrew Coyle, Robert Hunjet, and Claudia Szabo. “**An Analytic Latency Model for a Next-Hop Data-Ferrying Swarm on Random Geometric Graphs**”. In: *IEEE Access* 8 (2020), pp. 48929–48942; and

Bradley Fraser. “**Data Ferry Flocking for Bulk Information Transfer Under Ferry Buffer Constraints**”. In: *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*. Nov. 2018, pp. 1–6.

The reader is directed to [Appendix A](#) for signed Statements-of-Authorship for each of the publications listed above.

5.1 INTRODUCTION

Dynamic-path data-ferrying gives data-ferries freedom in choosing their next user-node to visit and typically employ strategies where that user-node target is either periodically updated, or updated in response to a change in conditions such as the ferry buffer state. As opposed to static-path data-ferrying of the previous chapter, where routes are generated based on the physical topology of the network being serviced, techniques that use buffer statistics to choose the next target have the advantage of being responsive to traffic conditions. Routes are typically not planned any further in advance than the ‘next-hop’ since the buffer state may change considerably after traffic exchange at that node, which in turn may affect the next next-hop decision.

Along with data-ferrying, data-relaying is another technique to provide and improve communications within disconnected wireless networks. Rather than delivering traffic to disconnected graph components, data-relaying is used to permanently join (or bridge) two or more disjoint graph components; reducing the number of graph components and improving overall network connectivity. Data-relaying obviously reduces packet latency considerably compared to data-ferrying given that the time taken for ferry movement dominates that of wireless transmission used in data-relaying. However, once in a data-relaying state, the node can no longer transfer data to other disconnected network components and it either must choose to switch behaviours or rely on swarm team members to service those components. Nodes that can perform both functions are referred to with the general term *resource-node*, reserving data-ferry and data-relay for nodes performing those specific functions.

The requirement for data-ferrying and data-relaying services changes with the connectivity of the underlying user-node network graph. A useful tool in modelling such networks is the **RGG** and the reader is directed to **Section 3.2** for an introduction to **RGGs** and their connectivity properties. Using **RGGs** as a network model, an adaptive algorithm that provides both data-ferrying and data-relaying is developed and mathematically modelled below.

5.1.1 *Algorithm*

The next-hop decision rule determines which user-node a resource-node should visit next by inspecting its current buffer state. That is, given a set of resource-nodes $\mathcal{R}$, then each resource-node $i \in \mathcal{R}$ decides on the next user to visit by evaluating the value function

$$v_j = \frac{\sum_{k \in \mathcal{B}_i(j)} w_k}{\|x_j - x_i\|}, \quad (5.1)$$

where v_j is the value in visiting user-node $j \in \mathcal{U}$, w_k is the waiting time of packet k destined for node j in node i ’s buffer $\mathcal{B}_i$, x_i is the two-dimensional position vector of node i and $\|\cdot\|$ is the Euclidean norm. Value functions that inspect traffic buffers are a common approach to next-hop decision making in the data ferrying literature [PLX12b; SMT13; Wu+12; CY05; CY07]. Note

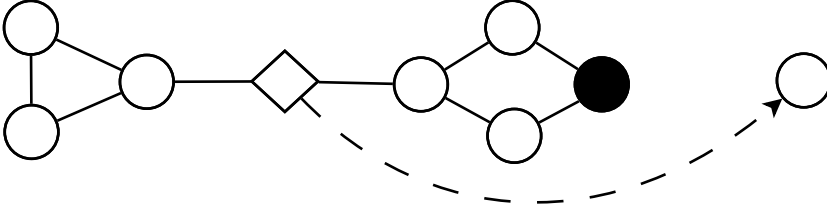


FIGURE 5.1: The filled agent becomes a black hole sink for data destined for the isolated agent if a single resource agent (diamond) permanently relays (or ferries) between the left and middle clusters. To overcome this issue, the resource agent does not forward traffic for the isolated agent but instead visits it directly (dashed trajectory) when a route is not available.

that finding the optimal value function is out of scope and the reader is directed to the references for investigations of this nature.

Since this control rule is data-driven, only those nodes that are packet destinations will be visited. This can be rectified through the additional rule of visiting nodes if a timer expires regardless of its traffic queue. In this work however, we assume at least some traffic is generated for all nodes. This is a reasonable assumption in, for example, tactical military networks, where the primary purpose is to ensure that the **Position-Location Information (PLI)** of each node is continually disseminated throughout the network.

The routing rule describes how nodes interact with each other when they come into communication range; a modified version of geographic routing is used [Sto02; MWH01]. In geographic routing, data is exchanged between nodes based on their geographical distance to the destination. When choosing the next-hop destination from any connected peers, the neighbor that is closest to the destination is selected. This rule is implemented as part of the adaptive swarming algorithm with a caveat. The recipient node must either be:

1. The final packet destination (or have a route to the destination); or
2. Another resource-node (or have a route to another resource-node).

This prevents ‘black hole’ data sinks (see Fig. 5.1) from forming by only allowing the sending resource-node to deliver the packet to the intended recipient or pass it to a peer resource-node. While an intermediate user-node may be physically closer to the destination, the sending resource-node has no guarantee that another resource-node exists in the system to continue the delivery.

User-nodes employ the same geographic routing rule but without the black hole caveat. Traffic then builds up in user-nodes on the edges of graph components which helps resource-nodes to localize their behavior to component gaps. User-nodes will also transmit all current packets in their buffer to any connected resource-node regardless of distance to the recipient. This solves some edge cases where for example a resource-node would need to approach a solitary user-node from the correct direction in order to facilitate traffic flow imposed by the geographic routing.

Algorithm 2 presents pseudocode for the operation of the resource-node. During each (simulation) control loop, resource-nodes calculate their next-hop and move towards it. The next-hop decision is recalculated whenever the buffer state changes.

Algorithm 2 Adaptive Ferry-Relay Algorithm

```

1 procedure USER-NODE
2   while true do
3     if User in contact with User then
4       if route exists to closer user or route exists to recipient then
5         sendFurtherTraffic()
6         receiveCloserTraffic()
7     if User in contact with Ferry then
8       sendAllTraffic()
9 procedure RESOURCE-NODE
10  while true do
11    updateTarget ← false
12    if Ferry in contact with User then
13      if route exists to recipient then
14        sendTraffic()
15        receiveTraffic()
16        updateTarget ← true
17    if Ferry in contact with Ferry then
18      if route exists to closer Ferry or route exists to recipient then
19        sendFurtherTraffic()
20        receiveCloserTraffic()
21        updateTarget ← true
22    if updateTarget then
23      updateAndMoveToTarget()

```

Figure 5.2 presents two scenarios where the agents produce collaborative behaviour. In the top subfigure, both resource agents are able to bridge between the three user agent clusters and produce a stationary, stable solution. In the bottom subfigure, the distance between clusters is such that resource agents must ferry between clusters. Due to the black hole rule, ferries do not only move between their two respective clusters. The actual behaviour depends on whether there is a route to resource-node Right or not, when resource-node Left comes into contact with the middle cluster (and vice-versa). If there is, resource-node Left can forward on traffic for recipients on that side of the network and move back to its far cluster, safe with the knowledge that data destined for agents on the other side of the network are with a resource-node more local to the destination(s). If there is not a route, it must personally travel to the recipient(s) (or their component) since it cannot be sure another resource agent is there to finish the delivery.

5.1.2 *Locality*

The heuristic algorithm rules produce coordinated behaviour at the global level via a process known as stigmergy [DBT00]. Stigmergy refers to indirect communication through the environment by altering it in some way that will influence individuals at a later time. In this case, the stigmergic variable is network traffic that, in some sense, acts like a pheromone to guide resource-node behaviour. It does this in two ways: firstly, by adding and removing

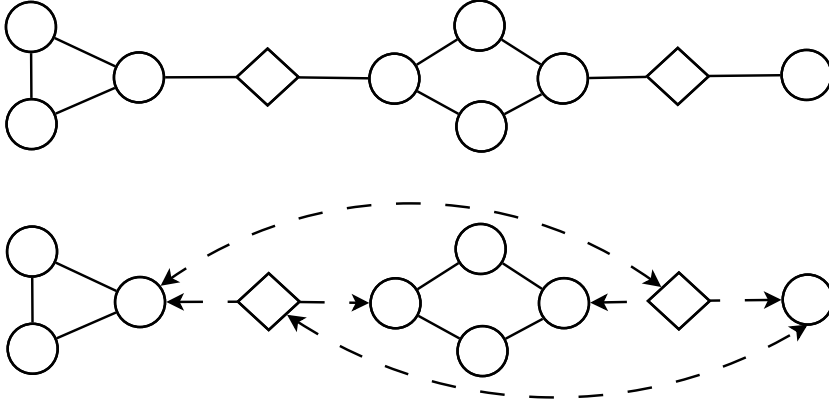


FIGURE 5.2: Two scenarios with two resource agents (diamonds, Left and Right) conducting relaying behaviour (top) and ferrying behaviour (bottom).

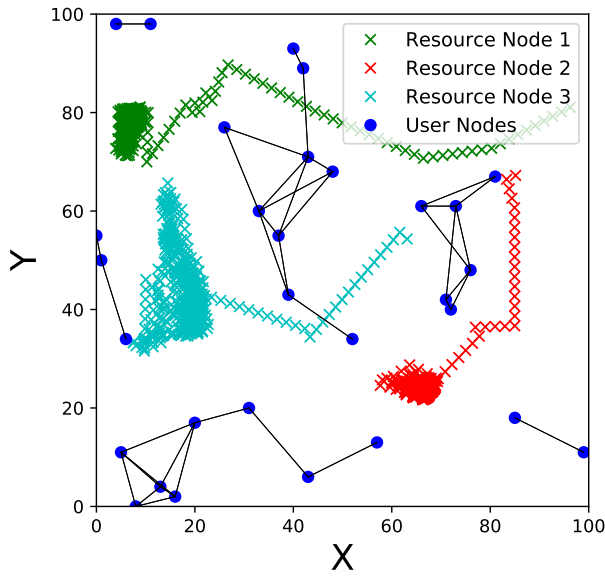


FIGURE 5.3: Results of the algorithm with a 30 node (blue) RGG, $\mathbb{E}[D] = 3.5$. Crosses indicate the location histories of the three resource-nodes.

data packets at user-nodes, resource-nodes communicate (influence) which user-nodes need visiting at a later time; and secondly, handing off traffic in a geographic way allows resource agents to operate within their current local area. This produces a seemingly planned behaviour where resource-nodes appear to be assigned to different areas. For example, Fig. 5.3 shows the location histories of a 30-node RGG with three resource-nodes ($\mathbb{E}[D] = 3.5$). The resource-nodes start from similar locations to find stable operating areas that connect the graph.

The concept of passing user-node data to a resource-node that is more local to the recipient's location is an important one, and we use the descriptor *locality* of a resource-node to refer to the size of the neighbourhood that it operates in. Therefore, a resource-node displaying maximum locality is stationary and is unequivocally displaying data-relaying behaviour. Less than maximum locality represents data-ferry behaviour with the degree of locality indicating the size of its operating area. This concept of resource-

node locality is used later to analyse resource-node behaviour at different connectivity levels.

5.2 EXPERIMENTAL ANALYSIS

We investigate the algorithm from two perspectives: first from a pragmatic performance point of view where the algorithm's ability to transfer data around a disconnected network is evaluated, and second from an adaptability perspective. Specifically, we quantify the algorithm's inherent adaptability to the environment as shown below.

Firstly we examine the algorithm's ability to transfer data around a disconnected network while minimising packet latency. This problem is related to the **TSP** [App06] where the aim is to find the minimum distance tour that visits all nodes within a graph. It is noted that static routes such as **TSP** solutions cannot be the optimal solution due to the nature of **MANET** traffic [WL10]. Visiting a node for which there is minimal traffic wastes resource time and planning for the next next-hop is sub-optimal since next-hop traffic may affect that plan. We note that these static next-hop solutions are more commonly found in networks with all-to-one style traffic profiles such as is found in some wireless sensor networks. Nevertheless, the **TSP** solution provides a well understood benchmark for comparison and is used here for that purpose.

The original **TSP** must be modified to account for the range associated with each node and the connectivity of the graph. The optimal solution was approximated in this case by using a component-collapsed approach. In this approach, a **TSP**-optimal solution was computed using the Concorde **TSP** solver [Coo15] for component centroids. That is, each connected-component was collapsed into a single centroid node from which the **TSP** solution was generated. The centroid tour was then expanded back into the original components with the actual tour length consisting of the sum of the shortest distances between the components and the shortest distance between entry and exit nodes within a component. Although this approach is an approximation, it provides a straight forward implementation for comparison. Rigorous approximation algorithms for the **TSP** with neighbourhoods can be found for example, in [DM03]. Figure 5.4 plots the **TSP** tour length for the component-expanded **TSP** graphs consisting of a 100 node **RGG** on a 100-unit square.

Given that the proposed algorithm uses multiple resource-nodes to disseminate traffic, a multiple **TSP** solution was required. The multiple-**TSP** can be formulated where individual tours do not intersect, or intersect at a common graph vertex. Both of these options were considered unsuitable since data cannot be transferred to all network nodes in the former, while the latter would introduce a bias in the latency results depending on the location of the intersecting node. Instead, a single **TSP** tour was serviced by multiple resource-nodes that were optimally separated around the **TSP** tour and moved in the same direction and speed such as in Chapter 4 [Cha+11; FHC17].

In this approach, the shortest possible waiting time between when a packet is generated at a user-node and when it is picked up by a resource-

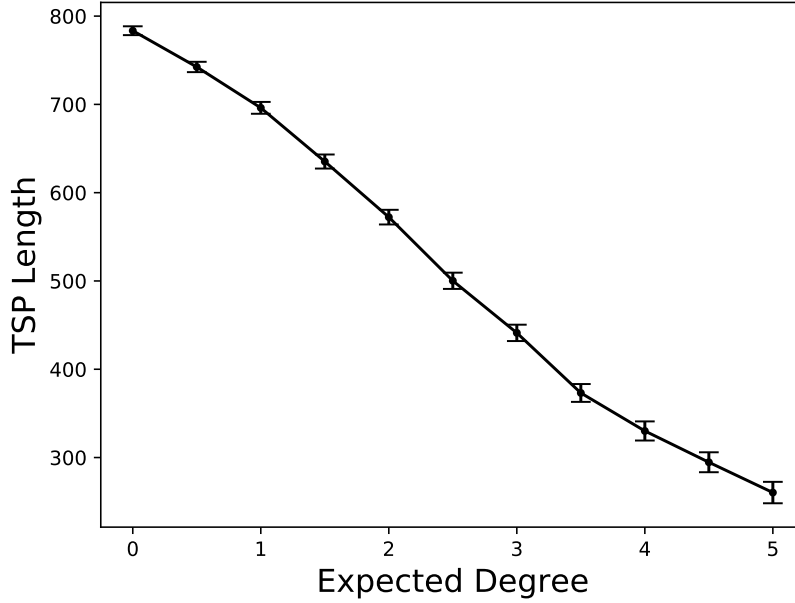


FIGURE 5.4: TSP tour length with increasing graph connectivity. Error bars indicate a 95% CI over 100 graph instances.

node is zero, occurring when these events are simultaneous. If l is the tour length and $n = |\mathcal{R}|$ is the number of resource-nodes, the waiting time is then $\frac{l}{n}$. Assuming resource-nodes move at one unit per time step, the smallest delivery time for a packet is $r + \epsilon$ for some small $\epsilon > 0$ and occurs when the packet destination and next-hop are one and the same and is just beyond the communications range of the source node. Similarly, the longest delivery time occurs when the destination node is the node ‘behind’ the source node in the tour and is just beyond communications range. The delivery time is then $l - (r + \epsilon)$. The latency L is approximated as the mean of the shortest and longest delivery times and the waiting time, giving:

$$L = \left(\frac{1}{2}\right) \left[l - (r + \epsilon) + \frac{l}{n} - (r + \epsilon) \right] \approx \frac{l}{2} \left(1 + \frac{1}{n} \right) - r \quad (5.2)$$

Using this analysis and the **TSP** solution, benchmark latency values were generated for comparison to the proposed algorithm.

Simulation experiments were conducted using **MASON** in a 100-unit square field. Results were averaged over 100, 100-node **RGG** instances with individual simulations running for 2000 discrete time-steps. User-nodes uniform-randomly generated a packet for another node each simulation time-step and both user-nodes and resource-nodes had the same communications range calculated from the expected degree (see Eqn. 3.13). Latency values were calculated from delivered packets that were transported by one or more resource-nodes. The fixed-range communications model and routing model from Section 3.1.2 and Section 3.1.3 respectively, were used.

5.2.1 Results

Figure 5.5 shows the latencies achieved for the two algorithms for various graph connectivities and resource-node numbers. The adaptive algorithm

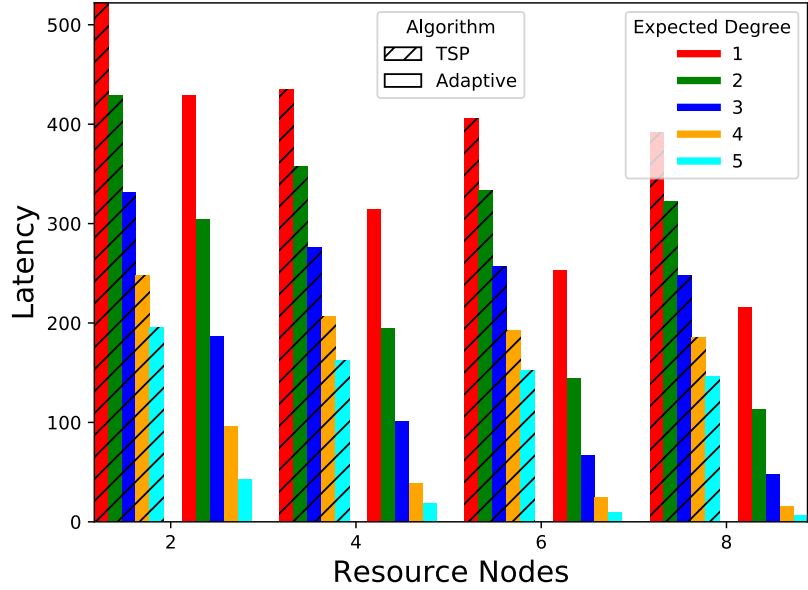


FIGURE 5.5: Latency results for the three algorithms over various graph connectivities. Confidence interval error bars (not shown) are minuscule due to the large number of latency samples used.

significantly outperforms the **TSP** solution as expected and is consistent with similar results [WL10]. The **TSP**-based solution shows a fast asymptotic convergence to a non-zero latency with increasing resource-nodes. This is unsurprising since from Eqn. 5.2, the mean latency is $\frac{L}{2} - r$ as $n \rightarrow \infty$ which represents the case where there are so many ferries, packets experience no waiting time before being picked and the only delay is the transfer time. This approach does not make use of ferry-to-ferry transfer to speed up delivery.

As the expected degree of the **RGG** increases, latencies in both algorithms decrease. However, at high expected degrees, the latency of the proposed algorithm is nearly zero in the scenarios with many resource-nodes, suggesting that the resource-nodes are able to coordinate themselves to produce nearly full connectivity. To achieve this, resource-nodes would move from a ferrying behaviour between many user-nodes (or components), to localised ferrying between few user-nodes to stationary data-relaying. Note that full graph connectivity in this case occurs *almost surely* at an expected degree of $\ln(n) = \ln(100) \approx 4.6$.

It is noted that it is somewhat unfair to compare the algorithm to the **TSP** solution in high degree cases as it does not have the ability to stop the salesmen in places where they could act as relays. This however is the crux of the problem, an algorithm that is adaptive to the environment is required.

The geographic routing within the algorithm assists the mobile resource-nodes in acting locally by connecting traffic flow with spatial movement. The locality of a resource-node is therefore dependent on:

- Node density/connectivity of the graph since the amount of expected movement required to come into contact with a destination node is different in sparse and dense graphs; and the
- Number of resources, since even in high connectivity graphs with few

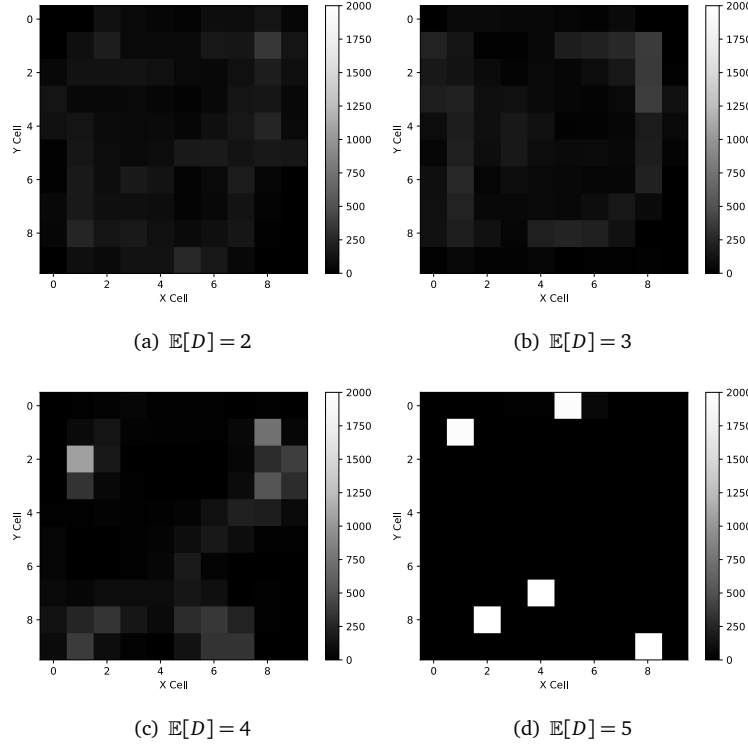


FIGURE 5.6: The spatial history of resource-nodes during a simulation run of five resource-nodes at different connectivity levels. As connectivity increases, resource-nodes spend more time in fewer locations and become more local.

components, a single resource-node would still be required to move between components to prevent the black hole problem.

Data-ferry and data-relay behaviours lie on opposite ends of the resource-node locality spectrum and hence node locality can be used to indicate what behavioural state a resource-node is in. The hypothesis (and most useful outcome) is that sparse networks use predominantly data-ferrying behaviour while dense networks use predominantly stationary data-relaying behaviour (dependent on resource-node availability). To test this, the 100 unit square field was discretized into a 10×10 unit grid of cells and resource-node locations were recorded with respect to the grid during algorithm operation.

Figure 5.6 shows the spatial history of a swarm of five resource-nodes during a full simulation for $\mathbb{E}[D] = 2$ to $\mathbb{E}[D] = 4$ on the same RGG. It can be seen that low connectivity produces a relatively uniform distribution of the resource-node locations but becomes more localised as connectivity increases. In the case of a single resource (not shown), higher connectivity scenarios produce a regular well-defined pattern which represents the ferrying path between the few disconnected network components.

This spatial representation can be viewed as a histogram where each cell represents a bin. This allows the comparison between histograms using an appropriate distance measure. The χ^2 histogram distance [PW10]

$$\chi^2(X, Y) = \frac{1}{2} \sum_{i=1}^n \frac{(x_i - y_i)^2}{(x_i + y_i)} \quad (5.3)$$

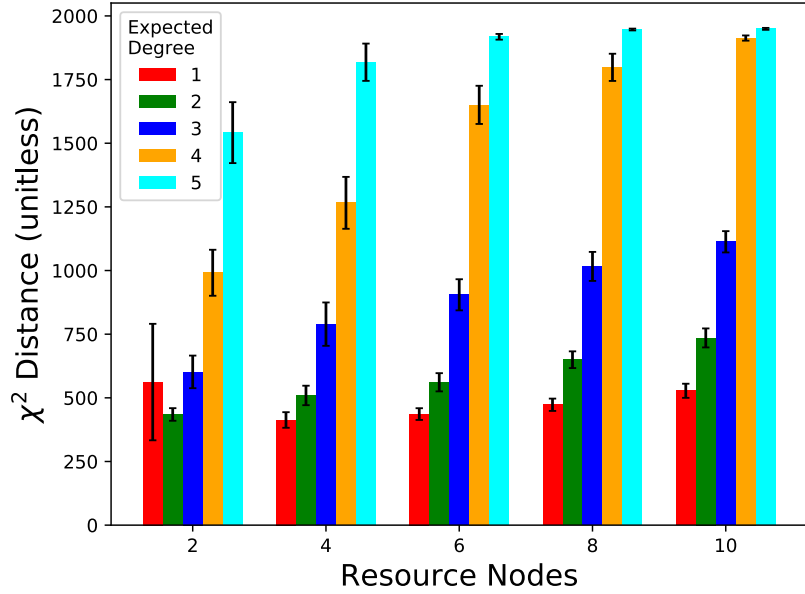


FIGURE 5.7: χ^2 histogram distance deviation of resource-node locality from the discrete uniform distribution. Error bars represent a 95% CI.

where x_i and y_i are the n bin values for each histogram X and Y , is one such metric. A uniform histogram with bin values of 2000 iterations/100 bins = 20 was used for comparison. This histogram represents the case where the resource-node spends an equal amount of time in each cell and is assumed to be the case most representative of a pure data-ferrying solution.

Figure 5.7 plots the χ^2 histogram distance of the spatial distribution of resource-nodes from the uniform reference distribution. In general, low connectivity graphs show the smallest deviation from the reference uniform histogram, indicating a large amount of movement of resource-nodes. As connectivity increases, the distance metric also increases indicating local behaviour. These results show that the same algorithm is able to produce both spatially local and general behaviour to transfer network traffic depending on connectivity and the number of resources available. Moreover, these results are achieved without knowledge of the network topology (aside from user-node locations) or traffic conditions.

5.2.2 Traffic Adaptability

User-node density is one aspect of the environment that drives the emergent behaviour in the above algorithm. Another important driver is user-node traffic generation. An investigation into how the user traffic recipient distribution affects node behaviour (through the next-hop decision rule) exists in the literature [SMT13]. We qualitatively confirm this result here.

To analyse the adaptability of the algorithm to the traffic stimulus, the above experiments were repeated with a high data rate flow between a randomly chosen source and destination pair. It is expected and desired that resource-node locality analysis demonstrates patterns that favour the high data rate pairs.

Figure 5.8 shows the resource-node locality response for 10 resource-

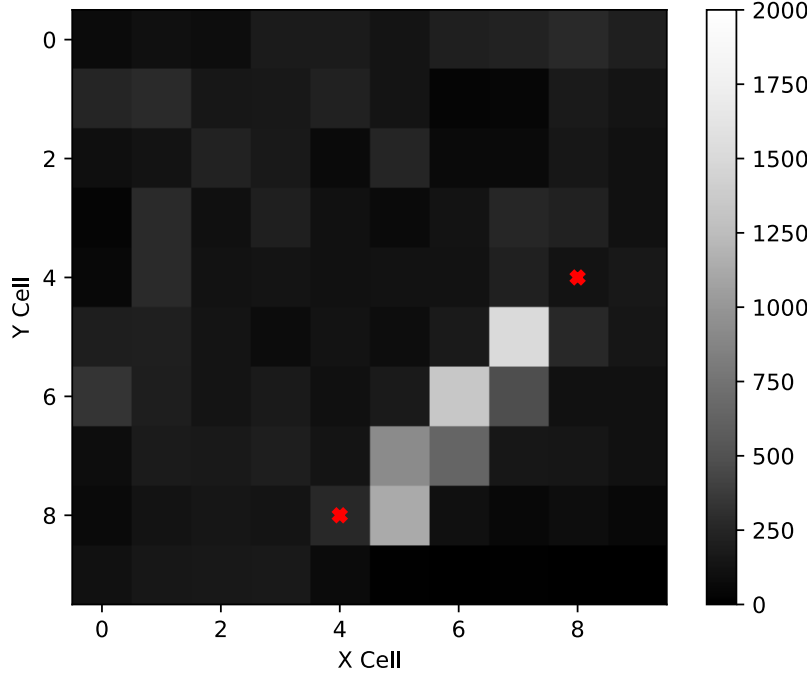


FIGURE 5.8: Aggregate resource-node locality of 10 nodes and a single high data rate pair (red crosses). User-nodes within the RGG had an expected degree of $\mathbb{E}[D] = 1$.

nodes in a low connectivity **RGG**. Red crosses indicate the cells containing the high traffic rate pair. The lighter cells indicate a commonly used path between the high traffic pair used by the resource-node, suggesting that the algorithm also adapts to traffic requirements. As expected, it was also observed that resource-nodes favour constructing a route between components containing high traffic pairs as connectivity increased.

Cooperative behaviour between several resources nodes was also observed in servicing the high traffic rate pair. This occurred when resource-nodes independently encountered a high traffic producer causing the next-hop value function to suggest to immediately head to the other user-node in the pair. It then often encountered a different resource-node doing the same thing in the reverse direction. This behaviour can be described using the chain-relay model [HB06] and is analysed below in **Section 5.4.1**.

5.2.3 Discussion

In this section we attempt to describe the behaviour displayed within the proposed algorithm using known formulations of collective and emergent behaviour.

[GGT07] suggests that the only way for a collective to adapt its behaviour is through the modulation of individual behaviours in response to perturbations, and that asking the questions:

1. “How does the perturbation modulate the individual behaviours of the interacting animals?”; and
2. “How does the modulation of the individual behaviour shape the properties at the collective level”?

can be used to identify the mechanisms of collective adaptation. To answer the first question in the node density context, it is helpful to consider the follow up question “Does it [the perturbation] stimulate animals to perform a particular behaviour or task, or does it prevent them from doing so?”. User-node coverage areas act as a kind of exclusion zone for resource-nodes in that once resource-nodes come into contact with them and traffic is exchanged, all motivation to be there is removed. In the pheromone analogy, the pheromone intensity drops immediately to zero, allowing pheromones for other user-nodes to takeover. This is true for network components too and these connectivity structures limit the operating area of individuals. Or equivalently, their locality increases. The perturbation, then, modulates individual behaviour by affecting where individuals wish to move. The second, and in our case easier, answer is that at the collective level, individuals become more or less local to an area depending on graph connectivity and produces the desired data dissemination behaviour.

[Hol07] suggests that emergence within multi-agent systems usually consists of top-down feedback from the global to local level. Interactions between agents are thought to react to global influences, yet at the local-level give rise to patterns observable at the global-level. These resultant patterns are an emergent phenomenon and are not specified by the agent interaction rules. It is further suggested that this type of emergence can be classified according to the net effect of the feedback. When the feedback from the global to local level is net negative, it imposes a constraining or moderating influence on the actions of the agents and is thought to generally result in a stable solution. Pheromone following behaviour is an example of this behaviour where the global feedback is in the form of pheromone evaporation which removes unneeded paths from the colony history, constraining the paths available for ants to follow. In a similar way, the global influence of network density constrains the movement options of the resource agents within the network, leading to stable behaviour, particularly in high expected degree graphs. Under this definition of emergence, dynamicism of the user-nodes’ location would introduce a positive feedback into the system leading to unstable behaviour.

5.3 MATHEMATICAL MODEL

Swarming systems are often difficult to model mathematically due to the number of individuals and the dynamics involved. Accordingly, in order to model the packet latency of the algorithm we make the simplifying assumption that routing between user-nodes is disabled, or equivalently, $\mathbb{E}[D] = 0$. Communication is only allowed between user-nodes and resource-nodes, and between resource-nodes. While this assumption is simply to facilitate model development, such disconnected network topologies may be representative of:

- Extreme RF denial or harsh transmission situations; and
- Situations where nodes only wish to communicate with resource-nodes. For example, to save power in wireless sensor networks, or in low detection probability networks.

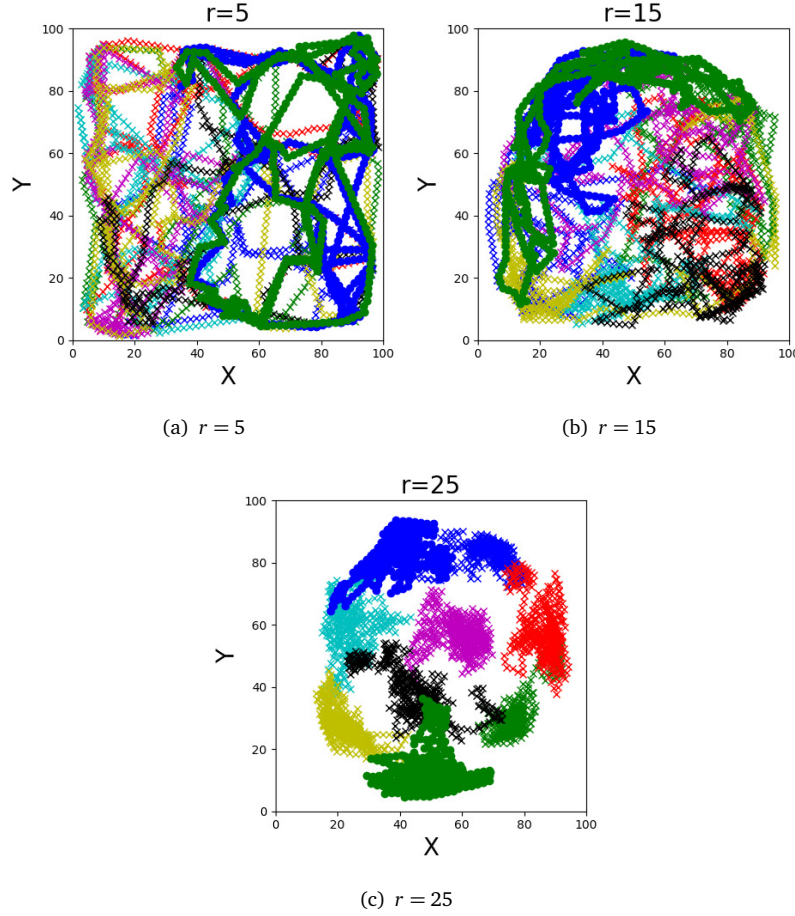


FIGURE 5.9: Resource-node history traces for a 50 user-node RGG (not shown) and 9 resource-nodes ; grid-like behaviour is generated as resource-node range increases.

It is also a conservative assumption, and hence may make sense for allocation of resources.

With the above assumption in place we observed that the resource-nodes had a tendency to generate a grid-like structure as resource-node density increased. As the communication range increases, resource-nodes localize to progressively smaller operating areas. This is demonstrated in Fig. 5.9. We thus begin by developing a model based on this behaviour.

5.3.1 Initial Model

Assume that an RGG of user-nodes is divided into a square grid of k cells such that there are $\sqrt{k}$ cells along the grid side. Each cell represents a localized operating area and contains a single resource-node as in Fig. 5.10. Using this grid layout and the assumption that user-node forwarding is disabled, we develop an analytic expression for the packet delay between user-nodes.

Mean Wait Delay

We begin by approximating the algorithm behaviour within a cell as following a TSP tour [App06]. The current best bounds of an optimal TSP tour $T^*(n)$

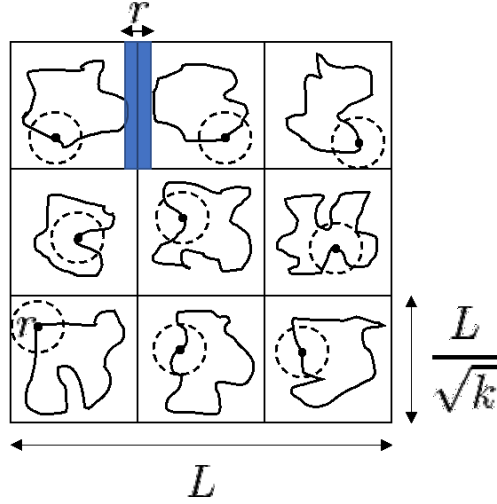


FIGURE 5.10: The model; a grid of k cells each with their own resource-node following a TSP solution with radius r . The blue strip represents a region where adjacent resource-nodes can perform a hand-off.

on a random set of points in a unit square is given by [Ste15]

$$0.63\sqrt{n} \leq T^*(n) \leq 0.92\sqrt{n} \quad (5.4)$$

where n is the number of nodes in the unit square. In order to be representative of the typical case, we use the value

$$T^*(n) = 0.70787\sqrt{n} \quad (5.5)$$

derived from extensive simulation [VJ97]. If there are N total user-nodes then there are

$$n_{\text{cell}} = \frac{N}{k} \quad (5.6)$$

nodes in each cell. Assuming a constant movement speed of one and grid size of L , the mean TSP time, or cell time, is given by

$$t_{\text{cell}} = 0.70787\sqrt{n_{\text{cell}}}\frac{L}{\sqrt{k}}. \quad (5.7)$$

If r is the radius of the communications range, the ratio of the resource-node coverage area to its cell is given by

$$A_c = \frac{\pi r^2}{\left(\frac{L}{\sqrt{k}}\right)^2} = \frac{\pi r^2 k}{L^2} \quad (5.8)$$

where $0 < A_c \leq 1$. We approximate the mean time user-nodes must wait for a resource-node to visit as $\frac{t_{\text{cell}}}{2}$ modified by the coverage ratio to give

$$t_{\text{wait}} = \frac{t_{\text{cell}}}{2}(1 - A_c). \quad (5.9)$$

Intermediate Cell Time

We now estimate the time packets spend in intermediate cells on their way to their destination cell. Assume that if two resource-nodes are within a strip of size

$$\frac{r}{2} \frac{L}{\sqrt{k}} \quad (5.10)$$

of a cell boundary they can hand-off data to each other. The proportion of this hand-off area per cell (ignoring overlap when there is more than one hand-off edge) is

$$A_h = \frac{\frac{rL}{2\sqrt{k}}}{\left(\frac{L}{\sqrt{k}}\right)^2} = \frac{r\sqrt{k}}{2L}, \quad (5.11)$$

and the mean number of hand-off edges in a grid (cells can have 2 to 4 hand-off edges depending on their location within the grid) is given by

$$h = \frac{4(\sqrt{k}-1)}{\sqrt{k}}. \quad (5.12)$$

Therefore the hand-off area ratio is

$$h_r = A_h h \quad (5.13)$$

where $0 < h_r \leq 1$. The approximate intermediate cell time is then

$$t_{\text{intermediate}} = \frac{t_{\text{cell}}}{2} (1 - h_r). \quad (5.14)$$

Transfer Time

In order to estimate the time taken for a packet to reach its destination cell, we must estimate the number of resource-node hand-offs (or number of intermediate cells traversed). Assuming a uniform-random traffic distribution, the probability that no steps between cells is required because the source and destination are in the same cell is

$$p_{\text{no steps}} = \frac{n_{\text{cell}}}{N} \quad (5.15)$$

where $0 \leq p_{\text{no steps}} \leq 1$. Further assuming shortest-path Manhattan routing between cells, we require the number of steps between cells for data to reach their destination. Consider that along one dimension of the grid there are $(\sqrt{k}-1)$ unit-length paths, $(\sqrt{k}-2)$ two-unit length paths, etc. Then in two dimensions, the combined path length is

$$\sum_{i=1}^{\sqrt{k}-1} 2i(\sqrt{k}-i). \quad (5.16)$$

Now, there are k possible source to destination cell pairs, of which $k - \sqrt{k}$ have a path that is greater than one (source and destination are not in the same cell), so the mean cell path length is

$$\frac{1}{k - \sqrt{k}} \sum_{i=1}^{\sqrt{k}-1} 2i(\sqrt{k}-i) = \frac{\sqrt{k}+1}{3}. \quad (5.17)$$

Accounting for the no step probability, the mean number of cell steps required to reach the destination cell is then

$$s = (1 - p_{\text{no steps}}) \frac{\sqrt{k}+1}{3}. \quad (5.18)$$

To calculate a transfer time we use this result and $t_{\text{intermediate}}$ to get

$$t_{\text{transfer}} = t_{\text{intermediate}}(1 + 2s). \quad (5.19)$$

Latency

Finally, the latency is given by

$$t_{\text{latency}} = t_{\text{wait}} + t_{\text{transfer}}. \quad (5.20)$$

Note that in the special case of $k = 1$, $t_{\text{latency}} \approx t_{\text{cell}}$ where on average packets wait half the TSP length to be picked up and then spend another half of the TSP length being delivered.

This model produces a reasonable estimate for data latency (not shown) but is limited to the high-density case where resource-nodes form a well-structured grid. To make the model applicable to low-density cases too, several improvements must be made.

5.3.2 Model Improvements

The modelling of swarms as physical particles is a useful approach as it provides a statistical approximation of the collective and is scalable with the swarm population size. There are also many theories from the physical sciences from which models can be drawn; for example: from active matter [YTQ08], fluid dynamics [Pim+13], matter states [AP12], and general chemical kinetics [VMP14].

As our problem centres on agent interaction, models of particle collisions are of interest since we can interpret collisions as hand-offs. This removes the need to estimate resource-node hand-offs through the calculation of hand-off edges and opportunities. We first demonstrate how particle collisions can be modelled using statistical mechanics methods and then apply this idea to the system model.

Collision Rate

A well-known approach to deriving the collision rate in a gas is by using the following conceptual equation [AP11]

$$\frac{(\text{collisional volume})(\text{density})}{(\text{time delta})}. \quad (5.21)$$

In three dimensions, a cylindrical volume with a cross-sectional radius of $2r$ where r is the radius of the gas particle is prescribed. The length of the cylinder is simply the particle speed multiplied by the time delta over which the rate is being measured. Any other particle whose origin lies within the cylinder will cause a collision. The density part of the equation is simply the number of particles divided by the containing volume. The time delta over which the rate is measured cancels with that used in the cylinder length if they are chosen to be the same. Applying this approach to our two-dimensional model we obtain the collision rate (or frequency) f_c as

$$f_c = (2r + \pi r^2) \left(\frac{k}{L^2} \right) \quad (5.22)$$

where r is the communication radius of the k resource-nodes and L is the side length of the square operating area. Since gas particles are typically moving at high speeds, the length of the cylinder is quite long leading to a large cylindrical volume and so the semi-sphere caps at the end of the

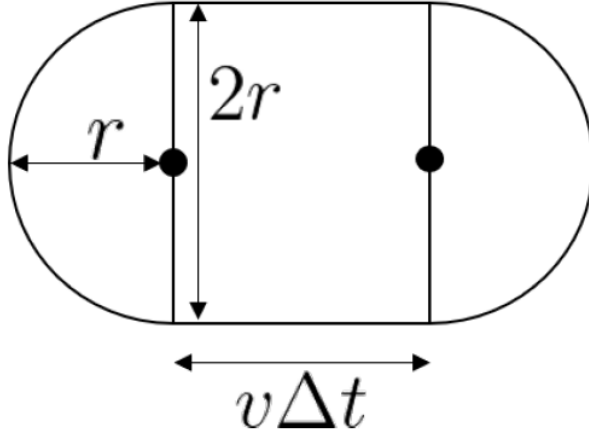


FIGURE 5.11: The collisional area for a single particle of radius r travelling at velocity v for time Δt . Solid circles represent the starting and finishing origins of the particles over the time Δt .

cylinder are usually omitted. In our case where each node moves a distance of 1 in each time step, the collisional area is $2r \times 1 \times 1 = 2r$. However, the combined end-cap area of πr^2 is significant and so is included in the collisional area as in Fig. 5.11. This collision expression is directly dependent on both the communication radius and the number of resource-nodes in the system.

Figure 5.12 plots the simulated collision rate and the analytic expression for $k = 9$ and $L = 100$. The analytic expression matches reasonably well up until near the saturation point where some nodes continually stay within each other's communication range; analogous to the non-physical case of particles occupying the same physical space. That is, there is not enough space for them to 'bounce' off each other. Just before saturation, the resource-nodes form a rough 3×3 grid (see the bottom diagram of Fig. 5.9). Along one dimension in this arrangement, the combined range could be $4 \times 25 = 100 = L$. However, how far apart they are spaced depends partly on where the user-nodes are. For example, in a particular instance, user-nodes may not have spawned right at the very edge of both sides of the field in that slice, meaning that the combined range is not required to be 100 to reach all users. This leads to an inflated collision rate since resource-nodes continually stay within range of each other.

To use collision rate in our model, we find the number of collisions per length L as in Fig. 5.13. The collisional area is then $2rL$ and the probability that a collision occurs is $p = \frac{2rL}{L^2} = \frac{2r}{L}$. Given this probability, we can consider all possible cases of where individual resources may be located; either inside or outside the collisional area. This is approximated using the Binomial distribution as

$$f_c = \sum_{i=1}^{k-1} \min\left(i, \frac{L}{r}\right) \binom{k}{i} p^i (1-p)^{(k-i+1)} \quad (5.23)$$

where $\min\left(i, \frac{L}{r}\right)$ caps the maximum number of collisions to be the minimum number required to transfer data across the length L given a radius r , and $\binom{a}{b}$ represents the a choose b operation $\frac{a!}{b!(a-b)!}$.

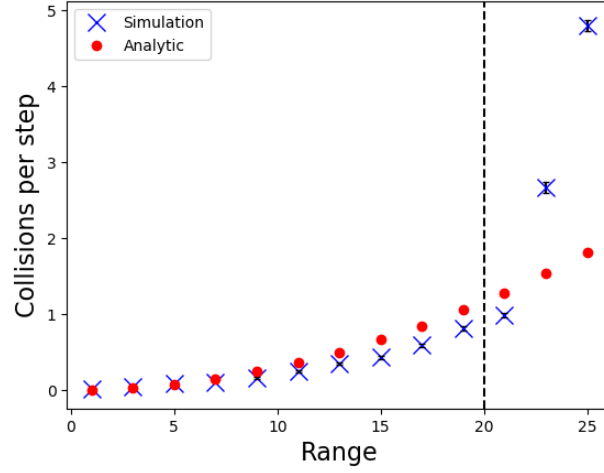


FIGURE 5.12: Comparison of simulated and analytic collision rate for a system of 9 resource-nodes and 50 user-nodes with increasing range r .

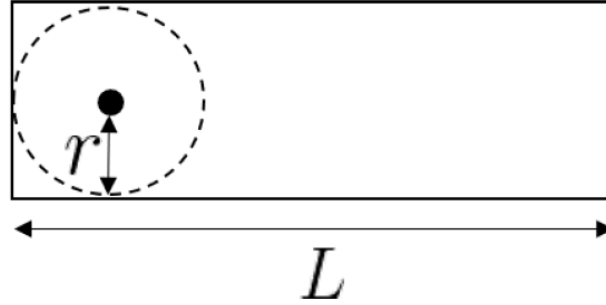


FIGURE 5.13: The collisional area for the length L .

Equivalent Cell Size

The current model estimates the number of user-nodes in each cell by dividing the total number of user-nodes by the number of resources. However, in lower density cases, resource-nodes travel further around the operating area since the time between collisions is higher. The resource-node cells are less well-defined and can overlap. For example, Fig. 5.14 conceptually shows the operating cells of two nodes, blue and green, shaded in their respective colors. User-nodes in the overlapping portion are serviced more often. Note that these cells are not static and the physical area in which they cover changes with time. It is rather the expected coverage at a given time. We denote this expected operating area as the equivalent cell size and estimate it below. Firstly, we can estimate the number of resource-nodes involved in packet delivery as

$$k_{\text{del}} = 1 + f_c \quad (5.24)$$

where $1 \leq k_{\text{del}} \leq \sqrt{k}$ since the number of collisions can be considered as the number of hand-offs. The number of user-nodes per each cell is then

$$n_{\text{cell}} = \frac{N}{k_{\text{del}}^2}, \quad (5.25)$$

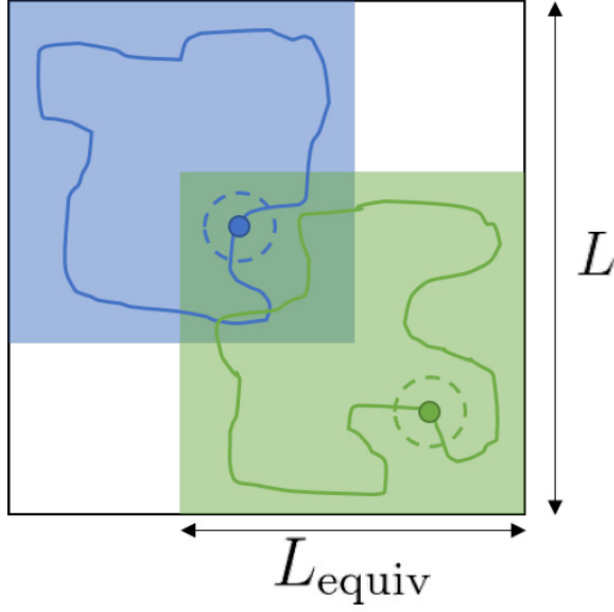


FIGURE 5.14: A low density case where two resource-node cells overlap.

and the number of resource-nodes per cell is

$$k_{\text{cell}} = \frac{n_{\text{cell}}k}{N}. \quad (5.26)$$

We can now estimate the equivalent cell size a resource-node occupies as

$$L_{\text{equiv}} = \frac{L}{k_{\text{del}}} \quad (5.27)$$

with $0 \leq L_{\text{equiv}} \leq L$. To account for the fact that resource-nodes do not need to go to the edges of a cell due to their range, we add a correction of $2r$ to give

$$L_{\text{equiv}} = \frac{L}{k_{\text{del}}} - 2r. \quad (5.28)$$

These new values for n_{cell} and L_{equiv} are used in the calculation of t_{cell} and the mean wait delay is adjusted to be

$$t_{\text{wait}} = \frac{t_{\text{cell}}}{2} \left(1 - \frac{\pi r^2}{L_{\text{equiv}}^2} \right) \frac{1}{k_{\text{cell}}} \quad (5.29)$$

to account for the fact that a user-node may be serviced by multiple resource-nodes. The mean length calculated from Manhattan routing is also changed from $\frac{\sqrt{k}+1}{3}$ to $\frac{\sqrt{k_{\text{del}}+1}}{3}$ and $t_{\text{intermediate}} = \frac{t_{\text{cell}}}{2}$.

5.3.3 Model Performance

To evaluate the model's performance we examine two metrics: the number of resource-node hand-offs and overall packet latency. All simulation results were gathered from 5 trials of 2500 iterations using a 50 user-node, 100 unit square RGG.

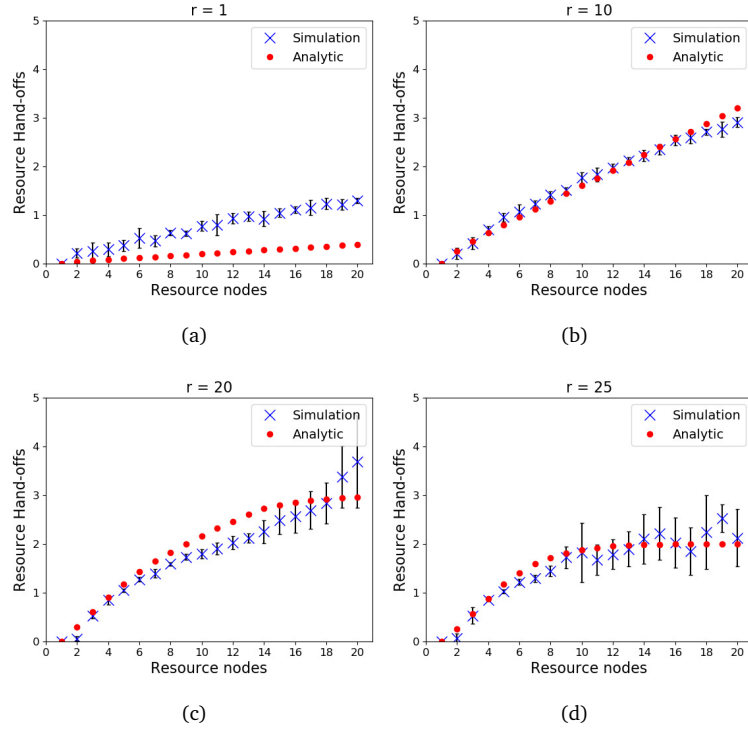


FIGURE 5.15: Analytic and simulated results for the number of hand-offs between resource-nodes. Error bars display a 95% CI.

Resource-Node Hand-offs

Figure 5.15 plots the number of hand-offs between resource-nodes and the number of Manhattan steps calculated in the model. As the communication range increases and the resource-nodes become more structured, it becomes easier to estimate hand-offs because cell boundaries are more well-defined and the estimation becomes more accurate. As mentioned previously in the particle collision calculation when $r = 25$, the hand-offs plateau after 9 resource-nodes since the 3×3 configuration fits into the 100×100 operating area. Extra resources do not help in this situation and the simulation hand-offs begin to vary much more since the extra nodes are still used in the delivery. A similar situation occurs in the $r = 20$ case after 16 resource-nodes.

Ignoring unnecessary hand-offs could be a candidate for algorithm optimization, however a better optimization is to have an understanding of how many resources nodes are required to begin with. The latency model was initially developed to understand the marginal benefit of adding additional resource-nodes to the system so that optimal team numbers could be devised.

We note the larger discrepancy between the model and simulation in the $r = 1$ case and hypothesis that it is due to the correlated nature of the agent movement. That is, more interactions are likely in the simulation since all agents are moving on straight line trajectories between user-nodes as opposed to the random trajectories the particle collision model describes. This of course is true for all results but the effect is hidden as the larger ranges begin to dominate the movement.

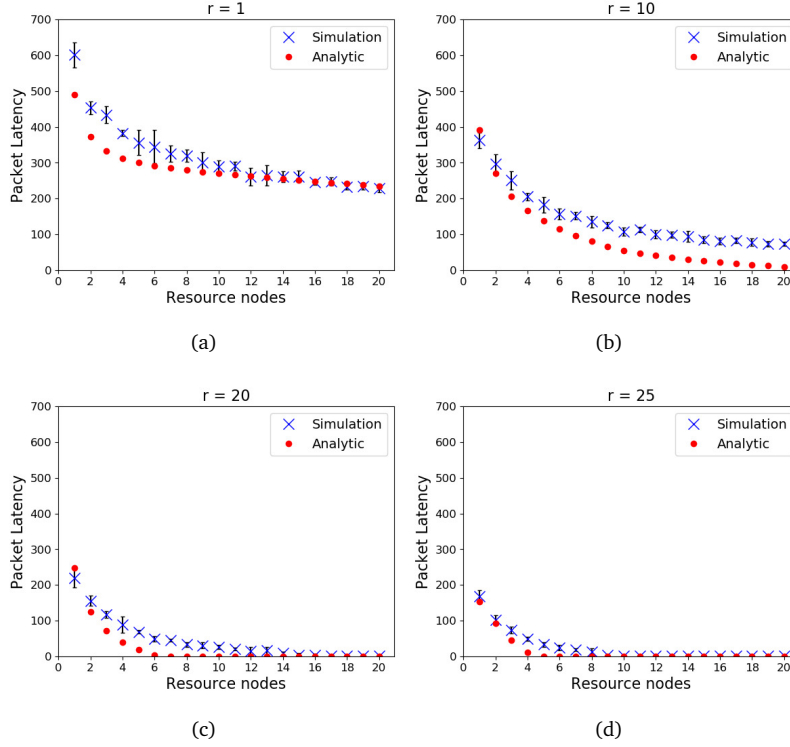


FIGURE 5.16: Analytic and simulated results for the data latency. Error bars display a 95% CI.

Data Latency

Figure 5.16 plots the data latency as measured by user-nodes and the analytic model. Similarly to the hand-off plots, the model becomes more accurate as the radius increases due to the improved structure in the node movement.

Figure 5.17 plots the **Root Mean Squared Error (RMSE)** and **Mean Absolute Error (MAE)** along the resource-node number dimension. Interestingly, the curves are very similar indicating that the magnitude of individual errors are roughly consistent since the **RMSE** is more sensitive to larger deviations [HK06]. This can be interpreted to mean that the model deviates from the simulation by a (near) constant factor. We note that the model is using an optimal **TSP** tour to estimate behaviour within a cell and the constant deviation may be due to this assumption. We can directly compare the next-hop behaviour and the **TSP** bound in the $r = 1, n = 1$ case where the latency is only approximately 20% longer; an interesting result given the unplanned, next-hop approach.

Statistical Tests

Table 5.1 summarizes the statistical hypothesis tests for the model with bracketed values recording the test statistic and associated p – value. The **Kolmogorov–Smirnov (KS)** [Mas51] and **Anderson-Darling (AD)** [SS87] tests test for Type-I errors where the null hypothesis is true but is rejected. Thus they aim to Reject (R) or Not Reject (NR) the null hypothesis that both samples come from the same distribution. In the case of $r = 1$, both tests cannot reject the null hypothesis at a 5% significance level. All the

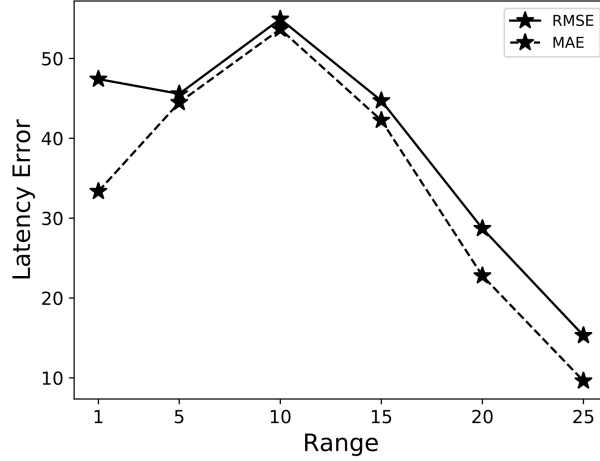


FIGURE 5.17: RMSE and MAE in the data latency between the model and simulation with increasing range r .

TABLE 5.1: Summary of statistical testing. Bracketed values record the test-statistic and p -value of each test.

r	Kolmogorov-Smirnov	Anderson-Darling	Pearson
1	NR (0.25,0.480)	NR (-0.07,>0.250)	(0.98, <0.001)
5	R (0.50,0.020)	R (2.82,0.030)	(1.00, <0.001)
10	R (0.60,<0.001)	R (5.13,0.003)	(1.00, <0.001)
15	R (0.60,<0.001)	R (8.25,<0.001)	(0.97, <0.001)
20	R (0.70,<0.001)	R (13.16,<0.001)	(0.94, <0.001)
25	R (0.80,<0.001)	R (18.01,<0.001)	(0.96, <0.001)

remaining ranges are however strongly rejected.

The Pearson Product-Moment Correlation [FPP07] measures the linear relationship between the two samples with a statistic value of -1, 0 or 1 indicating a negative, zero or positive correlation between samples. The test indicates a high positive correlation between the curves for all ranges with p -values indicating that it would be highly unlikely for two uncorrelated data sets to produce the corresponding test statistic by chance.

In summary, the model latency data follows the general trend of increasing (or decreasing) linearly with the simulation data. The model is able to pass statistical hypothesis tests that are much more sensitive to the shape and location of the data in the $r = 1$ case but not for larger ranges. This is an unsurprising result given the number of assumptions and simplifications made in the model.

5.3.4 Applicability to Higher Connectivity Cases

Initially to build the model, edges between user-nodes were ignored to create RGGs with an expected degree of zero. Figure 5.18 compares the model to simulations where user-node routing was allowed, as per the original algorithm. Topologies were tested with $r = \{5, 10, 15, 20\}$, corresponding to $\mathbb{E}[D] = \{0.39, 1.57, 3.53, 6.28\}$ with the connectivity threshold occurring at $\mathbb{E}[D] = 3.91$ ($r_t \approx 16$), almost surely. Note that $r = 1$ was not tested since the results are nearly identical to the first simulation.

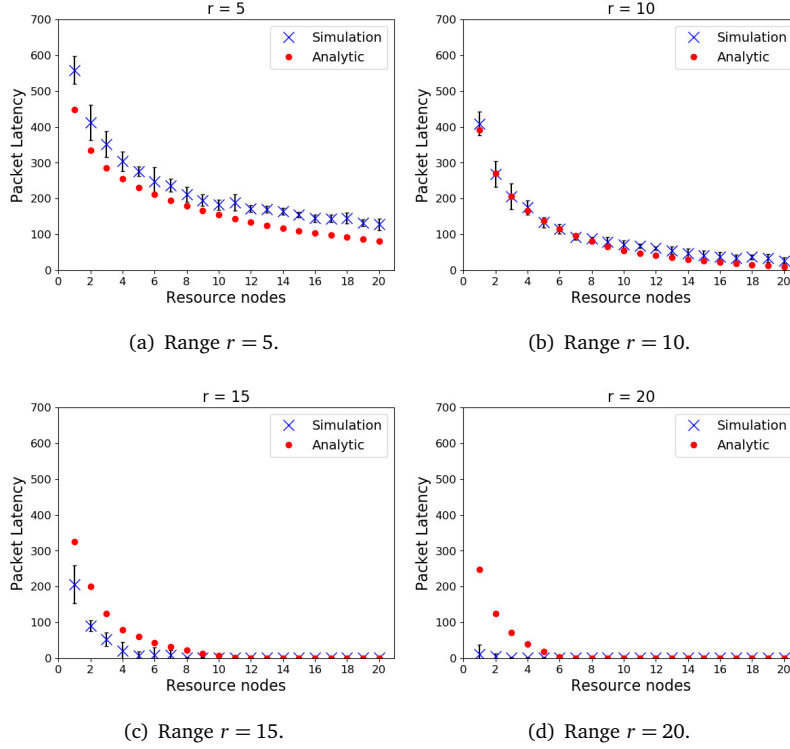


FIGURE 5.18: Analytic and simulated results for the data latency with user-node routing enabled. Error bars display a 95% CI. Note that we show smaller range values for the connected case since the graph quickly becomes connected at higher ranges.

As expected, the simulation results reveal a lower latency as compared to the fully disconnected case since data can be routed through components of connected user-nodes. The magnitude of this effect increases with increasing radius.

The near constant offset between the model and simulation results observed in the fully disconnected case appears to be non-constant here. The curve shapes however are maintained except in the $r = 20$ case which is well beyond the connectivity threshold. Model latency is based on the number of nodes within a cell, however in higher connectivity cases, components containing more than one node effectively appear as a single node due to the routing; something the model cannot account for.

5.3.5 Phase Transitions

In this section we discuss how the model results align with how emergent systems are thought to arise; namely, through the interaction of their constituent members. We also discuss how phase transitions can occur in these systems and look for evidence of those transitions in the latency model.

Emergent behaviour is generated through the interaction of individual entities where those interactions can cause the participants to change their state [Cha11]. Thus, in order for the emergent property to form, interactions are necessary. Indeed, studies have characterized how the emergent property breaks down when those interactions are interfered with [FSH17] (this work is the basis of Chapter 8). The results above demonstrate a similar

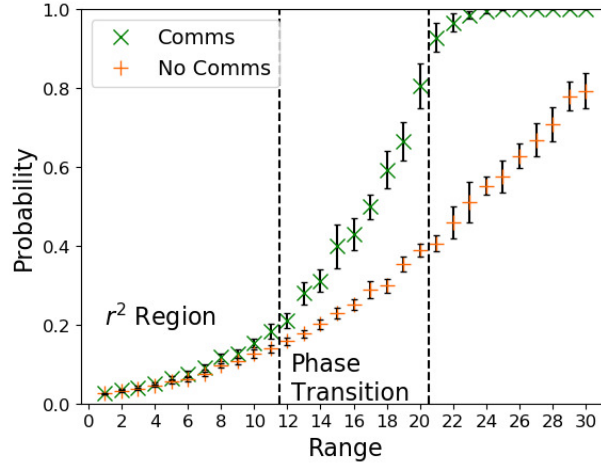


FIGURE 5.19: A phase transition in the probability of data latency being less than 50 time-steps. Error bars display a 95% CI.

phenomenon whereby reducing the opportunity for interaction (by reducing the density via range or node numbers) reduces the development of localized resource-node behaviour presenting as a less efficient algorithm.

Some systems that exhibit emergent behaviour also contain phase changes whereby the system radically changes as a result of an incremental change in some variable [Mog06]; connectivity in random graphs is one such example [KWB01]. In these graphs, an incremental increase in expected node degree causes the graph to radically shift from a disconnected to connected state with high probability. Analogously, in our system we examine the incremental increase of resource-node range and its effect on latency.

Figure 5.19 plots the probability of data latency being less than an arbitrarily chosen threshold of 50 time-steps using 9 resource-nodes. The ‘comms’ data set represents the adaptive swarm resource algorithm while the ‘no comms’ data set represents the same algorithm but with communication between resource-nodes having been disabled. In the case where communication ranges are small, there is little opportunity for nodes to interact and the minor improvement in latency with increasing range is simply a consequence of additional coverage. In this regime, the two algorithms are equivalent. As resource-nodes begin to interact and self-organize, a phase transition occurs wherein the probability of satisfying the latency threshold dramatically increases.

Phase transitions often exhibit a sigmoid shape (or step function in the extreme case). The adaptive algorithm displays a shape somewhat akin to the sigmoid function. However, given that there are 9 resource-nodes and the density is reasonably high (50 user-nodes in a 100unit square), it is highly likely that at least one packet would have a latency less than 50, leading to a non-zero probability even for small ranges; altering the shape from a traditional sigmoid. It was found that the lower the chosen threshold, the sharper the phase transition, and a phase transition was not observed in the case without resource-node communications. Rather, the curve is simply proportional to r^2 and even with a range of 30 that produces a combined coverage ratio of $9 \times \pi \times 30^2 / 100^2 \approx 2.5$ times the operating area, the

probability is not guaranteed; the nodes are not organized, producing a high coverage overlap between them.

While we have seemingly found a phase transition in the data latency, it is actually a different representation of the phase transition found in the connectivity of the resource-node graph. Choosing the latency threshold to be 0 is equivalent to requiring that a path is available to every user-node from every user-node¹. This occurs when the resource-nodes are connected and their combined coverage encompasses all user-nodes in the network.

¹Packets are routed instantaneously if a path is available during a time-step in the simulation.

From a practical point of view, we can use these phase transition graphs in system design if they too could be modelled. For example, the range can be chosen such that it meets an acceptably high probability for a given latency; saving transmission power in this case. We hypothesise that a similar curve would be observed when holding range constant and the resource-node count is the free variable.

5.4 ADDITIONAL INVESTIGATION

Until now, we have assumed that resource-node ferry buffers are large enough to pick up all data from any and all user-nodes in which they come into contact. While digital storage is cheap, here we consider the case where many individual sensors and devices (user-nodes) that may have been disconnected from the larger network for some time, all require data transport by ferry. In this case of mass data transfer, ferry buffer size becomes important. To deal with the case where the amount of data to be transported is greater than that of an individual ferry buffer, flocking data-ferrying is investigated.

5.4.1 Data-Ferry Flocking

Data-ferry flocking sees two or more nodes that are connected from a communications perspective, move in formation for the purpose of ferrying data. Conceptually, it can be considered as a group of n autonomous platforms that act as one while ferrying data between user-nodes. An important property of the collective is that the buffer size is n times larger than an individual. To see how different flocking strategies affect delay and throughput, consider the two user-node ferry problem.

The two user-node ferry problem can be described as simplex data-ferry between two user-nodes separated by distance L and having communications range r_u . The distance a ferry must travel between the user-nodes is then $L - 2r_u$. In the flocking model in 1-dimension, n ferries with buffer b , data rate R , speed v and communications range r_f (where $r_f = r_u = r$ for simplicity) are chained together along the line joining the two user-nodes at the maximum communications range as in Fig. 5.20(c). Due to the formation, the reduced distance to be traversed between two user-nodes is $(L - r(n + 1))$.

Following the analysis from [HB06], the throughput T experienced by the receiver is calculated as $T = \frac{Bb}{t_{\text{cycle}}}$ where B is the number of buffers, of size b , delivered per cycle and t_{cycle} is the cycle time. The cycle time from sender to receiver and back is composed of ferry travel time and data

transmission time (propagation and other queuing are ignored). Note that simultaneous transmission is assumed and so (off-)loading of the formation requires n transmissions of time b/R to transfer $B = n$ buffers (out-of) in-to the ferry flock. Therefore,

$$t_{\text{cycle}}^{\text{formation}} = 2 \left(\frac{L - r(n+1)}{v} + \frac{nb}{R} \right). \quad (5.30)$$

Throughput can then be calculated as

$$\begin{aligned} T^{\text{formation}} &= \frac{nb}{2 \left(\frac{L - r(n+1)}{v} + \frac{nb}{R} \right)} \\ &= \left[2 \left(\frac{L - r(n+1)}{vnb} + \frac{1}{R} \right) \right]^{-1}. \end{aligned} \quad (5.31)$$

The average delay is calculated as the average of best and worse case delays experienced by the packets. The best case delay occurs for the packet that is loaded last. It experiences zero loading delay, the travel time from sender to receiver and off-loading delay, so

$$d_{\text{best}}^{\text{formation}} = \frac{L - r(n+1)}{v} + \frac{nb}{R}. \quad (5.32)$$

The worst case delay occurs when a packet is loaded first, being generated just as the ferry leaves. The buffer must wait a full cycle time before departing on the ferry as well as the travel time from sender to receiver. There is no off-loading delay since it is first in the queue. Therefore, the worst-case delay is given by

$$d_{\text{worst}}^{\text{formation}} = 3 \left(\frac{L - r(n+1)}{v} \right) + 2 \left(\frac{nb}{R} \right) \quad (5.33)$$

and the average delay is then

$$d_{\text{avg}}^{\text{formation}} = 2 \left(\frac{L - r(n+1)}{v} \right) + \frac{3}{2} \left(\frac{nb}{R} \right). \quad (5.34)$$

Two other ferrying schemes are proposed in [HB06], the chain-relay model where ferries travel only part way between nodes before off-loading buffers to another ferry, and the conveyor belt model where all ferries travel the entire distance between nodes; only transmitting to pick up and drop off data at sender or receiver. The schemes are depicted in Fig. 5.20(a,b) and their throughput and delays are given by

$$T^{\text{chain-relay}} = T^{\text{formation}}, \quad (5.35)$$

$$d_{\text{avg}}^{\text{chain-relay}} = \left(1 + \frac{1}{n} \right) \frac{L - r(n+1)}{v} + \left(n + \frac{1}{2} \right) \frac{b}{R}, \quad (5.36)$$

$$T^{\text{conveyor}} = \left[2 \left(\frac{L - 2r}{vnb} + \frac{1}{Rn} \right) \right]^{-1} \quad (5.37)$$

and

$$d_{\text{avg}}^{\text{conveyor}} = \left(1 + \frac{1}{n} \right) \frac{L - 2r}{v} + \left(\frac{1}{2} + \frac{1}{n} \right) \frac{b}{R} \quad (5.38)$$

respectively [HB06]. The chain-relay model can be considered transmission dominant while the conveyor model considered as travel dominant. The proposed formation approach is a hybrid of the two. Note that by solving

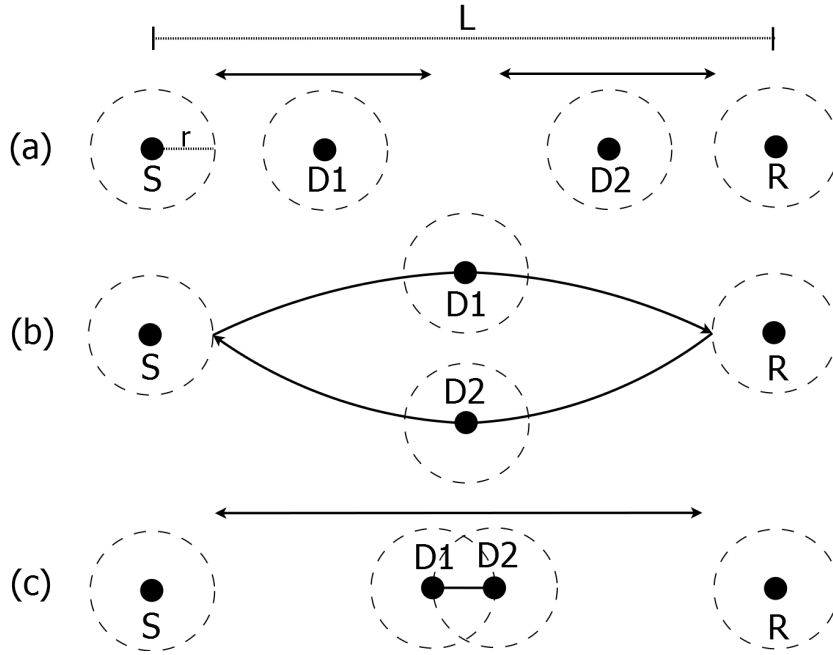


FIGURE 5.20: Three ferrying schemes with two ferries $D1$ and $D2$ ferrying between sender S and receiver R separated by distance L . Dashed lines are communications ranges of radius r , solid lines with arrows are movement tracks and without arrows indicate a communications link. (a) Chain relay, (b) Conveyor (2nd dimension exaggerated for clarity), (c) Flocking formation.

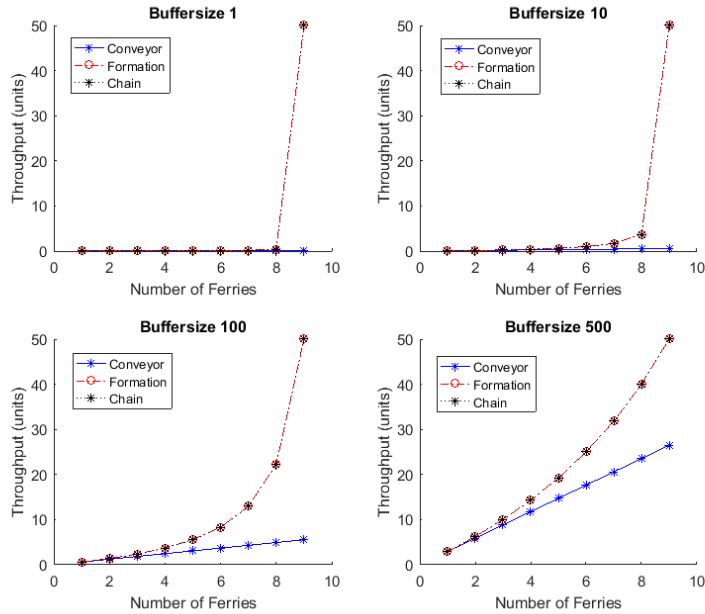


FIGURE 5.21: Throughput for the two resource-node ferrying problem with increasing buffer size.

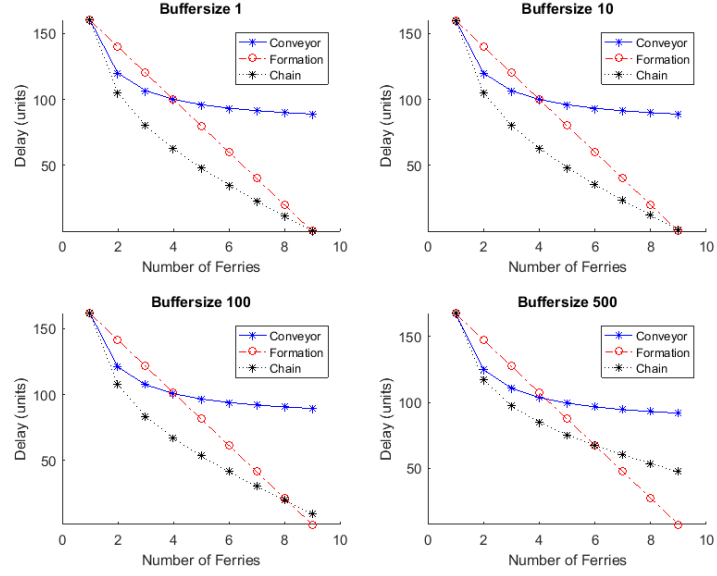


FIGURE 5.22: Delay for the two resource-node ferrying problem with increasing buffer size.

the equation $d_{\text{avg}}^{\text{chain-relay}} = d_{\text{avg}}^{\text{formation}}$ gives $n = (L - r)/(r - \frac{vb}{2R})$ where the two schemes are equivalent. When $R \rightarrow \infty$, $n = \frac{L-r}{r}$ and the schemes both represent a fully-connected multi-hop bridge between sender and receiver.

Figure 5.21 and 5.22 show the throughput and delay for the models with different buffer sizes when $R = 100$, $v = 1$, $L = 100$ and $r = 10$. As can be seen, model performance varies widely with different buffer sizes. The effect at play here is the ratio of R and v (here we assume fast radios). In the throughput and delay equations, the variables R and v control whether travel delay or transmission delay is the dominating factor. However, this analysis assumes all generated packets will fit in the buffer in one cycle. What we are interested in is the packet generation rates the schemes can support before the transmitter queue gets so large that it is never completely emptied. If the sender generates g packets per second then the total number of packets generated per cycle is $g \times t_{\text{cycle}}$. To fit all packets into the buffer then $b = g \times t_{\text{cycle}}$. Therefore $g = \frac{b}{t_{\text{cycle}}}$. Figure 5.23 shows maximum supported packet generation rates with changing buffer size. The flocking formation model outperforms the other two but its advantage quickly diminishes as buffer size increases. This is expected since the formation model collectively has a larger buffer size. But, as individual buffer size increases, the collective buffer size is not needed.²

²Intuitively, if you are going on a long journey, make sure you have enough luggage space to make it worthwhile.

When moving to multiple nodes in two dimensions, the flocking algorithm presented below can be considered the counterpart to the flocking formation model presented in this section. Here, the algorithm joins multiple ferries together along the direction of travel. As we shall see, the algorithm below creates a similar topology but in two dimensions.

5.4.2 Algorithm

Flocking refers to the emergent behaviour that can be observed when certain species of birds are in-flight. Individuals within the flock position themselves

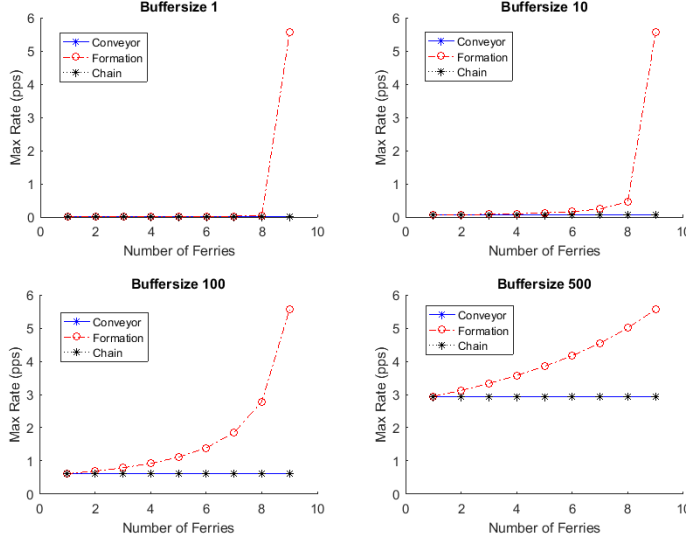


FIGURE 5.23: Threshold packet generation.

according to flockmates that they observe. This gives rise to patterns and structures that are properties of the collective. It was first modelled using the simple rules of separation, alignment and cohesion to generate realistic movement for computer animation [Rey87b]. A theoretical framework for a distributed flocking algorithm is given in [OS06] and has been used for applications such as surveillance [SB15b].

Using the notation of [OS06] and given the equations of motion

$$\dot{q}_i = p_i \quad (5.39)$$

$$\dot{p}_i = u_i \quad (5.40)$$

where $q_i, p_i \in \mathbb{R}^2$ and using adjacency matrix a_{ij} we have

$$a_{ij}(q) \neq 0 \quad (5.41)$$

if

$$(i, j) \in \mathcal{E} \quad (5.42)$$

where $i \neq j$ and $i, j \in \mathcal{R}$. Edges and routing were again determined using the fixed-range communications model and routing model from Section 3.1.2 and Section 3.1.3 respectively.

Each node i applies the control input $u_i = f_i^g + f_i^d + f_i^\gamma$ where f_i^g represents the gradient of the potential function induced by the attractive/repulsive force of flock-mates (a global minimum occurs when flock-mates are equally spaced) and f_i^d is a velocity consensus term and f_i^γ provides navigational feedback for a group objective.

The control function is composed of two elements $u_i = u_i^\alpha + u_i^\gamma$ where

$$u_i^\alpha = \underbrace{\sum_{j \in \mathcal{N}_i} \alpha(\|q_j - q_i\|_\sigma) n_{ij}}_{\text{Gradient term}} + \underbrace{\sum_{j \in \mathcal{N}_i} a_{ij}(a)(p_j - p_i)}_{\text{Consensus term}} \quad (5.43)$$

where $\mathcal{N}_i$ is the neighbour set of node i . The action function $\alpha(z)$ is defined [OS06] as

$$\alpha(z) = \rho_h(z/r_\alpha)(z - d_\alpha) \quad (5.44)$$

$$(z) = \frac{1}{2}[(a+b)\sigma_1(z+c) + (a-b)] \quad (5.45)$$

where in Eqn. 5.44, $r_\alpha = \|r\|_\sigma$ and $d_\alpha = \|d\|_\sigma$ are constants representing the communication range and the desired spacing of nodes, modified by the σ -norm

$$\|z\|_\alpha = \frac{1}{\epsilon} \left[\sqrt{1 + \epsilon \|z\|^2} - 1 \right] \quad (5.46)$$

where $\|\cdot\|$ is the Euclidean norm and $\epsilon \in (0, 1)$ [OS06]. Notice that $\|z\|_\alpha$ is differentiable everywhere. $\rho_h(z)$ is a *bump* function that smoothly varies between 0 and 1, and is used to construct a smooth potential function and adjacency matrix with finite cut-off. It is defined [OS06] as

$$\rho_h(z) = \begin{cases} 1, & \text{for } z \in [0, h) \\ \frac{1}{2} \left[1 + \cos \left(\pi \frac{(z-h)}{(1-h)} \right) \right], & \text{for } z \in [h, 1] \\ 0, & \text{otherwise.} \end{cases} \quad (5.47)$$

Equation 5.45 is a sigmoidal function with parameters a, b, c such that

$$0 < a \leq b, \quad c = |a - b| / \sqrt{4ab} \quad (5.48)$$

with $\sigma_1(z) = z / \sqrt{1 + z^2}$ [OS06]. In Eqn. 5.43, n_{ij} is a vector along the line connecting q_i and q_j (again modified by the σ -norm), that is,

$$n_{ij} = \frac{q_j - q_i}{\sqrt{1 + \epsilon \|q_j - q_i\|^2}}. \quad (5.49)$$

The adjacency matrix a_{ij} in the consensus term of Eqn. 5.43 selects which neighbours' motion to take into consideration based on their position and is defined [OS06] as

$$a_{ij}(q) = \frac{\rho_h(\|q_j - q_i\|_\alpha)}{r_\alpha} \in [0, 1]. \quad (5.50)$$

The navigational feedback part of the control input is defined as

$$f_i^\gamma(q_i, p_i, q_t, p_t) = -c_1(q_i - q_t) - c_2(p_i - p_t) \quad (5.51)$$

where c_1 and c_2 are constants balancing the importance of the group objective's current and future positions. In the proposed formation algorithm, the group objective is set to the current target to which the flock is ferrying data. For simplicity, user-nodes are static and so the target $(q_t, p_t) = (q_t, \mathbf{0})$, where $t \in \mathcal{U}$. Figure 5.24 shows the final topology for a flock using this control algorithm where $q_t = \mathbf{0}$ and the communication range $r = 20$ and the desired spacing of nodes $d = 20$. Moving in this way, the flock can transfer data between flockmates and fill neighbouring buffers while also having a large coverage area to opportunistically interact with user-nodes that are met during the migration to the current group objective.

Next-hop target selection is done in the same manner as before, i. e. Eqn. 5.1, but in the flocking case, this function must be evaluated using all packets

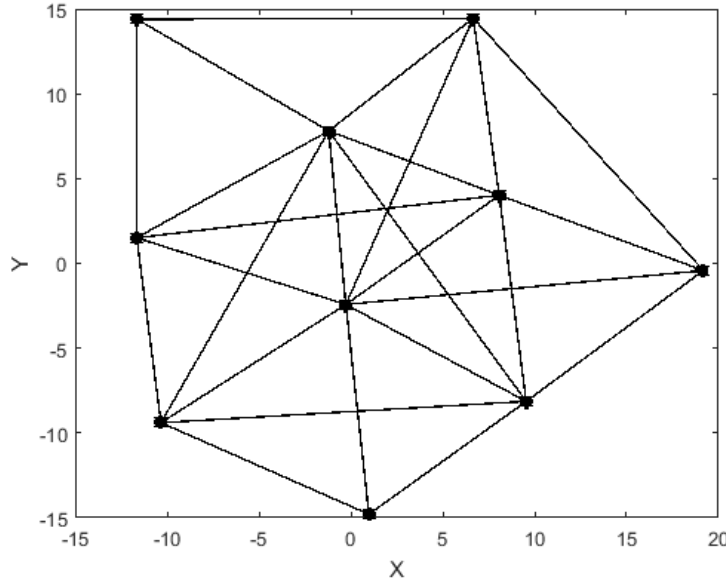


FIGURE 5.24: A flock of 10 nodes with the origin as the group objective. Lines represent valid communications links.

in the collective buffer. The next user-node to visit is then disseminated through the flock as a group objective. Global knowledge of the local flock is required to support this decision-making process. A centralised, flock leader approach is chosen in the following implementation where each node forwards its buffer information to the flockmate with the largest node identifier. To enable this, it is assumed that communication between flock-mates is fast and reliable; a reasonable assumption if $d < r$. In reality, nodes would be equipped with fast, short-range radios and **UAS** platforms would enable a line-of-sight free-space **RF**-propagation environment. This communication channel would also permit regular broadcast of ferry locations for input into the control algorithm.

When the flock comes into contact with any user-node, any flocking ferry containing data for the user-node is delivered. Multi-hop transport is used if required. Data is also picked up from the user-node until there is no data left or there is no more space within the flock. Again, multi-hop transport is used if required. After this occurs, the group objective is recalculated to take the new traffic into account. Algorithm 3 describes the operation of the data-ferry flocking.

5.4.3 Simulation

The formation flocking algorithm was compared against the control algorithm of individually-tasked ferries with no collective buffer (the ‘individual’ algorithm) for different ferry buffer sizes. The network was modelled as 10 user-nodes employing 5, 10 and 20 ferries. User-nodes would generate one packet per simulation step and uniform-randomly assign them to another user-node. Ten random topologies were generated where each algorithm was run for 5 replications of 10000 steps with a range of buffer sizes. Each simulation uniform-randomly generated user-node positions across a $600 \times$

Algorithm 3 Ferry Flocking Algorithm

```

1 procedure FERRY
2   while true do
3     if Ferry in contact with User then
4       if route exists to recipient then
5         sendTrafficFromFlock()
6       while flock buffer not full and user has packet do
7         storePacketInFlock()
8         sendBufferStatisticsToFlockLeader()
9       broadcastPosition()
10      move(getFlockMatePositions(), getGroupObjective())

```

600 unit field. A single random starting location was chosen for all ferries on the field. Initial ferry targets were also randomly chosen. Both algorithms employ the collective waiting time metric to decide the next hop. Both user-nodes and ferries have a communications range r of 50 units and ferries move at 1 unit-per-step. Packet latency was captured throughout the simulation. Note that a single buffer can be transferred in one time step and multi-hop transport incurs an extra 1 step-per-hop latency.

5.4.4 *Results*

Figure 5.25 depicts the average latency of the two algorithms for the 5 and 10 ferry case for a single topology. Interestingly, for very small buffer sizes, algorithm performance is similar. The flocking approach however has an advantage as buffer size increases but is still relatively small. This advantage continues until a crossover point is reached, at which point, the benefit of a large collective buffer with decreased distance between user-nodes (due to the formation coverage) over servicing multiple queues (user-nodes) simultaneously, is lost. This behaviour was also seen in the theoretical analysis. With a higher number of ferries, buffer sizes do not need to be as large before the crossover point is reached. In the 20 ferry case (not shown), the individual model outperforms the formation model immediately.

Figure 5.26 plots the ratio of the latencies obtained using the formation algorithm and the individual algorithm over all 10 topologies for the 5 and 10 ferry experiments. Again, the formation model favours smaller buffer sizes but loses its effectiveness as they increase. The point where this occurs depends on the number of ferries in the system, with more ferries requiring a smaller buffer. It is expected that this point is also dependent on the data generation rates of user-nodes, the speed of ferries and the size of the area of operations (user-node density) as these variables affect how much data can be generated during inter-ferry visits.

5.4.5 *Mathematical Model*

The simulation was also run with a single data ferry to test the theoretical buffer size that is required before latency improvement stops. In the case of one ferry, both models are equivalent. Intuitively, the ferry buffer would need to be quite large. To test this hypothesis, we first need the average

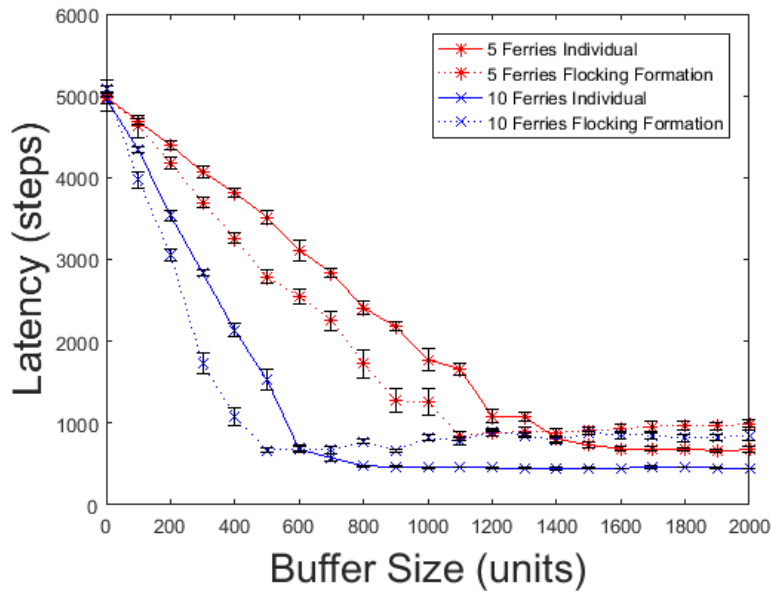


FIGURE 5.25: Average latency for a single topology. The individual and flocking formation results refer to the individual ferry algorithm and the flocking formation algorithm respectively. Error bars indicate one standard deviation.

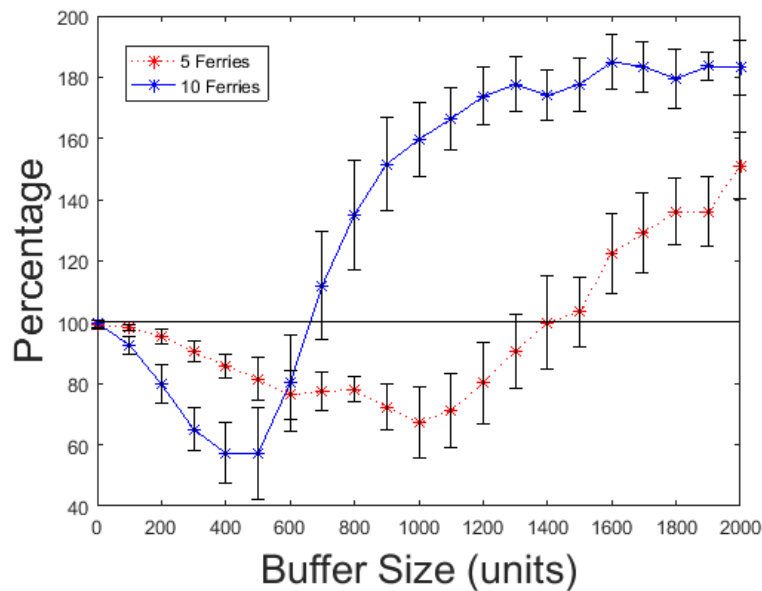


FIGURE 5.26: The mean latency ratio (formation on individual) for all topologies for different buffer sizes. Data points below the 100% threshold indicate superior performance of the formation algorithm. Error bars mark one standard deviation.

distance between nodes on the field. This can be obtained by solving the quadruple integral

$$\int_0^L \int_0^L \int_0^L \int_0^L \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} dx_1 dx_2 dy_1 dy_2 \quad (5.52)$$

where L is the side length of the field and the x and y variables represent Cartesian coordinates of the two points. For a unit square, the mean distance between two randomly generated points is $(2 + \sqrt{2} + 5 \ln(\sqrt{2} + 1))/15 \approx 0.5214$ [Wei04]. If a single ferry travels to user-node 1, then in that time each user has generated $0.5214 \times L - 2r = 0.5214 \times 600 - 100 = P$ packets. The ferry picks up these P packets where $P/9 = p$ are destined for each user-node. The ferry subsequently has equal probability of choosing user-nodes 2 to 10 but an ascending order is assumed for simplicity. Upon arriving at user-node 2, the ferry now picks up $2P$ packets and delivers p packets. Of the packets in the buffer at this point, $2p$ are for node 1, none are for node 2 (since it is at node 2 and node 2 doesn't address packets to itself) and $(p + 2p)$, consisting of the first p from node 1 and $2p$ from node 2, for the rest of the user-nodes. The total is therefore $2p + 8(p + 2p) = 26p$. Similarly, upon arriving and exchanging data with node 3, the ferry buffers $51p$ packets giving a sequence of $\{9, 26, 51, 84\}$. Solving this sequence gives

$$a_n = 4n^2 + 5n \quad (5.53)$$

and checking the 10th term for the 10th user-node yields $a_{10} = 450$. The ferry therefore buffers $450p = 450((0.5214 \times 600) - 100)/9 = 10,642$ packets at which point the sequence repeats. It is therefore estimated that the ferry requires a buffer of approximately 10,500 units in size before a latency plateau occurs.

Figure 5.27 shows the latency response of a single ferry with increasing buffer size for three tested topologies. While the data are noisy, a plateau can be seen to occur at approximately the 10,000 unit buffer size as predicted in the analysis above.

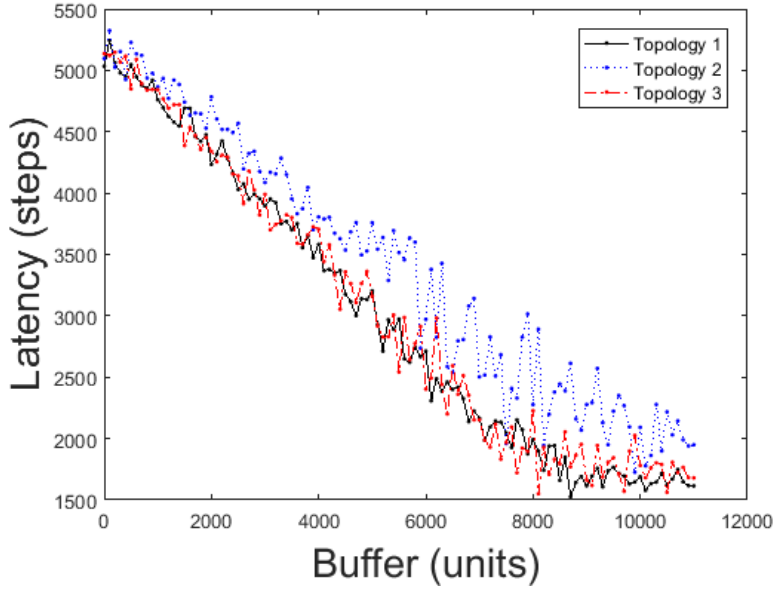


FIGURE 5.27: Mean latency as a function of buffer size for a single ferry for three tested topologies (error bars not shown).

5.5 SUMMARY

Due to the nature of **RF**-propagation, wireless communications can easily degrade or break; slow and fast fading, path-lost and (multi-path) interference all affect the strength and quality of the received signal. Many of these properties are dependent on the propagation environment and so it is unlikely that networks that span large areas will experience the same connectivity state. Other factors like local interference sources may also create localised degradation leading to varying connectivity states. Remediation approaches therefore require techniques to handle both highly-disconnected and semi-disconnected networks. As such, this chapter has introduced a swarm-intelligent approach to the coordination of a swarm of resource-nodes that can perform both data-ferry and data-relay functions dependent upon the underlying network connectivity state. **RGGs** were used to provide a framework for varying connectivity within experimentation. The algorithm was shown to outperform a traditional **TSP** tour solution of the user-node graph. A study of the resulting emergent behaviour of the resource-node swarm revealed localised behaviour where resource-nodes would operate more locally with increasing expected degree. This allowed resource-nodes to transition from data-ferrying around the entire map, to operating within graph component gaps, to finally performing stationary data-relaying; in some cases producing a fully-connected graph.

Under the assumptions of uniform traffic distribution and zero graph connectivity, we used observations of emergent structure within resource-nodes and statistical mechanics techniques to generate a data latency model that performs well as compared to simulation. This approach to building the model is useful in understanding how to analyze the structure of mass deployments of (**UAS**) swarms for coordinated tasks, where that coordination

is achieved implicitly through swarm intelligence and emergent behaviour. Phase transitions akin to those seen in connectivity properties of random graphs were observed in the probability of the system meeting a specific latency threshold. The probability rapidly moved from low to high at a critical point as the communications range of resource-nodes increased. A non-cooperative, non-emergent version of the algorithm did not display this behaviour.

Finally, as part of additional experimentation, the assumption of an infinite ferry buffer was removed and flocking emergent behaviour was investigated to make use of a collective ferry buffer and increased ferry footprint. It was shown through simulation that the algorithm can outperform the traditional non-flocking approach when the ratio of traffic to ferry storage is high. This may occur due to excess data generation at user-nodes, small ferry buffer sizes and/or too few ferries. The threshold point, in terms of individual ferry buffer size, where a collective buffer was superior to an individual buffer was quantified via simulation for various topologies.

As we have seen, the coordination of resource-nodes for the distribution of data to a disconnected user-node network is a difficult spatio-temporal problem. Nodes must cooperate in a meaningful way in order to maximise efficiency of the available resources even while being out of contact of other peer resources. The target platforms of this thesis are cheap (likely disposable) COTS devices; as such, platforms are likely to be generic, with swappable payloads that allow for different mission functions. The question then arises how coordination can be done not only in space and time, but across role or function as well? Chapter 6 investigates this problem and proposes an extension to the algorithm developed here to make it applicable to multi-role missions.

6

Extending Coordination Across Heterogeneously-capable Nodes for Multi-role Missions

- **SYNOPSIS** Communications is only one of many possible functions that **UASs** can perform. Depending on payload configuration, some **UASs** are multi-capable and can perform multiple roles within a single mission. A swarm of multi-capable platforms then has the additional problem of coordinating across roles as well as in space. The communications-oriented algorithm of the previous chapter is extended across multiple roles to handle this case. A modified application of a multi-criteria decision-making tool allows cooperation between multiple agents vying for platform control. The resulting architecture is shown to outperform dedicated teams of equivalent size in terms of mission latency time. Evidence of emergent specialisation is observed when the role-capability of resource-nodes require it; allowing the system to adapt for a better outcome. This chapter is based on the paper:

Bradley Fraser, Claudia Szabo, Andrew Coyle, and Robert Hunjet. “**Leveraging emergent specialization in a heterogeneous multi-role swarm control architecture for positional-based UAS missions**”. In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. Canberra, Australia: IEEE, Dec. 2020;

6.1 INTRODUCTION

In addition to the communications-based tasks introduced in the preceding chapters, **UASs** are capable of performing many other roles depending on their equipped payload. For example, **UAS** platforms have been extensively used for surveillance, target acquisition, reconnaissance [SB15b; Zhu+17], search and rescue [AYT18], and emergency logistics [GCT18], with significant potential for humanitarian assistance, disaster relief efforts and military use. Some **UASs** are capable of simultaneously performing more than one of these roles via multi-payload configurations [Elb19], in a *multi-role* capability. The autonomous control of swarming **UASs** is inherently difficult due to challenges of coordination, role allocation, and the engineering of emergent behaviour; the addition of platforms that are capable of multiple, possibly heterogeneous tasks, adds another dimension to the existing spatio-temporal problem and further complicates task scheduling and allocation.

Biological swarm control, on which the algorithm of Chapter 5 is developed, is resilient to individual losses and its distributed nature is inherently fault tolerant [ML16]. As such, there has been a large amount of research on using these techniques in swarm robotics [DL14]. Pheromone influencing, or stigmergy, found in many social insect species, supports the development of collective-wide emergent behaviour without the need for persistent intra-swarm communications [DBT00]. Moreover, cooperative behaviour between homogeneous and heterogeneous individuals of these species is well-documented [GGT07]. These characteristics make the approach suitable to the multi-role problem to address the following autonomous control questions:

1. How can movement-based UAS roles be coordinated between peer UASs?
2. What role should a particular UAS be performing at a particular point in time?
3. How can a commander influence the behavior of the collective?

Building on work of the previous chapter, the communications-oriented swarm control algorithm is extended to handle multi-capable resource-nodes for multi-role missions. The multi-agent decision architecture that is developed allows for any number of dynamic and heterogeneous agents whose individual mission can be abstracted to moving to a particular geographic position in an AO.

6.1.1 Algorithm

The common requirement for many UAS roles is the ability for the node to move to within range of a given target-node for a given piece of hardware specific to that role. For example, moving to within radio range of a disconnected communications target-node, moving to within drop range of a logistics target-node and moving to within camera range of a surveillance target-node. Thus, for these communication, logistics and surveillance roles we abstract the problem to deciding which target-node, of which role, a UAS should service at a given time. Node control takes the form of providing trajectory solutions that best meet the (conflicting) goals of different roles. The main objective of the proposed architecture is the management of mission assets in a perpetual mission where all targets are continually visited. A small time between visits facilitates the upload and download of data packets for communications, better situational awareness of surveillance targets and a smaller time between resupply of logistic targets.

The problem is stated as follows: at each time-step t , each resource-node in $\mathcal{R}$ chooses a single target-node from the set $\mathcal{U} = \mathcal{C} \cup \mathcal{S} \cup \mathcal{L} = \{x : x \in \mathcal{C} \text{ or } x \in \mathcal{S} \text{ or } x \in \mathcal{L}\}$ where $\mathcal{C}$ is a set of communications target-nodes, $\mathcal{S}$ is a set of surveillance target-nodes, and $\mathcal{L}$ is a set of logistics target-nodes, such that the service time of all target-nodes is minimized and adheres to commander role priorities. Similarly to previous chapters, service time is defined as the time between resource-node visits to target-nodes and is used as the primary performance metric.

The multi-role swarm coordination problem is a two-level **Multi-Agent System (MAS)**. The first level system is internal to the node where agents representing the interests of each role cooperate for control of the node. The second level system considers each node to be an individual agent where nodes must also coordinate to achieve a common mission.

In the following description, the term resource-node is used to denote a platform-level entity, while agent is used to denote a software agent internal to a resource-node. For example, a resource-node may contain a *communications* agent, and/or a *surveillance* agent and/or a *logistics* agent. This approach generalises to other not yet known tasks.

6.1.2 Node-Level Multi-Agent System

In the node-level **MAS**, each resource-node contains a set of m agents $\mathcal{A} = \{a_1, \dots, a_m\}$ representing the m roles of which it is capable. Each agent contains a list of target-node locations that are known in advance and must select one of these target-node locations as a position suggestion for the resource-node to move towards. Given the resource-node can only move to a single location, the position suggestions from all agents must be de-conflicted in such a way that provides the maximum benefit. This is achieved via the scoring of possible positions and multi-criteria decision-making.

Chapter 5 developed a swarming-inspired control algorithm to transfer data around a disconnected network. There, network traffic itself was used as a pheromone to determine resource-node movement behaviour. The resource-node would inspect its packet buffer and, using a value function based on packet waiting times, would determine the value of visiting all target-nodes for which it had traffic. The target-node with the maximum value would be set as the next-hop destination. Traffic flows then influenced individual and collective node movement: in the former by being the basis on which the next-hop location was selected, and in the latter by being transferred to other nodes by geographic routing.

In the multi-role problem, surveillance target-node and logistics target-node do not generate data traffic that can be picked up and transferred between resource-nodes. Instead, a digital pheromone based on distance to the target-node and time since it was last visited is continually generated within the resource-nodes for each target-node. To ensure pheromones of different roles can be scored and compared equally, the same process is used to generate the pheromone for the communications role as well. This comes at the expense of treating all communications nodes equally with respect to traffic, rather than being reactive to it.

The value v_j for visiting target-node j is defined as:

$$v_j = \max(0, t_j - \|\vec{x}_j - \vec{x}_i\|), \quad (6.1)$$

where t_j is the time since resource-node i has visited target-node j , $\vec{x}_i$ is a two-dimensional position vector and $\|\cdot\|$ is the Euclidean norm. Subtracting the target-node distance produces a behaviour such that local target-nodes are preferred. Value functions that inspect traffic buffers are a common approach to next-hop decision-making in the data ferrying literature [**SMT13**; **PLX12b**].

TABLE 6.1: The TOPSIS decision matrix. Diagonal entries represent an agent scoring its own suggestion.

Options $\mathcal{O}$ (agent suggestion)	Criteria $\mathcal{C}$ (agent score)		
	$s_1(\cdot)$	$s_{\dots}(\cdot)$	$s_m(\cdot)$
o_1	$s_1(o_1)$	$s_{\dots}(o_1)$	$s_m(o_1)$
$o_{\dots}$	$s_1(o_{\dots})$	$s_{\dots}(o_{\dots})$	$s_m(o_{\dots})$
o_m	$s_1(o_m)$	$s_{\dots}(o_m)$	$s_m(o_m)$

Waiting times for each target-node are incremented each simulation step and reset to zero whenever the resource-node moves to within range of the target-node (or possibly when in contact with other resource-node). The position of the target-node with the largest value is returned as the agent's preferred next-hop position suggestion.

Multi-criteria Decision Making

During each control loop, m locations are produced, each representing the best location to visit from the perspective of each agent. One option would be to simply treat all roles equally and take the maximum of them; equivalent to having a single agent that does not distinguish between roles at all. However, this does not provide a mechanism by which a human operator can influence role priorities. To add this functionality, a modified version of the multi-criteria decision-making tool **TOPSIS** [TH11] was used.

TOPSIS compares a set of options based on a set of criteria. For example, style, cost and fuel consumption may be a set of criteria for a set of vehicle options one might want to purchase. **TOPSIS** is a compensatory algorithm and so allows trade-offs between criteria. Additionally, the algorithm accepts a set of weights for the evaluation criteria as input and so provides a convenient way to influence the priority of different roles. Mission intent can be translated into these weights at the beginning of the mission, but they can also be updated online via weight update messages that are flooded through the resource-node network.

In our implementation, the options $\mathcal{O} = \{o_1, \dots, o_m\}$ are the agent position suggestions, and the criteria $\mathcal{C} = \{c_1, \dots, c_m\}$ are the 'goodness' of each suggestion as evaluated by the scoring function of each agent. The **TOPSIS** decision matrix is shown in Tab. 6.1.

Agents score each others' suggestion using the same value function that was used to generate their own initial suggestion. However, the position suggestion of another agent may not correspond to one of an agent's own target-nodes. That is, the position suggested by agent A, may not match a target-node of agent B. The probability of this occurring in part depends on how many target-nodes are common to both agents A and B. In the simulations below, we consider the worst case where the target-node sets are disjoint. In this case, agents would always return a score of zero for their peers' suggestion, making the matrix in Tab. 6.1 diagonal, with diagonal entries representing an agent scoring its own suggestion. After **TOPSIS** normalization, the matrix becomes the identity and the outcome is solely

determined by the user-defined weight inputs (with ties broken randomly in the case of equal weights). This does not facilitate multi-agent interaction. To enable this interaction and allow agents to influence each another, the scoring function returns the sum of the values of all target-nodes that would be visited (that is, within range) if the node were placed in the suggested position. That is, agent i 's calculated value v_k in visiting agent k 's suggestion $\vec{x}_k$ is

$$v_k = \sum_{j \in \mathcal{T}} \max(0, t_j - \|\vec{x}_j - \vec{x}_k\|), \quad (6.2)$$

where the target-node set $\mathcal{T} = \{x_l\} \forall l \in \mathcal{C} \text{ or } \mathcal{S} \text{ or } \mathcal{L} : \|\vec{x}_l - \vec{x}_k\| < r$ where r is the role-dependent range.

The modified scoring function differs from the typical **TOPSIS** process and facilitates a form of interaction between the multiple agents. While a peer agent's suggestion may not be optimal for the current agent, it may provide some value which allows the **TOPSIS** process to perform trade-offs. Interestingly, a peer agent's suggestion may yield a higher visit value than an agent can produce itself. This could occur in the case where targets are clustered together since initial agent suggestions are generated by examining the locations of single target-nodes, not arbitrary locations in the **AO**. Peer suggestions can be considered arbitrary from another agent's point-of-view as they do not match any of their own targets. This interaction enhances an agent's ability to explore the possible space of positions.

An alternative brute-force approach would be to use every (discretized) position in the **AO** as position suggestions and have all agents score them. Not only is this computationally inefficient given that **TOPSIS** is $\mathcal{O}(m^2)$ complex [HW16], but many suggestions will be generated that are not applicable to any of the agents at all.

While not implemented, we note that a possible improvement to the scoring function would be to not just score the target-nodes in range at the suggested location, but additionally score all target-nodes on the way to the suggested location as well. We also note that, although not considered here, the scoring approach captures the case of intersecting target-node sets, such as a soldier being the target-node for a communications service and a logistics drop.

Figure 6.1 depicts the scoring process between agents in the resource-node (**UAS**). The communications agent has three possible targets to visit: c_1 , c_2 and c_3 . From Eqn. (6.1), it has determined the closest target-node c_1 provides that agent with the greatest value. Meanwhile, the surveillance agent has suggested the position s_1 . While this position does not correspond to any of the communications target-nodes, the communications agent considers which of its target-nodes would be in range (denoted by the dotted circle) if it were in that position. In this case, the position suggested by the surveillance agent would allow the communications agent to service target-nodes c_2 and c_3 (but not its preferred one c_1). The value to the communications agent in doing this is calculated using Eqn. (6.2). The values of Eqn. (6.1) and Eqn. (6.2) from both agents define the decision matrix in Tab. 6.1. Figure 6.2 shows the node-level multi-role decision process.

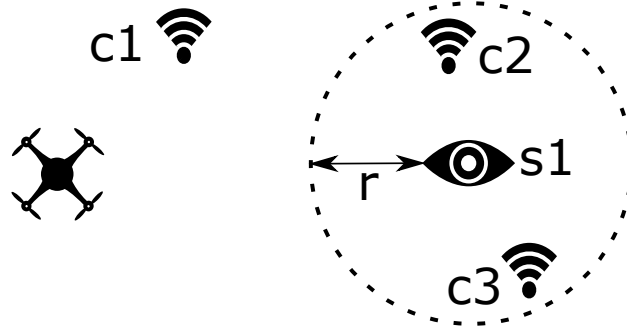


FIGURE 6.1: The communications agent scoring the surveillance agent's position suggestion. The communications agent prefers its closest target-node c_1 but considers which of its target-nodes would be serviced (i.e. target-nodes c_2 and c_3) if the surveillance target-node s_1 was selected.

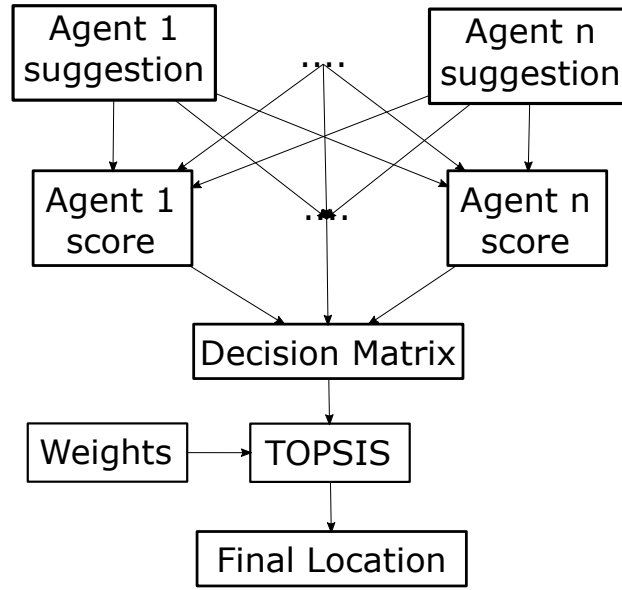


FIGURE 6.2: The multi-role decision architecture. Agents produce a location suggestion which is then scored by all other agents producing the **TOPSIS** decision matrix. Using the input weights, the **TOPSIS** process produces the location to visit next.

To facilitate coordination between resource-nodes, we use Alg. 2 developed in Chapter 5. It is assumed that a delay-tolerant networking protocol [Fal03] with geographic routing is running on the communications-capable resource-nodes to facilitate actual data transfer for the communications role. This implies that surveillance-capable resource-nodes are equipped with the required sensors and logic to conduct the surveillance task. A similar statement is assumed for the logistics role.

6.2 EXPERIMENTAL ANALYSIS

Each set of target-nodes are uniform-randomly placed on a 100×100 unit field and $|C| = |S| = |\mathcal{L}| = 10$. For simplicity, the communications range r_c , surveillance range r_s and logistics range r_l use a fixed disc model (see Section 3.1.2) with equal radii $r_c = r_s = r_l = r$. Whenever resource-nodes come into range of a target-node they are able to act on, the corresponding action is assumed to take place immediately and without error. That is, data

is transferred and/or surveillance is acquired and/or supplies are dropped.

A Python environment utilising particle-model movement was used for simulations with the NetworkX [HSS19] library performing graph computations for data routing (see Section 3.1.3). Simple movement models were used for the resource-nodes while all target-node locations are static and are known in advance by resource-nodes. Results are an aggregate of 10 trials of 2000 discrete time steps with 95% CI shown under the assumption of normally-distributed errors. The single-role approach where each role is performed by a dedicated team was used as a baseline for comparison. The self-organizing behaviour is identical to the multi-role case but nodes can only perform a single role. In the following descriptions, n refers to the total number of resource-nodes and the notation $(\cdot, \cdot, \cdot)$ refers to the number of resource-nodes in a communications, surveillance and logistic team respectively.

The time between visits to target-nodes is used as the performance metric and is measured in discrete time-steps. It is desirable for this value to be as small as possible as it corresponds to faster data transfer (communications), better situational awareness (surveillance) and better supplied units (logistics). Given that the swarm is to support the mission in perpetuity, the mission is never considered to be over and practical considerations such as flight time are ignored. The experiments below examine the architecture under conditions of interest: equal role priorities, unequal role priorities, tolerance to node faults, dynamically required roles and heterogeneously-capable nodes.

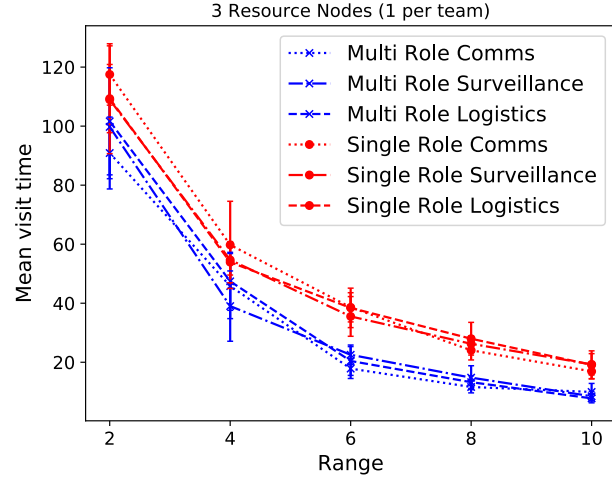
6.2.1 Results

Equal Role Priorities

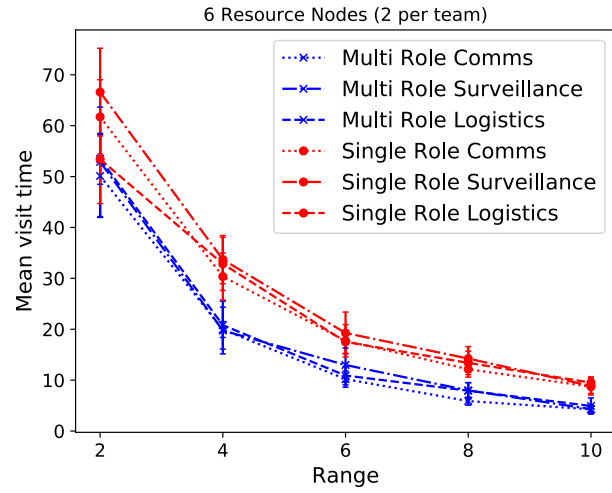
The first experiment compares the performance between dedicated teams and a single, equivalent-sized multi-role team when role priorities are equal. Figure 6.3 plots the mean visit time for $n = 3, 6$ and $r = 2, 4, 6, 8, 10$. As resource-node numbers and their range increase, the opportunity to hand-off pheromones between themselves also increases and they begin to self-organize. Even though self-organization occurs in both approaches, the larger the team, the more benefit the self-organization gives and so the multi-role team outperforms the dedicated ones, which as an aggregate is larger and outperforms the smaller dedicated teams as evidenced by the lower mean visit times

Unequal Role Priorities

The second experiments tests the architecture's ability to coordinate nodes in accordance with specific role priorities. The communications role was assigned as being three times more important than the other two roles. For the dedicated teaming approach, $n = 5, 10, 15$ resource-nodes were tested where the teams were divided into $(3, 1, 1)$, $(6, 2, 2)$ and $(9, 3, 3)$ respectively for communications, surveillance and logistics roles. The multi-role team had $n = 5, 10, 15$ nodes also but used the TOPSIS weights $(0.6, 0.2, 0.2)$ where



(a) Three resource nodes (1 per team).



(b) Six resource nodes (2 per team).

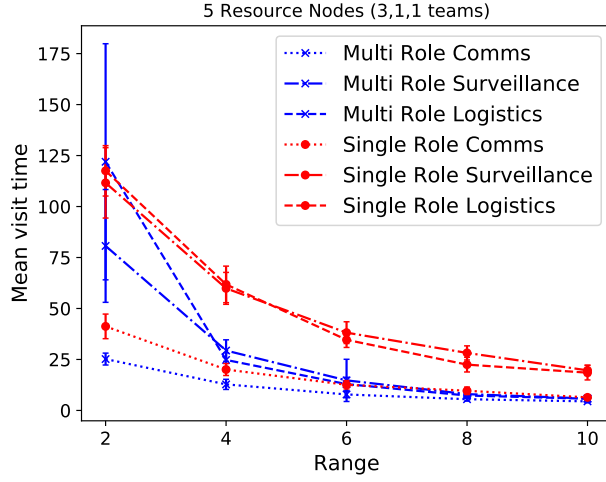
FIGURE 6.3: Mean visit times when role priorities are equal.

$\sum_{i=1}^m w_i = 1$ with w_i being the weight for role i . Note that $n = 5, 10, 15$ was chosen to easily divide nodes into three teams.

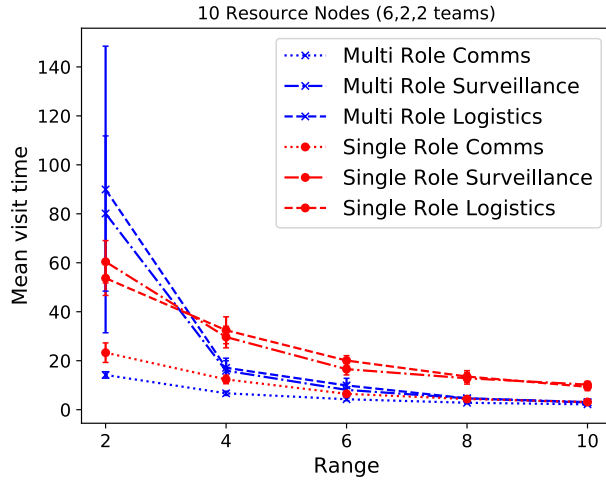
Figure 6.4 plots the mean visit times for the unequal priority simulations. As expected, the communications role has the lowest visit times in all cases for both approaches. However, this comes at a large cost to the other roles in the dedicated team approach. In the multi-role approach, the communications task is favoured but visit times for the other two roles are much lower than the dedicated teams; in some cases, just as low as dedicated team's priority task.

Fault Tolerance

Another important consideration for the architecture is its response to failed resource-nodes. Intuitively, failed nodes in the smaller dedicated teams will have a much larger impact on performance when compared to the multi-role case since the multi-role architecture allows for all other resource-nodes



(a) Five resource nodes (3,1,1 teams.)



(b) 10 resource nodes (6,2,2 teams).

FIGURE 6.4: Mean visit times when role priorities are unequal. The communications role is twice as important as the surveillance and logistics roles.

to cover for any loss of other resource-nodes. This can be seen in Fig. 6.5 where 12 resource-nodes represent a full team; $n = 12$ and an equal (4, 4, 4) split respectively for the multi and single-role approaches. Simulations were then ran for $n = 9, 10, 11$ resource-nodes representing 1, 2 and 3 failed resource-nodes. For the single-role case, the worst case is shown where all the failed resource-nodes come from the same specialized (communications) team which shows a significant decrease in performance. The multi-role architecture however shows only a slight decrease in performance despite losing 25% of the team.

Dynamics

There may be situations where the requirement for particular roles are transitory. For example, a wireless network that is connected does not require a data ferrying service. However, the network may be intentionally jammed at a later time. Based on the described scenario, we examine how the

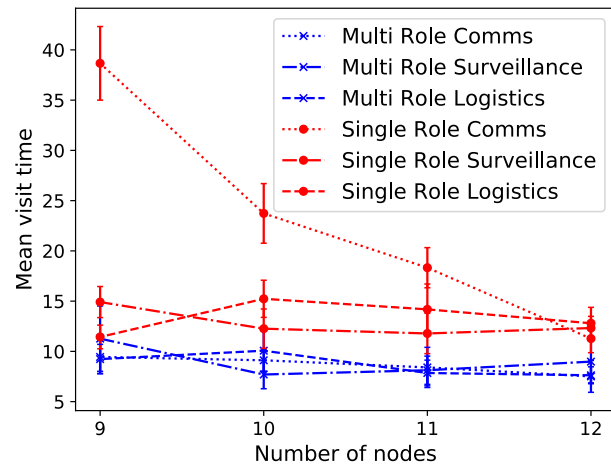


FIGURE 6.5: Impact on performance of failed nodes.

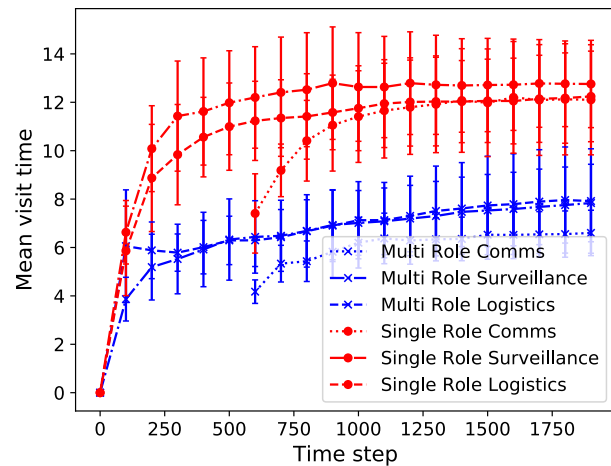


FIGURE 6.6: Both the multi-role and single-role approaches quickly react to the dynamically introduced role requirement at time step 500.

architecture reacts to a dynamically introduced role such as in this scenario. Initially, the communications role is turned off and only turned on at time step 500. Figure 6.6 shows the mean visit time response compared to the single-role case, and the visit time difference between the communications role and the mean of the other two roles as the simulation progresses. In both approaches, the new requirement is quickly absorbed into the system operation, with the mean quickly approaching that of the other two roles (due to the equal priorities). There is no clear winner between the approaches for this property however, it is noted that the dedicated team would need to be launched once the communications role was required, whereas the multi-role team would already be on-station performing other duties.

Heterogeneous Roles

Chapter 5 shows that resource-nodes are able to self-organize into a structure that facilitates data transfer, with resource-nodes tending to stay within a local operating area with the size of that area dependent on the amount of interaction between them. We perform heterogeneity experiments to test the

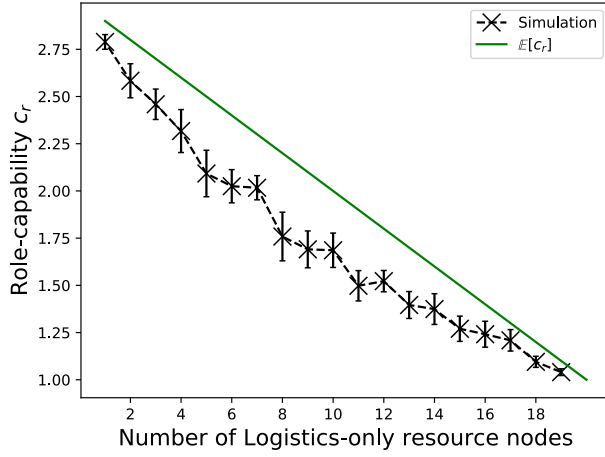


FIGURE 6.7: Mean number of roles a resource-nodes is capable of (c_r) when visiting a logistics target-node versus the number of logistics-only resource-nodes.

hypothesis that this behaviour will also be generated in the role dimension of the coordination problem. Twenty multi-role resource-nodes were split into two groups: a *multi-role group* with all three role agents and a *logistics-only group* with only a logistics agent. Role priorities were set equally and assumed that logistics agents had infinite resources and continually serviced logistics target-nodes. To discern which resource-nodes were performing which function, the number of agents on-board of resource-nodes (i. e. the *role-capability* c_r) were recorded when they came into contact with a logistics target-node; results are shown in Fig. 6.7.

When the number of logistics-only nodes is small, the servicing resource-nodes on average contained close to 3 agents. Though not plotted, zero logistics-only nodes would yield a maximum role-capability of 3. Likewise, when the number of logistics-only nodes is large, node-capability is close to 1. This result is unsurprising since it reflects the proportion of the type of nodes in the system. The simulated results show an approximated downward trend of role-capability with an increasing number of logisitcs-only nodes. However, before we can answer questions about ‘role locality’ we must examine the expected role-capability. This is performed in the next section.

6.3 MATHEMATICAL MODEL

The expected role-capability based on the proportion of logistics-only nodes can be calculated as follows. If x is the number of logistics-only resource-nodes, and n the total number of resource-nodes, then based on proportions, expected role-capability is given by

$$\begin{aligned} \mathbb{E}[c_r] &= a \left(\frac{x}{n} \right) + b \left(\frac{n-x}{n} \right) \\ &= \left(\frac{x}{n} \right) (a - b) + b \end{aligned} \quad (6.3)$$

where a and b are constants representing the number of roles. In the case above,

$$\mathbb{E}[c_r] = 3 - 2 \left(\frac{x}{20} \right) \quad (6.4)$$

Equation 6.4 is shown in Fig. 6.7 as the linear green line.

As noted, the extremal cases are the points where the simulated and proportion-based calculation match most closely. However, the discrepancy in the rest of the cases suggests that there is some other process at play rather than just pure proportions. For example, when proportions are equal (10 resource-nodes each), visiting resource-nodes' role capability is less than the expected value of 2. In order for this result to occur, the 10 three-role-capable resource-nodes must preferentially select the communications and surveillance tasks, leaving the logistics-only resource-nodes to the logistics task. This suggests that self-organisation does indeed occur across the role dimension of the coordination as well as the spatial one.

6.3.1 A Method for Team Size Calculation

Using the mathematical model of the algorithm in [Chapter 5](#), it is possible to derive the marginal benefit of adding additional resource-nodes to the problem. Given this known relationship, and under the assumption that it holds for other similar roles, team numbers can theoretically be optimised using constraint optimisation via the process of Lagrange Multipliers. For example, assuming we have D UASs and the proportion that do role i is P_i with the constraint $\sum_i P_i = 1$, and the assumed utility function of the entire team consisting of exponential marginal benefits is

$$f(P_1 \dots P_m) = 1 - \sum_i^m \exp^{-\alpha_i P_i D} \quad (6.5)$$

where α_i is some value related to importances of roles, then when $m = 2$ for example,

$$f(P_1, P_2) = 1 - \exp^{-\alpha_1 P_1 D} - \exp^{-\alpha_2 P_2 D}. \quad (6.6)$$

Letting

$$g(P_1, P_2) = P_1 + P_2 - 1 \quad (6.7)$$

and using the method of Lagrange Multipliers,

$$\begin{aligned} \mathcal{L}(P_1, P_2, \lambda) &= f(P_1, P_2) + \lambda g(P_1, P_2) \\ &= 1 - \exp^{-\alpha_1 P_1 D} - \exp^{-\alpha_2 P_2 D} + \lambda(P_1 + P_2 - 1). \end{aligned} \quad (6.8)$$

The gradient is given by

$$\begin{aligned} \nabla_{P_1, P_2, \lambda} \mathcal{L}(P_1, P_2, \lambda) &= \left(\frac{\partial \mathcal{L}}{\partial P_1}, \frac{\partial \mathcal{L}}{\partial P_2}, \frac{\partial \mathcal{L}}{\partial \lambda} \right) \\ &= (D\alpha_1 e^{-D\alpha_1 P_1} + \lambda, D\alpha_2 e^{-D\alpha_2 P_2} + \lambda, P_1 + P_2 - 1) \end{aligned}$$

and $\nabla_{P_1, P_2, \lambda} \mathcal{L}(P_1, P_2, \lambda) = 0$ yields

$$-D\alpha_1 e^{-D\alpha_1 P_1} + \lambda = 0 \quad (6.9)$$

$$-D\alpha_2 e^{-D\alpha_2 P_2} + \lambda = 0 \quad (6.10)$$

$$P_1 + P_2 - 1 = 0. \quad (6.11)$$

Hence,

$$\begin{aligned}
 \alpha_1 e^{-D\alpha_1 P_1} &= \alpha_2 e^{-D\alpha_2 P_2} \\
 -D\alpha_1 P_1 + D\alpha_2 P_2 &= \ln\left(\frac{\alpha_2}{\alpha_1}\right) \\
 \alpha_1 P_1 - \alpha_2(1 - P_1) &= -\frac{\ln\left(\frac{\alpha_2}{\alpha_1}\right)}{D} \\
 P_1(\alpha_1 + \alpha_2) - \alpha_2 &= -\frac{\ln\left(\frac{\alpha_2}{\alpha_1}\right)}{D} \\
 P_1 &= \frac{D\alpha_2 - \ln\left(\frac{\alpha_2}{\alpha_1}\right)}{D(\alpha_1 + \alpha_2)} \quad (6.12)
 \end{aligned}$$

$$P_2 = \frac{D\alpha_1 + \ln\left(\frac{\alpha_2}{\alpha_1}\right)}{D(\alpha_1 + \alpha_2)} \quad (6.13)$$

So if importances are equal for example, i.e. $\alpha_1 = \alpha_2$, then $p_1 = p_2 = 0.5$ and $p_1 + p_2 = 1$. When considering multi-capable nodes, an interesting observation is that, in some sense, $\sum_i P_i > 1$ within this framework and may provide an avenue for analytic comparison as part of future work.

This approach was not pursued due to the considerable effort required to model a single role, which would need to be replicated for each new role. However, the approach could be readily employed for example where there are multiple user-node networks, each with a team of resource-nodes following the algorithm from [Chapter 5](#). The finite number of resource-nodes could then be apportioned with this approach.

6.3.2 Discussion

Specialization can often perform better than policies that force homogeneity [[LMAM02](#)]. In our case, the multi-role capable resource-nodes specialize in the communications and surveillance tasks appropriate to the available logistics resources. This is analogous to certain biological systems. For example, honey bees dynamically adapt foraging behaviour for pollen, nectar and water depending on colony demand [[CP88](#)]. Unlike non-emergent specialization such as in morphological caste systems found in termites, emergent specialization is not pre-defined and allows the system to vary the degree of specialization to optimally solve the problem [[NSE08](#)]. In [[Fra+20a](#)] we have developed a mathematical model for the latency data packets experience when being transferred by the single-role algorithm in [[Fra+19](#)]. Using constraint optimization and role importances, it is possible to optimally assign the number of resources to each team. This would be an approach to non-emergent specialization but is not adaptive to changing requirements or node losses.

Results were gathered under the assumption that role target-nodes were uniformly distributed across the field. Further experimentation is required for non-uniform placement, particularly the situation where all role targets of a particular type are clustered together some distance away from other role clusters. In this case, the multi-role architecture will likely form a data (pheromone) ferrying bridge between the clusters of homogeneous type. This behaviour would likely underperform compared to that of dedicated teams.

Another variable that may significantly change results are the role-dependent ranges. In this work, they were assumed to be equal but this is almost surely not practical in real-world situations. Related to the equality of role ranges is the fact that experiments were conducted under the assumption of a 2D environment with altitude being ignored. Some roles will require nodes to be closer to their respective targets than others. Verticality obviously contributes to this distance as well, however we postulate that the architecture is still valid in 3D environments provided that nodes of different altitudes can still communicate.

6.4 SUMMARY

This chapter introduced an extensible swarm control architecture that produces self-organizing behaviour in space and across roles. Analysis shows that this multi-role architecture outperforms equivalent teams of nodes that were dedicated to a specific role only. Additionally, the heterogeneous role experiments above demonstrate evidence of specialization, where individuals perform only particular task(s) in order to increase collective performance.

Analysis suggests that swarm platforms that are capable of multiple functions within a single mission are more efficient than the specific single-role alternative in the case of uniform-random node targets. This is appealing for many military and humanitarian crisis scenarios, such as, for example, when surveillance data must be acquired and transferred back to a headquarters in the absence of long-range data links. A more complex scenario could see an initially loaded logistics platform progressively expend its payload but still be used in other capacities.

- **FROM ADAPTABILITY TO SCALABILITY** Having presented several efficacious algorithms for the coordination of multiple nodes in difficult communications environments for various **UAS** applications, as well as investigations into their adaptability, our attention is now turned to addressing swarm scalability issues. Specifically, we consider scalability from two perspectives. We first look at ensuring scalable design when employing **MADRL** architectures to generate swarm behaviour. Secondly, we look at how deployment of many agents may affect the communications environment and in turn how this affects the generation of emergent behaviour.

7

Towards Scalable Policy Transplantation

► SYNOPSIS

Heuristic design of large multi-agent algorithms, particularly those that leverage emergent properties, is an inherently difficult task. The large number of agents and their numerous interactions rapidly increase problem complexity. An alternative to expert algorithm design is the optimization of swarm behaviour through iterative simulation. One such approach that has garnered recent attention for state-of-the-art performance in cooperative games is that of **MADRL**. **MADRL** utilizes the power of **DNNs** to explore the large state-spaces associated with **MASs**. However, the use of **DNNs** come with significant compute requirements and do not always scale well with an increasing number of swarm agents. To address this issue, this chapter moves away from expertly-designed systems to learned control policies and investigates policy transplantation; the process of transplanting a single learned policy from a small set of trained agents, into a larger swarm. This scalable ‘train small, deploy big’ paradigm avoids training large swarms, removes the need for retraining when the number of swarm members change, and therefore provides the flexibility to alter the deployed swarm size on-the-fly as mission parameters dictate. However, policy transplantation is not always successful, even with homogeneous swarms. Rather, it is conjectured in this thesis that in order for policy transplantation to be successful, the learned policy must be indistinct to the agent on which it is trained. This chapter is based on the paper:

Bradley Fraser and Giuseppe Laurito. “Train Small, Deploy Big: Do Relative World Views Permit Swarm-Safety During Policy Transplantation for Multi-Agent Reinforcement Learning Problems?” In: *AI 2020: Advances in Artificial Intelligence*. Springer International Publishing, 2020, pp. 269–280.

The reader is directed to **Appendix A** for signed Statements-of-Authorship for the publication listed above.

7.1 INTRODUCTION

Multi-agent problem complexity rapidly increases with the number of agents and their individual behavioural intelligence [**PL05**], making the heuristic design of such algorithms a difficult task. Both *teams* of exquisite agents as

well as *swarms* of lesser-capable agents suffer from this algorithm design problem. In the latter, algorithms often leverage swarm intelligent algorithms [BDT99] that are known to generate useful emergent behaviour. For example, there are many algorithms that make use of Reynolds' boids [Rey87a] or similar rules for the cooperative autonomous deployment of communications and sensor nodes in MANETs and WSNs [KBJ13a; Hun15a; SB15a; Yua+18; Fra18b; Fra18a].

However, heuristic design of swarm intelligent algorithms by domain experts is also a difficult task as the intelligence within these algorithms is an emergent global property. One interpretation of such emergent behaviour, known as Weak Emergence, specifies that any new properties arising in the system are a result of the interactions between individuals, but the property itself cannot be explained using reductionist methods [Bed08]. Investigations must rather rely on post-mortem analysis or simulation (such as is used in the former chapters of this thesis). This then leaves algorithm design in a rather difficult state of iterative trial-and-error. To speed up this iterative design, the trial-and-error process can be wrapped in an optimization routine where variables that govern individual agent actions are optimized against the fitness of the global property that is produced. Thus the significant developments in deep reinforcement learning techniques [Sil+16; Mni+15; Mni+13] have been applied to the multi-agent problem space [Tam+15; NNN18] and specifically to swarm systems [HŠN19] and emergence [Bak+19] in recent years. These new multi-agent learning architectures have allowed the learning of cooperative solutions to complex multi-agent problems such as collision avoidance [Lon+17], surveillance [Liu+20; RK18], exploration [Gen+18], autonomous driving [Răd+19; BGSC20] and the monitoring of wildfires [JK18a; HS18] and flooding [BPZ19]. In such approaches, a control policy is learned for each agent in the swarm with various techniques designed to overcome the partial observation and non-stationarity problems associated with training multiple agents [HŠN19; Che19; Low+17; GEK17].

Optimization-based methods however are typically time intensive; the quintessential example being the evolutionary timescales required to produce the flocking, ant foraging and termite mound building of today's insects; generated through evolution and natural selection [DBT00]. From an artificial evolution perspective, the use of DNNs also comes with costs, a significant one being time-complexity. For example, the equivalent size of a Linear Program (LP) representation of a simple, fully-connected DNN, as is used in this thesis, can be shown to be

$$\mathcal{O}\left(\left(\frac{mw^{\mathcal{O}(k^2)}}{\epsilon}\right)^{n+m+N} D\right), \quad (7.1)$$

where k is the number of layers, w is width (maximum number of neurons) of the network, n and m are the input and output dimensions, N is the total number of network parameters and D is the number of training samples for some arbitrary $\epsilon > 0$ [BMP18]. Complexity is thus exponential in network size and therefore can also be exponential in swarm size if $n = n(a)$ where a is the number of agents. Further, in the multi-agent learning problem, architectures require that each agent has its own network and thus complexity

becomes

$$\mathcal{O}\left(a\left(\frac{mw^{\mathcal{O}(k^2)}}{\epsilon}\right)^{n(a)+m+N}D\right). \quad (7.2)$$

In the actor-critic learning architecture considered in this article, each agent requires two of these networks, one of which has global knowledge of all agents in system. Complexity therefore becomes

$$\mathcal{O}\left(a\left(\frac{mw^{\mathcal{O}(k^2)}}{\epsilon}\right)^{a(n(a)+m)+N}D\right); \quad (7.3)$$

an unscalable situation.

Although compute resources in the cloud [Hwa17] have significantly increased to meet the demand required by the machine learning revolution [Sej18], training an arbitrarily large number of agents is still an open problem. In an effort to avoid this problem, this chapter discusses *policy transplantation* to enable the ‘train small, deploy big’ paradigm that is suitable for some cooperative multi-agent problems.

Definition 1 (Policy Transplantation). *The process of copying a trained agent policy into one or more other agents.*

Policy transplantation has the obvious benefit that a smaller number of agents can be trained using less compute resources and time but enables an arbitrarily large number of swarm agents to be deployed. Depending on the application, this makes deploying the swarm more flexible as it is independent of the number of originally trained agents. This is particularly useful where the swarm size is dynamic, for example, when physical platforms may be subject to attrition and need replacing mid-operation. It also allows operators to dynamically scale the number of platforms to the problem as the mission evolves.

Policy transplantation assumes that the learned policy is both generalizable to other arbitrary agents and that the policy itself performs well in the presence of many other individuals; a situation that it did not experience during training. The former assumption raises concerns for heterogeneously-capable agents and task specialization within learned solutions. However, there are some tasks where solutions can be learned using homogeneous agents without the need for specialization. But even in these cases, as is shown in this chapter, whether or not policy transplantation is successful, or *swarm-safe*¹, is not obvious.

Definition 2 (Swarm-safety). *If the performance of the transplanted swarm is dependent upon the quality of the transplanted policy only, rather than issues associated with the suitability of the policy to its new host(s), such as distinctness, the policy (and by extension the associated cooperative learned solution of the trained swarm) is said to be swarm-safe. Swarm-safety implies two conditions:*

1. *The learned policies are indistinct to the agents on which they were trained; and*
2. *The choice of which policy to transplant is arbitrary; they are considered to be identical.*

¹The term attributed to Kent Rosser (Defence Science and Technology Group).

This chapter discusses what is required for the swarm-safe property to hold and conjectures that swarm-safety is dependent on the world-view given to agents. To that end, using a multi-agent cooperative navigation task as a case study, it is demonstrated how different world-views within the **RL** framework affect the swarm-safe property.

RL is a framework where agents maximize some notion of reward by continually interacting with their environment [SB17a]. Concretely, an agent observes its environmental state and then chooses an action. The environment ‘responds’ to this action by moving the agent to a new state and provides a reward signal. Using the action-state pair and reward, the agent must try to maximize the total reward received from the environment by choosing appropriate actions from a specific policy. In order to observe its environment, an observation vector is defined and provides the agent with its view of the world. In **DRL**, this observation vector typically defines the input layer of a **DNN**. The **DNN** acts as a function approximator that learns the correct policy through repeated exposure to environmental experience. The observation vector represents any information that is available to the agent and is (or is thought to be) useful in learning a task-specific policy. The vector itself is human-designed and how information is represented in the vector is therefore up to the designer. Data such as the agent position for example may be explicitly input as an x and y coordinate, or be pre-processed in some other way before being passed to the input layer neurons. It is shown that when observational data are input in this explicit, absolute way, the learned solution is not swarm-safe. However, when data are pre-processed into a implicit, relative world-view, the learned solution is swarm-safe even though both solutions achieve the same outcome. We therefore make the following definitions on agent observations vectors:

Definition 3 (Absolute (Explicit) World-View). *Peer data are represented using a frame-of-reference that is common to all agents.*

Definition 4 (Relative (Implicit) World-View). *Peer data are pre-processed using the agent’s own data in such a way as to produce an agent-centric frame-of-reference.*

In addition to this simulated analysis of policy transplantation under different world-view representations, a swarm-safe policy generated in simulation is transplanted into robotic aerial platforms and tested in a real-world field experiment. These field-trials show encouraging results for not only the deployment of reinforcement-learned swarm control, but also for *scalable*, swarm-size-independent policy design. While there are some examples of successful policy transplantation, such as in [Li+17; Răd+19], to the best of our knowledge this is the first discussion and attempt at quantifying the swarm-safe condition.

7.1.1 Algorithm

To examine different world-views and their effect on the swarm-safe property, we use the surveillance-based ‘cooperative navigation’ task considered in [Low+17]. This is convenient for comparison to published results as well as being of particular interest to the narrative of this thesis.

In the surveillance task, a set of agents $\mathcal{A}$ must move to cover a set of landmarks² $\mathcal{L}$ while avoiding collisions between peers. To cover a landmark, an agent must move to within sensor range r_s of the origin of a landmark. For uniformity and comparison to published results, we adopt the same learning architecture, MADDPG [Low+17] described in Section 3.3. During training, this actor-critic architecture allows the critic access to all agents' actions and observations. That is, it has global knowledge of the system. Once trained, the actors are deployed without the global critic and execute in a distributed fashion using only local information.

This approach to multi-agent task allocation can be considered a broadcast method [CW11] since agents only have access to raw information about their peers. No coordination information is communicated; a very different approach to that found in auctioning or consensus-based algorithms [LT19]. This also limits data transfer between nodes making it suitable for harsh communications environments. Practically this information may be transmitted wirelessly, or an agent may observe its peers, optically for example. In fact, should this positional information be available via non-RF means, coordination can be achieved without RF at all, making it resilient to traditional RF-jamming as well as alleviating RF transmission concerns. In parallel to candidature work, a method and real-world field experiments to localise peer UAS in this manner has been developed [LFR20].

²The term *landmark* is used in the originally published work [Low+17]. Equivalently, the term *target* is used interchangeably within the chapter.

7.1.2 Observation Vector Models

In their published results, the authors of [Low+17] use the following observation vector

$$O = \left[\{p_i\} \forall i \in \mathcal{D}, \{v_i\} \forall i \in \mathcal{D}, \{\{l_{ij}\} \forall i \in \mathcal{D}\} \forall j \in \mathcal{L}, \{\{a_{ij}\} \forall i \in \mathcal{D}\} \forall j \in \mathcal{A} \right] \quad (7.4)$$

where p is the agent position, v is the agent velocity, l is a vector to a landmark from the agent, a is a vector to a peer from the agent and $\mathcal{D} = \{x, y\}$. In the following, we denote the different observation models according to their implicit or explicit description of peers and landmarks. The vector above treats both landmark and peer agent positions explicitly. Ignoring information about the agent itself, the length of the vector is $|\mathcal{L}| + |\mathcal{A}|$ and is referred to as the *EE* model where the order is landmarks then peers. The four possible observations models are summarised in Tab. 7.1. The *EE* model however has two issues:

1. Due to the fixed-length nature of neural network inputs, it is not scalable in the agent or landmarks dimensions. Swarms would need retraining if either of these variables changed.
2. It is problematic when peer information is not available. For example, if agents have a limited communication range, they may not always be able to acquire information about their peers. Zero-padding or indicator neurons would need to be added to the vector to handle this situation.

TABLE 7.1: The four possible observation vector models generated by treating peer agent and landmark positions either explicitly or implicitly.

		<i>Landmarks</i>	
		Implicit	Explicit
<i>Peers</i>	Implicit	Scalable in peers Scalable in landmarks Dynamic peer/landmark information	Scalable in peers Dynamic peer information Not scalable in targets
	Explicit	Not scalable in peers Scalable in targets No dynamic peer information	Not scalable in peers Not scalable in targets No dynamic information

7.1.3 The Bullseye World Representation

Inherently, vectors provide distance and direction information to the agents and hence the learning process. A relative, *implicit* representation of this information is required. To accomplish this, we use a modified version of the ‘histograms over observed neighbourhood relations’ proposed in [HŠN18] and similarly in [Li+17]. An agent’s maximum communications range is divided into a number of distance bins and a histogram is generated for objects that fall into each bin. Visually, this is represented as a number of concentric circles centred on the agent itself. For direction, a similar approach divides the space around an agent into a number of segments. The approach generates a number of ‘distance-segments’ that map to a specific neuron at the input layer. This neuron therefore takes an integer that counts the number of objects in that segment. For example, Fig. 7.1 shows the ‘bulls-eye’ model with $b = 3$ histogram bins and $s = 4$ segments. In the II model, landmarks (red circles) and peers (blue circles) have their own bulls-eye vectors creating an observation vector of static length (again ignoring agent specific information) $\dim O = 2 \times b \times s = 24$, that is,

$$O = \left[\underbrace{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2}_{\text{peers}}, \underbrace{0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1}_{\text{landmarks}} \right]. \quad (7.5)$$

From this representation, it is easy for the agent (black circle) to decide to move towards the bottom right landmark, since it has peers ‘near’ and ‘towards’ the other landmark. Though not considered here, we note that other information such as the location of obstacles and landmark priorities may be encoded in this way. We modify the above representation so that the first bin is equivalent to the sensor range r_s so that any non-zero integer in any segment representing the first bin indicates that a target(s) is covered. Similarly, the largest bin represents infinity so that all landmarks fall within a bin (equivalently, landmark distances are capped to the distance of the largest bin). Note that landmark positions are known in advance and are static. Since the same representation is used for peer locations, the learning process interprets the situation of having peers within the inner circle as something to be avoided, since this is the only situation where negative rewards occur due to collisions. Aggressive collision avoidance behaviour is observed during flight experimentation in Section 7.2.3 even when vehicles are reasonably far apart due to this representation.

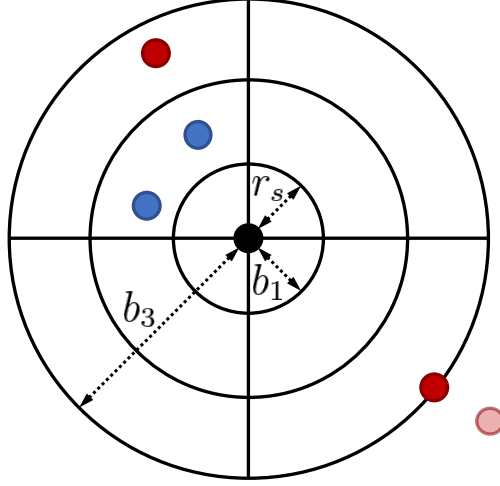


FIGURE 7.1: A visual representation of the ‘bullseye’ observation vector model [HŠN18] where the peer agents (blue circles) and landmarks (red circles) are localised relative to the agent (black circle). The first bin distance $b_1 = r_s$ assists with landmark covering. The relative distance of the bottom right landmark (transparent) d has been set to the largest bin distance b_3 , i.e. $d = \max(d, b_3)$ so that all landmarks can be seen.

For our application, and to ‘train small, deploy big’, only the *EI* and *II* models are of interest. That is, we require flexibility in the number of agents but can concede flexibility in the number of landmarks. We therefore begin by investigating the *EI* model with the rationale that it contains more information than the *II* model, since the bulls-eye representation has an inherently reduced resolution of the world.

7.2 EXPERIMENTAL ANALYSIS

The first experiment compares the performance of the *EE* (from [Low+17]) and *EI* models. Agents were placed on 2×2 unit field with a size and sensor range of 0.1. A disc propagation model with a range $r_c = 0.5$ was applied to each agent to decide if communications were available. Swarms were trained for 1×10^6 episodes of 50 time-steps where actor and critic networks consisted of 4 hidden-layers having 128, 64, 32 and 16 neurons respectively with rectified linear activation functions. Landmarks were randomly placed on the field according to a uniform distribution during each episode. Episodes ended if all landmarks were covered or the maximum number of time-steps was exceeded. Results show the mean and 95% CI of 1000 benchmark episodes.

Agents received a shared reward $R \leq 0$ that encourages minimisation between their position and a landmark, as well as accounting for collisions [Low+17], that is,

$$R = \left(- \sum_{\forall i \in \mathcal{L}} \min(\|\vec{p}_i - \vec{p}_j\| \forall j \in \mathcal{A}) \right) - n_{\text{collisions}} \quad (7.6)$$

where $\vec{\cdot}$ denotes a two-dimensional vector, $\|\cdot\|$ is the l_2 Euclidean norm and $n_{\text{collisions}}$ is the number of collisions experienced during a time-step.

7.2.1 Fully Explicit and Hybrid Models

The *EE* model used the observation vector from Eqn. (7.4) where $|\mathcal{A}| = |\mathcal{L}| = 3$. The *EI* model being a hybrid used the following observation vector

$$O = \left[\{p_i\} \forall i \in \mathcal{D}, \{v_i\} \forall i \in \mathcal{D}, \{\{l_{ij}\} \forall i \in \mathcal{D} \forall j \in \mathcal{L}, \{c_i\} \forall i \in \mathcal{E}_p : c \in \mathbb{Z}^+\right] \quad (7.7)$$

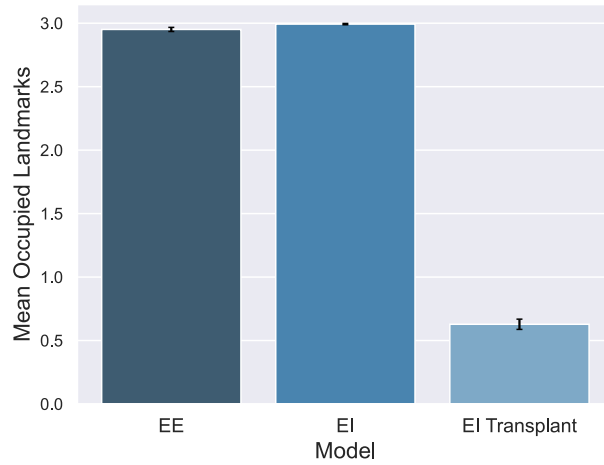
where c is the count of peers in segment i of the peer distance-segment set $\mathcal{E}_p$ where $|\mathcal{E}_p| = (b = 4) \times (s = 8) = 32$. In both cases, a peer-information time-out of 10 time-steps was used. In the *EE* model, missing peer information was zero-padded.

Figure 7.2 plots the number of occupied landmarks and the number of collisions respectively. It can be seen that both models learn to de-conflict targets and perform collision avoidance even though they cannot always communicate. With confidence that the *EI* model performs as well as published literature, we now examine the policy-transplantation of the model. The *MADDPG* implementation was modified so that one of the *EI* agent policies could be transplanted into six agents. Again looking at Fig. 7.2, we can see that transplantation performs very poorly with a high number of collisions; it is not swarm-safe.

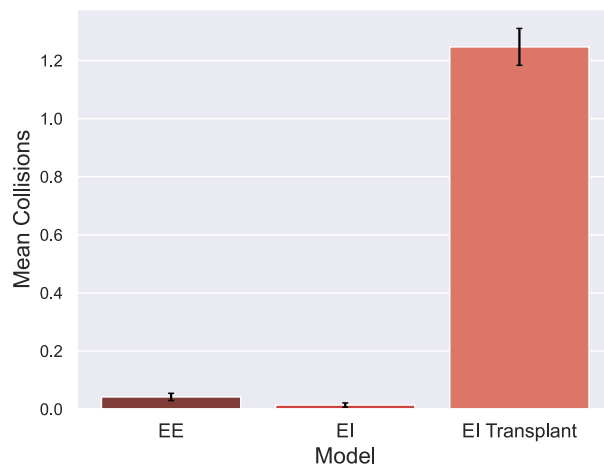
On further investigation, it was found that each agent learned to go to a specific position in the landmark vector. That is, one agent would always go to the landmark encoded in the first position of the landmark set within the observation vector. Another agent would go to the second position and so on. While this is a reasonable strategy for landmark de-confliction, it very much resembles a pre-planned solution. Note that an omniscient oracle did not dictate which agent went to which landmark, but rather the landmarks were ‘selected’ through the reinforcement trial and error process. This is much like the ‘pedestrian dance’ where one attempts to de-conflict their path with another agent (or human) in a non-communicative type way due to a perceived collision [Win18]. A resolution to this dance sees one party choose a side and stick with it, relying on the other to adjust its trajectory. At some point during learning, an agent increasingly began to go to a particular landmark while the others increasingly did not, resulting in a somewhat pre-planned solution. Upon transplanting a single agent policy into many, therefore saw all agents move to a single landmark as dictated by the policy. This caused cover failure as well as a large number of collisions and demonstrates that specialisation can occur even in agents of homogeneous type, and therefore homogeneity does not guarantee swarm-safety.

As well as not being swarm-safe, there are inefficiencies in the learned strategy. In an uncluttered field, an efficient approach would see agents go to their closest landmark. The strategy generated here however, will see agents cross paths to get to ‘their’ landmark if the random starting conditions dictated it; the solution is not reactive to the current environment. For example, the non-optimal solution in Fig. 7.3(a) assigned node 1 to target index 3, node 2 to target index 1 and node 3 to target index 2. Whereas the relative world view produced a non-assignment-based strategy (Fig. 7.3(b)).

The behaviour above also explains why the *EE* model performs well under range-limited propagation, something not considered in the original

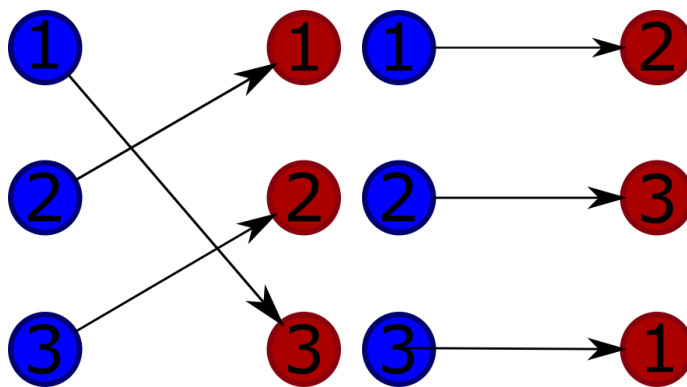


(a) Occupied landmarks



(b) Collisions

FIGURE 7.2: The *EE* and *EI* models perform equally well achieving near 100% coverage with few collisions. The transplanted *EI* model however, does not.



(a) Explicitly target location data created fixed assignments which can cause node cross-overs and is not an optimal strategy. (b) Representing targets implicitly prevented an assignment-based solution. Nodes would seek out their nearest target.

FIGURE 7.3: Different world views generated different coordination strategies.

publication [Low+17]. The position of peers, whether they are in range or not, is inconsequential to landmark selection since an agent’s landmark is pre-determined. Other agents’ locations only matter when they are close for collision avoidance reasons, at which point they can communicate anyway.

7.2.2 Fully Implicit Models

The same experiment was performed for the *II* model using the observation vector

$$O = [\{p_i\} \forall i \in \mathcal{D}, \{v_i\} \forall i \in \mathcal{D}, \{c_i\} \forall i \in \mathcal{E}_l : c \in \mathbb{Z}^+, \{c_i\} \forall i \in \mathcal{E}_p : c \in \mathbb{Z}^+], \quad (7.8)$$

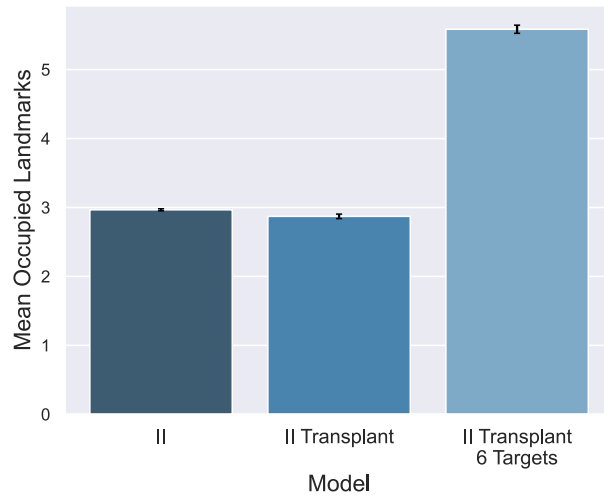
where $\mathcal{E}_l$ is the landmark distance-segment set. As shown in Fig. 7.4(a) and Fig. 7.4(b), the strategy learned using the *II* model performs just as well as the *EE* and *EI* models (first bar). The single transplanted policy also performs nearly as well in a swarm of six agents as it does when trained in three (second bar). There are however a considerably higher number of collisions, but these are attributed to collisions that occur when a single target is being covered by more than one agent. A simple fix to this issue would be to set the sensor radius much greater than the physical size of the agents themselves. This would hold true in a real situation where the sensor is a camera or microphone for example. The higher number of collisions does not occur when $|\mathcal{L}| \geq |\mathcal{A}|$ (third bar) since there is at least one landmark per agent, removing the multiple cover situation. As expected, the situation with more targets take a longer time to execute (see Fig. 7.4(c)). Importantly, this absolute representation of the world does not permit a strategy to be learned where a policy is unique to a given agent.

7.2.3 Physical Implementation

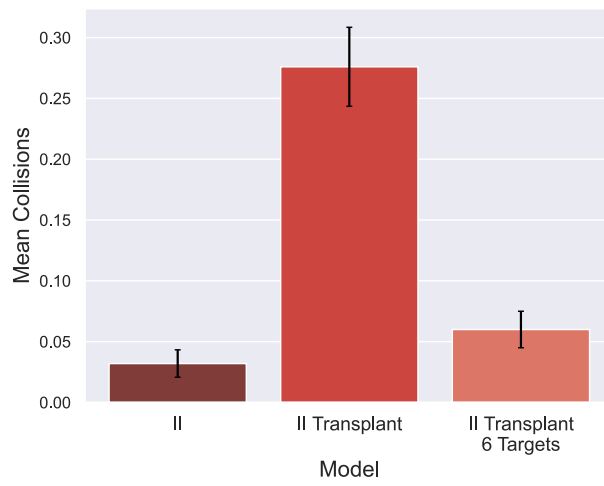
Experiments were performed using F550 Flamewheel hexacopter drones with 22×12 920KV motors and $10 \times 4.5''$ carbon fibre rotors. The vehicles were driven by a Cube Black autopilot [PX420] running ArduCopter 4.0.3 [Ard20] with a Here2 GPS module and powered by two 14.8V 4 cell serial 3Ah batteries. The frames were modified to carry a stripped down Intel core-i7 New Unit of Computing (NUC) computer and an ELP Sony IMX179 8MPix circuit board camera.

Communications were facilitated via high-power, long-range RFDesign RFD900+ serial modems. The Serial Line Internet Protocol (SLIP) [Rom88] running over the serial link allowed both the NUC and the autopilot to share the link (as an Internet Protocol (IP) network) and communicate with monitoring and Ground Control Station (GCS) software. Additionally, multi-point firmware running on the serial radios provided a one-to-all link-layer, which in conjunction with SLIP, emulated a traditional IP LAN. This facilitated intra-swarm communications via User Datagram Protocol (UDP)/IP multicast groups for the sharing of /node locations.

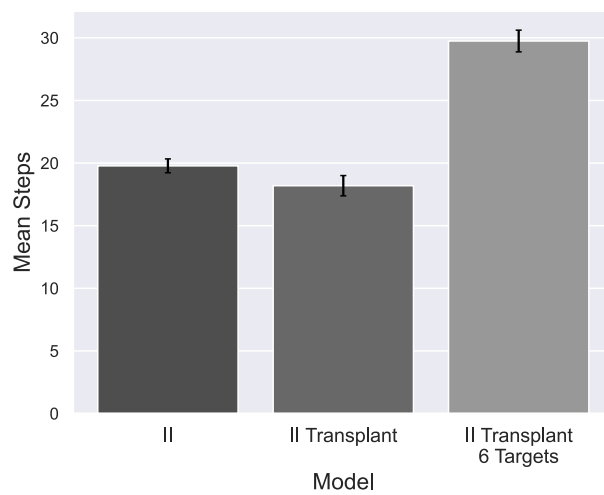
FrSky RXS-R receivers and a Taranis X9D controller bound to all UASs provided an independent autopilot communications link to facilitate safety pilot oversight. Figure 7.5 depicts the hexacopter small Uninhabited Aerial



(a) Occupied landmarks



(b) Collisions



(c) Time

FIGURE 7.4: Transplanting the fully relative *II* model is a swarm-safe operation with six agents covering all three targets. Collisions are increased in this case due to multiple agent coverings of single targets. When there are six targets and six agents, the transplanted policy covers nearly all six targets with minimal collisions. This more difficult problem requires more time to complete the task.

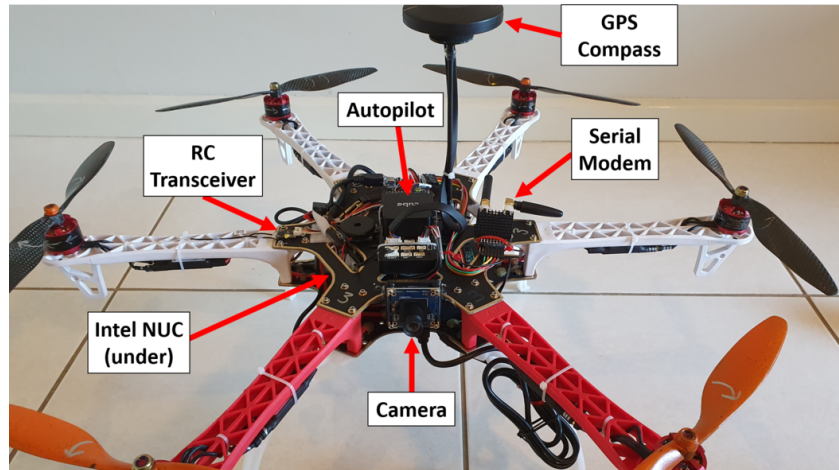


FIGURE 7.5: The F550 flamewheel hexacopter equipped with a Cube Black autopilot, Here2 GPS, RFDesign RFD900+ serial modem, ELP Sony IMX179 8MP USB camera, FrSky R-XSR transceiver and core-i7 Intel NUC. Take-off weight was approximately 2.3kg.

System (sUAS). Flights were conducted at the WTR Woomera, South Australia and were facilitated by Headquarters, Air Warfare Centre, RAAF.

Figure 7.6 depicts the control architecture for the autonomous behaviour. The custom ‘Drone’ application replaces the aforementioned HAIL application and provides a messaging-based API for platform control with the purpose of removing direct control from autonomous agents and in doing so providing a layer of protection against erroneous or unwanted commands. Separating the Drone application and its user(s) into different processes also protects it from crashing in the event of errors introduced by the user application(s). Additionally, Drone had been extensively tested for the control of fixed-wing aircraft in a related UAS rendezvous problem [FMR20]. Drone utilised the Python DroneKit interfacing library [3D 20] which provides many helper functions for building and sending MAVLink messages [Mei+13; Kou+19] to compatible autopilots.

The learned swarm controller continually sends velocity update commands to the autopilot via the Drone application, running in an inference (or test) mode; see Fig. 7.6. For safety, the aircraft were altitude separated by a constant distance based on their node identification numbers and autonomous altitude changes were not permitted; the experiments therefore took place on a 2D plane. Further, maximum velocity was restricted to 1ms^{-1} and autopilot ‘geo-fencing’ was used to mitigate fly-away risk. The aircraft were simultaneously launched by the GCS operator using the custom Drone software and were placed into autonomous mode once all aircraft were at altitude.

Several experiments involving a swarm of three and four nodes performed. All nodes contained a duplicate copy of the learned swarm controller, i.e. a single control policy was transplanted into all nodes. The sensor range of the nodes, i.e. the inner circle in Fig. 7.1, was set to 10m and a maximum communications range set to 100m, i.e. the outer circle represented 100m. Intermediary histogram bins were then equally spaced. The bullseye observation model contained 8 arcs and 5 histogram distance bins. Note that the target locations, shown in Fig. 7.7, were close enough

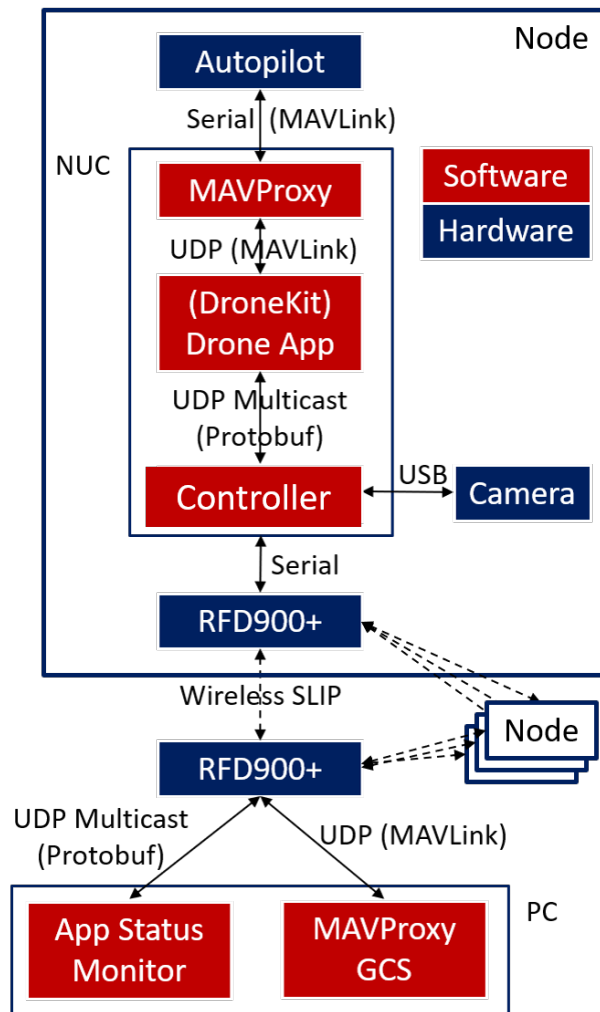


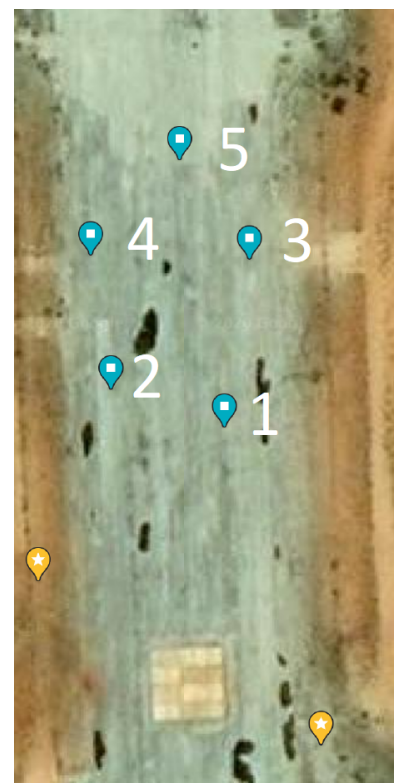
FIGURE 7.6: The platform **Command, Control and Communications (C3)** architecture. **SLIP** provides a seamless **IP** network on the serial link so that custom monitoring software can interact with the custom Drone and control applications, as well as allowing the **MAVLink GCS** [Ard19] to interact with the autopilot. Note that another instance of **MAVLink** was run on the **NUC** in order to provide multiple telemetry and control streams. Custom applications communicated by **Google Protocol Buffers (ProtoBufs)** [Goo20] while autopilot and **GCS** traffic used the **MAVLink** protocol. Multi-point firmware emulated an **IP LAN** for intra-swarm communications via **UDP/IP** multicast groups.

together such that nodes did not go out of **RF** communications range.

Results

Given that the swarm surveillance policies were trained in simulation using a simple particle model, but implemented on real aircraft, movement of the nodes were highly consistent with simulation. This is likely due to the use of rotary-wing aircraft that can, within reason, move almost instantaneously in any direction. Thus there is little mismatch between the motion models in the simulation and the real world. If different aircraft were used, such as fixed-wings, more care must be taken to match the simulation environment with real world dynamics³.

Figure 7.8 shows results from one of the physical flight tests. Nodes were launched in a square pattern (see Fig. 7.12) approximately 10m on a side. Nodes 4 and 5, being behind 1 and 3, immediately began collision avoidance



behaviour by performing large movements to the right and left on the x -axis respectively. This was a common characteristic observed throughout most flight tests. The **UASs** then began to make their way to one of the four surveillance targets (experiments used four of the five target locations).

The outputs of the **ANN** are relative x and y velocity adjustments. These adjustments are added to the current vehicle velocity to get the new desired velocity which is then sent to the flight controller. Several of these control outputs are required to gradually increase or decrease the speed of the vehicle. This creates the ‘drifting’ type behaviour seen around target locations in Fig. 7.8(a). The red ellipses around the target locations represent the region in which a node covers the target, equivalent to a circular sensor range with a radius of 10m. During training, once a node came to within the sensor range, a maximum reward of zero was returned. Thus any **ANN** output within that region is deemed equally good. Only once the vehicle drifts outside that region does a meaningful action move it back inside.

Figure 7.8(b) shows the collaborative swarm reward used within the training, Eqn. 7.6, during the test. The reward increases as nodes get closer to target locations and returns a maximum individual reward of zero when all nodes are within 10m.

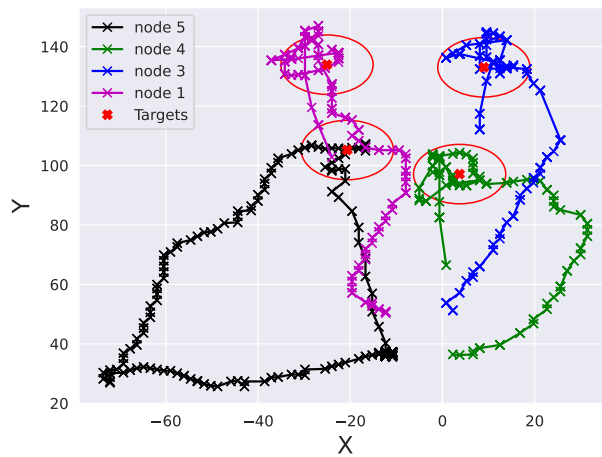
Figure 7.9 shows another test where node 5 was recalled during the experiment due to a hardware concern. The remaining **UAS** continued to coordinate on coverage for four targets. The trace for node 1 shows interesting behaviour where it initially covers the bottom left target. However, because node 5 was still technically in the system, it made the intelligent decision to move to the top left target, leaving the bottom left one for node 5.

Intelligent behaviour is also displayed in Fig. 7.10 where node 4 initially performs collision avoidance with the best option to move to the right, away from both other nodes ; moving backwards would significantly decrease the distance-based reward. Observing the positions of nodes 1 and 3, node 4 selects the lower left target. Again, node 1 selects the top left target.

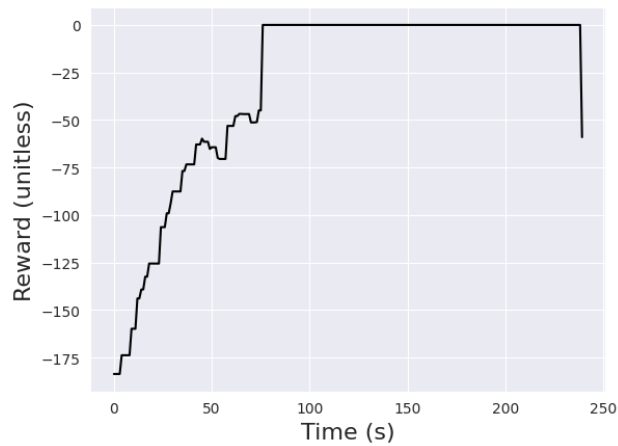
A failure mode is observed in Fig. 7.11 where node 1 stands-off from the bottom left target enacting collision avoidance due to node 3 drifting between the two left-most targets. The target topology is reasonably difficult in this sense. Had targets been further apart, the observational coordination problem would have been easier to solve. Note that the originally published work ([Low+17]) demonstrates cases where a single node covers multiple targets if they are close enough together, making the stand-off behaviour a legitimate action to take. In our case, the lower resolution of the world imposed by the bullseye model and the drifting behaviour causes node 3 to fail to make a final target decision, leaving node 1 no option but to stand-off and wait.

During the experiments shown in the above figures, no parameters required modification. Even when changing the swarm size from four to three, the node-centred, relative view of the world allowed the same algorithm to function correctly. Further, all nodes use the same copy of one of the three pre-trained swarm-safe policies meaning more **UAS** can be deployed (or removed) as required. While the controllers need revising in order to prevent overshoots and making trajectories smoother⁴, these initial experiments

⁴Planned work involves using the **DNN** controller to select targets, with routes being planned using standard algorithms, such as A* [RN10], and reactive collision avoidance using flocking-type behaviour [Rey87a].



(a) Swarm position traces and target locations. Red ellipses around targets indicate a circular sensor coverage region with a radius of 10m. Note that the straighter parts of the traces leading to the launch point indicate the return of the vehicle.



(b) Collaborative swarm reward.

FIGURE 7.8: Position traces and achieved rewards from a 4-node swarm during a flight test at the WTR.

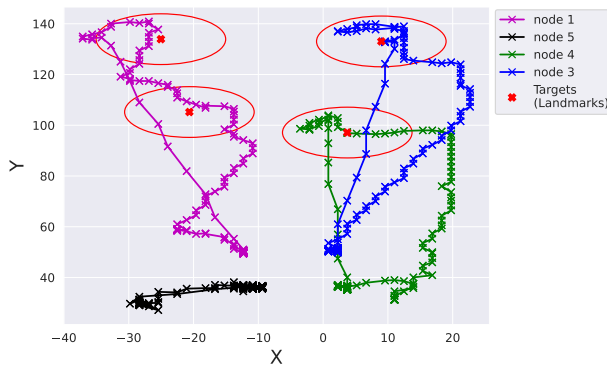


FIGURE 7.9: Node 5 was recalled due to a hardware concern leaving the other three nodes to coordinate on target coverage.

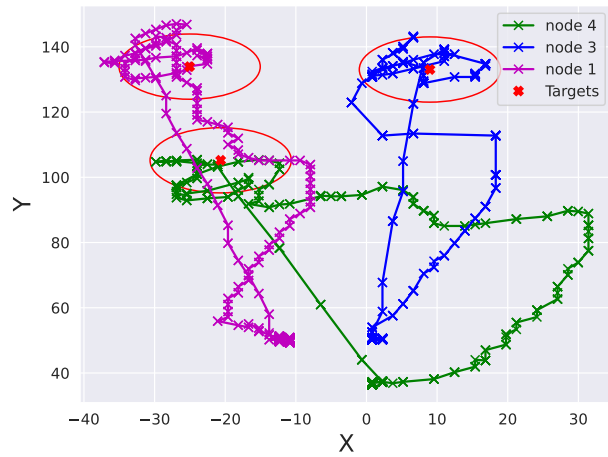


FIGURE 7.10: A three node run displaying intelligent coordination.

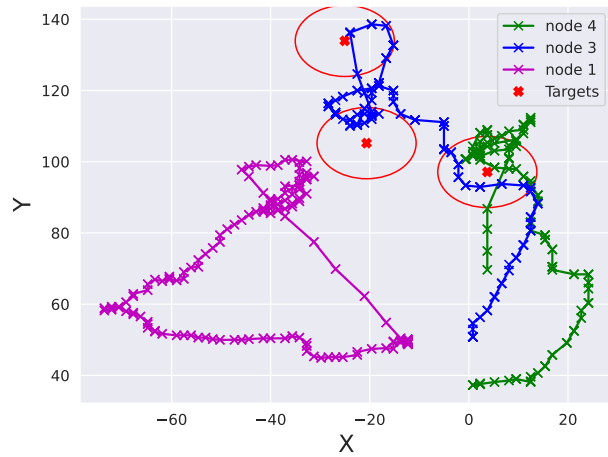


FIGURE 7.11: Node 1 stands-off from the bottom left target due to node 3 drifting between the two left-most targets.



FIGURE 7.12: A swarm of four UAS, each with a duplicate copy of the learned control policy, shortly after take-off at the WTR. UASs were launched from the hard-stand pad, seen as the checkered square at the bottom of Fig. 7.7.

provide encouraging results in not only training multi-agent swarms in simulation for deployment in the real world, but also using policy-transplantation to achieve a swarm-size-independent implementation.

7.2.4 Discussion

Swarm-safety as we have defined it, and as experiments show above, imply that learned agent strategies cannot support specialisation of agents of homogeneous type. When using an absolute world-view for example, agents specialised in the targets they chose as a deconfliction strategy. Such specialisation is not swarm-safe since policies are very much specific to the host agent in which they were trained, meaning that other methods of swarm evolution that encourage specialisation, e.g. [Smi+17; Smi+18; Wan+20], cannot be used here. Similarly, even if world-views were relative, it is assumed that agents of heterogeneous type cannot be mixed and matched; though this case is likely less relevant as the action-set of heterogeneously-capable agents would be different.

In the simulations above, both the relative and absolute world-view approaches achieve the multi-agent cooperative navigation task. However, one is swarm-safe while the other isn't. Typically, the reward function in the reinforcement learning process is selected to encourage desired behaviour. This raises the question of whether there are conditions that can be placed on the reward function itself to enforce swarm-safety. Or, is simply referring to all observable information in a manner relative to the agent enough?

Another interesting avenue of investigation is the optimisation landscape itself and how many solutions or strategies are actually available. Is it possible for a non-unique strategy to evolve using absolute information, but the unique solution is the easiest to find? For example, agents could select the closest landmark and move towards it, then if it encounters a peer closer to its intended landmark, it changes to another. From visual inspection, this is indeed the strategy that is learned in the relative *II* model case. Is non-uniqueness then simply a manifestation of reward-hacking [Amo+16]. If this is the case, the strategy that evolves in the relative approach will never appear when absolute information is available because the absolute case performs better given the way in which the reward function is defined. This line of investigation makes reward function conditions that enforce swarm-safety all the more important. However, we do not want to impose conditions to simply avoid a particular strategy evolving in favour of another, both of which we know are possible from a posteriori knowledge gained through empirical experimentation. Rather, a deeper more fundamental condition is sought that evolves a swarm-safe strategy while achieving the desired task. Thus, this chapter conjectures:

Conjecture 1 (Relative agent world-views). *Agent world-views (observation vectors) must be posed in a way relative to the agent to allow successful policy-transplantation in MADRL problems.*

It is noted here that despite their names, policy-transplantation is different from transfer learning [PY10] or imitation learning [Hus+17] paradigms. The former refers to the re-training of a network that has already been trained

for a particular purpose. For example, this is a highly adopted approach in image classification using the **You Only Look Once (YOLO)** network [RF18]. The **YOLO** network is available with weights for detecting numerous everyday objects trained using the ImageNet database [Sta16]. Using these weights as a starting point, the network can be re-trained with a new training set containing a different object. Indeed, this approach was taken in [LFR20] to detect the quadcopters described in this chapter, in an effort to determine peer locations without the use of wireless communications. In the latter learning paradigm, an expert is observed by the learning system during training. This is distinct from typical **RL** where no such expert exists.

7.3 SUMMARY

This chapter introduced the concept of policy-transplantation, a process by which a policy from a smaller trained swarm is copied into agents of a larger one. If the larger swarm's performance is solely dependent on the quality of the transplanted policy, not the suitability of its new hosts, the policy is said to be swarm-safe. Through simulation of a cooperative multi-agent task, it was found that both a relative and absolute representation of the data contained in the reinforcement learning observation vector produced a high-reward solution. However, only the relative data representation was swarm-safe. In addition, two other cited works that demonstrated successful policy transplantation employed a relative observation vector. It is concluded then that an agent's world-view fundamentally affects the uniqueness of the learned policies, and therefore swarm-safety, with a relative view likely permitting the property.

This chapter also described initial swarming experiments wherein multiple **UAS** aircraft were used to perform the surveillance behaviour in real world conditions. The policies were trained in simulation and transplanted into multiple real aircraft. Position traces of the swarm experiments show intelligent, coordinated decision-making behaviour in a variety of situations. The results are thus encouraging for not only the deployment of reinforcement-learned swarm control but also for *scalable*, swarm-size-independent policy-transplantation.

The requirement to train large swarm systems is one way in which the design of swarm systems may not be practically scalable. Another issue in scalability can occur during algorithm deployment when communications channels are congested or contested. This congestion and/or contention can prevent regular agent interaction and any generated emergent behaviour may breakdown, leading to poor algorithm performance. **Chapter 8** characterises this breakdown in behaviour through the introduction of a metric that quantifies the amount of emergent behaviour inherent to a system.

8

Quantifying the Breakdown of Emergent Behaviour Using State Autocorrelation

- **SYNOPSIS** This chapter considers scalability constraints imposed by the physical deployment of numerous nodes. Specifically, it is examined how inherent emergent behaviour breaks down due to information loss caused by congested communications channels. An interaction-based method is extended to identify emergent behaviour to introduce a quantitative metric for emergent behaviour based on the **ACF** of agent states. Analysis using agent-based simulation characterises the decay of inherent emergence and swarm efficiency with increasing data loss and delay. This chapter is based on the paper:

Bradley Fraser, Claudia Szabo, and Robert Hunjet. “**Simulating the effect of degraded wireless communications on emergent behavior**”. In: *2017 Winter Simulation Conference (WSC)*. Dec. 2017, pp. 4081–4092.

The reader is directed to **Appendix A** for signed Statements-of-Authorship for the publication listed above.

8.1 INTRODUCTION

Swarming-based systems can be found throughout the natural world and extensive research has analyzed the various emergent properties shown by such systems [BDT99; FM08]. This includes the flocking behaviour of birds and fish [Rey87b; OS06], the self-organizing nature of ant colonies [DBT00] and the intelligent foraging of honey bees [KB07]. Swarming-based systems have also been used for the intelligent placement of platforms for uses such as surveillance and target tracking, many of which exploit the simple rules of Reynolds’ Boids model [SB15b; MKK10; GCT15]. Several works have also examined a type of Boids-like algorithm where swarm members consider user-nodes to be part of the swarm [KBJ13b; Hun15b]. Indeed, this thesis utilises such flocking behaviour in **Chapter 5**. These algorithms require individual nodes to know the locations of their nearest-neighbour peers which is typically assumed to be abundant and readily available and these previous studies are performed in the absence of traffic and communications models. However, when wireless communications

are used for data exchange, issues regarding information dissemination arise. These include partial or complete data loss and an increased latency, significantly affecting the quality of the delivered service. A natural-world analogy would be a flock of birds who fly into clouds. In this instance, critical questions include “How much visibility is required for the birds to continue to produce the emergent flocking behaviour?” and “How is the flocking topology affected by the poor visibility?”. RF-based swarms then may not be practically scalable if swarm intelligent algorithms are vulnerable to degraded communication. To investigate this issue, the use of agent-based simulation that includes wireless propagation and communication models to study the relationship between the quality of the environment that facilitates communication, and the emergent behaviour exhibited by the swarm, is presented.

8.1.1 Algorithm

This study analyses the attraction and repulsion algorithm [Hun15b]. This algorithm produces a mesh-building emergent property where all nodes spread out into a regular rhomboid grid with nodes contained within a single connected component in a graph sense. It is closely related to Reynolds’ flocking algorithm [Rey87b] and is of interest for the use of providing resilient communications. The pseudocode for the algorithm is as follows:

Algorithm 4 Attraction-Repulsion Mesh Algorithm

```

1 procedure ATTRACTION-REPULSION NODE (minDist, maxDist)
2   while true do
3     broadcastPosition()
4     nearestNeighbour ← getNearestNeighbour()
5     if nearestNeighbour not null then
6       nearestNeighbourDist ← dist(nearestNeighbour)
7       if nearestNeighbourDist < minDist then
8         moveAway(nearestNeighbour)
9       if nearestNeighbourDist > maxDist then
10        moveTowards(nearestNeighbour)
```

At each step of the algorithm, each node finds their nearest-neighbour and will either move away or towards it, or stay put depending on the distance to the neighbour.

8.1.2 Metrics

As real-life physical implementations of swarming systems become feasible, it is of paramount importance that the system is evaluated to determine its suitability for purpose, if the desired emergent property materializes, and if this emergent property has any undesirable side-effects. To address this challenge, metrics for monitoring the performance of a swarm are needed. As the swarm is being built for a particular purpose or desired emergent property, some of these metrics will be application domain specific. However, existing work has introduced several generic metrics whose values are good indicators that emergent behaviour exists within the swarm. In this study,

we employ both *purpose-driven metrics* and *emergent behaviour metrics* to analyse the suitability of the attraction-repulsion algorithm and its physical implementation for data dissemination. Purpose-driven metrics describe *how* well the swarm is performing, while emergent behaviour metrics give an indication as to *why*. Our hypothesis is that a swarm will not perform well for a given task if its emergence is poor, however, high emergence does not necessarily mean the swarm will perform well for a certain task. These metrics are discussed below.

Purpose-driven Metrics

The purpose of the attraction-repulsion algorithm is to have swarm members form a highly-connected topology with a large coverage area. We propose the following purpose-driven metrics:

1. *Number of connected components* - The number of isolated subgraphs in the supergraph.
2. *Number of articulation points* - Also known as a cut-vertex, an articulation point is a vertex which, if removed (or cut), would cause the graph to become disconnected.
3. *Area coverage* -The coverage of a node is defined here as the area of a circle with radius $r=100$ and origin at the node's location. The topology coverage is then the sum of the circular areas over all nodes with overlapping regions counted only once.

These metrics focus on the physical layer topology but other candidate, purpose-driven metrics could be drawn from the communications literature, for example, traffic throughput and network capacity.

Emergent Behaviour Metrics

The statistics of interaction between nodes as a measure of emergent behaviour was examined in [Cha11] and has subsequently driven more complex techniques of identifying emergent behaviour [STC14]. In this paper we consider effective, directional interactions. They are effective in that an interaction between nodes induces an outcome and leads to a state change. They are also directional in that the interaction is initiated by one of the two interacting nodes [Cha11].

In the attraction-repulsion model, given a set of nodes A , a state change for node $i \in A$ does not occur when either:

1. Its neighbour set $N_i(t)$ is empty and therefore cannot position itself relative to another node .
2. The distance $d_{i,t}^{\min}$ of node i , at time $t \in \mathbb{Z}$, $t > 0$, to its nearest-neighbour is such that $R_{\min} < d_{i,t}^{\min} < R_{\max}$ where $R_{\min}$ and $R_{\max}$ are the minimum and maximum distances respectively within which represent the steady state position for the node .

In all other cases, an effective interaction occurs. An indicator function is defined in [Cha11] for an interaction, which in this case, takes the form

$$\delta_{i,t} = \begin{cases} 0, & R_{\min} < d_{i,t}^{\min} < R_{\max} \text{ or } |N_i(t)| = 0, \\ 1, & \text{otherwise} \end{cases} \quad (8.1)$$

where $\delta_{i,t}$ is the indicator function of node i at time t .

The following time-series statistics are considered:

- The total state change at time t , $I_t = \sum_{\forall i} \delta_{i,t}$; and
- The total cumulative state change at time t , $Z_t = \sum_{\forall i} X_{i,t}$ where $X_{i,t} = \sum_{l=1 \dots t} \delta_{i,l}$.

A time-series of state changes of each node, $x_i = \delta_{i,t}$, $1 \dots T$ is also captured and the use of the ACF on x_i as an emergence indicator is introduced. The ACF computes the similarity between observations as a function of the time lag between them and can be used to detect non-randomness in data [Box+15]. The use of the ACF comes with two assumptions:

1. The next state of a node is correlated with the current state (in some unknown way) if emergent behaviour exists; and similarly
2. Emergent behaviour presents as non-randomness in node interaction statistics [Cha11].

Specifically, we look at the lag 1 correlation which is the ACF coefficient value between time steps t and $t + 1$ in the series. This metric is of particular interest as it does not require global system knowledge. It therefore may be a candidate metric for an individual node to use to decide on-line if emergence is occurring and perhaps adjust its behaviour accordingly.

8.2 EXPERIMENTAL ANALYSIS

In this experimental analysis, the aim is to study the effect of lossless and lossy communications on the emergent behaviour exhibited by the swarm. The simulation environment chosen for this study was MASON. Time advances in discrete steps which have been equated to one second. Time within each step then proceeds according to the transmission time of node transmissions (queuing, propagation and other delays are ignored). Nodes are moved at the end of each second after all transmissions have taken place. Note that physical platform effects such as manoeuvrability are ignored and nodes can occupy the same physical space.

Wireless communications take place over a shared-medium, that is, all nodes are contending for the same medium in order to transmit with Medium Access Control (MAC) protocols regulating a medium's availability to the node. A common type of MAC is the contention-based, Carrier Sense Multiple Access (CSMA). This scheme adopts a 'listen before you talk' policy where (ideally) no two nodes can transmit simultaneously if they are within reception range of each other. A CSMA style transmission rule was modelled for nodes within the simulation; a single node is randomly selected for transmission, then any other node in its reception range is removed from

the pool of possible transmitters. In this way, simultaneous transmissions are removed while making medium access fair for all nodes. Transmission time refers to the time required to put the data ‘into the air’. For example, to transmit 100 bits at a data rate of 100 bits per second, would require one second. In this model, the transmitting node would use the channel (and consequently block other possible transmitters) for a full second.

In the simulations, radio parameters mimic a Wi-Fi-like waveform propagating in an outdoor environment with some clutter using the **LNPLM** and **BPSK** modulation (see [Section 3.1.2](#) and [Section 3.1.2](#)). As such, the radio frequency is 2.412GHz ([IEEE 802.11](#) channel 1), has a path-loss exponent of 2.68, with a shadowing random variable of 3dB and a noise floor of -95dBm. Propagation values are taken from the ‘Woodland’ environment in [[GFC13](#)]. The data rate was left fixed at the slowest Wi-Fi(broadcast) rate of 1Mbs^{-1} and a signal reception **SNR** threshold of 4dB.

The algorithmic factors that are explored in the study are:

1. Protocol Update Rate: A parameter is available to adjust how often a node adds positional updates to the send queue. Protocol packets are 98 bytes, assuming a **UDP IP** and ethernet **MAC** header. This payload contains a single integer for node identification, and three double-precision floats to encode the Cartesian coordinates of the node. Note that payload sizes are an approximation and ignore things such as error correction. Protocol packets are generated at regular intervals and added to the send queue.
2. User-node Traffic: nodes are able to send non-protocol traffic which has a two-fold effect. Firstly, it delays protocol traffic within the send queue of each node (assuming no traffic prioritization). Secondly, it uses up valuable medium time, blocking other nodes from potentially sending protocol traffic.

User traffic is generated using the [IEEE 802.16-4IPP](#) model [[BH01](#)]. This model simulates generation of common self-similar internet **Hypertext Transfer Protocol (HTTP)**, **Transmission Control Protocol (TCP)** and **File Transfer Protocol (FTP)** traffic. It uses a superposition of four **Interrupted Poisson Process (IPP)** operating on different time scales. Each **IPP** is modeled as a two-state Markov chain where a node can be either be in an *ON* state or an *OFF* state. When in the *ON* state, the node produces traffic via a Poisson process at a constant rate of λ packets per second (see [Tab. 8.1](#)). While the model was produced for the 802.16 standard (commonly known by its commercialized name ‘WiMAX’) nearly 20 years ago it is still in use today; most recently, to generate traffic in aerial nodes participating in **DTN QoS** experiments [[MV+16](#)]. A single 4IPP generator is initiated for each node with the 4IPP parameters being scaled for an input user traffic rate. User-node packets use the 4IPP model packet size of 192 bytes. Nodes are initially randomly distributed closely around point (500, 500) in a 1000 units square simulation environment and the attraction-repulsion algorithm has parameters $R_{\min} = 60\text{m}$ and $R_{\max} = 70\text{m}$.

Several other algorithmic parameters will also have a large bearing on the swarm’s performance:

1. Node speed: The algorithm under test is fundamentally based on movement, therefore the movement speed of the nodes will play an important role in the effectiveness of the swarm. Within the simulation, nodes move at the given speed or are stationary and do not use any physical platform modelling.
2. Data caching: Caching of peer information is required for the algorithm to work; caching protocol information for too long leads to data obsolescence. However, cleaning the cache too often leaves the algorithm with little to no information to use. This parameter should be explored for optimal performance but given it is a purely algorithmic parameter it is set to a fixed value of two seconds for all experiments.
3. Swarm size: As the number of swarm nodes increases, the medium contention also increases.

8.2.1 Results

Lossless Communications

We first analyse a lossless communication scenario to provide baseline values for all metrics under ideal conditions. Table 8.2 shows the parameter values for the experiment with the algorithm running under a lossless communications model.

Figure 8.1 shows the metrics after $T = 600$ steps for a particular run. Unsurprisingly, they indicate perfect performance of the algorithm. The network is consistently composed of one connected component with zero articulation points (Fig. 8.1(a)). It also makes use of the allowed propagation range (see Fig. 8.1(b)), expanding linearly until it nears a stable config-

TABLE 8.1: The basic 4IPP model from [BH01]. $c1$ is the transition probability of moving from the *ON* state to the *OFF* state. $c2$ is the transition probability of moving from the *OFF* state to the *ON* state.

Source	λ (packets s^{-1})	$c1$ (transitions s^{-1})	$c2$ (transitions s^{-1})	Mean Packets (packets s^{-1})
IPP1	2.68	4.57×10^{-1}	3.43×10^{-1}	1.15
IPP2	1.67	1.45×10^{-2}	1.08×10^{-2}	0.72
IPP3	1.39	4.57×10^{-4}	3.43×10^{-4}	0.59
IPP4	1.23	4.57×10^{-6}	3.43×10^{-6}	0.53
4IPP				3.0

TABLE 8.2: Parameters for the attraction-repulsion ideal baseline experiment.

Parameter	Value
SNR Threshold (dB)	$-\infty$ (no threshold)
MAC Scheme	None (no contention)
Number of nodes	25
Node speed	1.0
Update Rate (Hz)	∞ (information always available)
User traffic (KBs s^{-1})	0

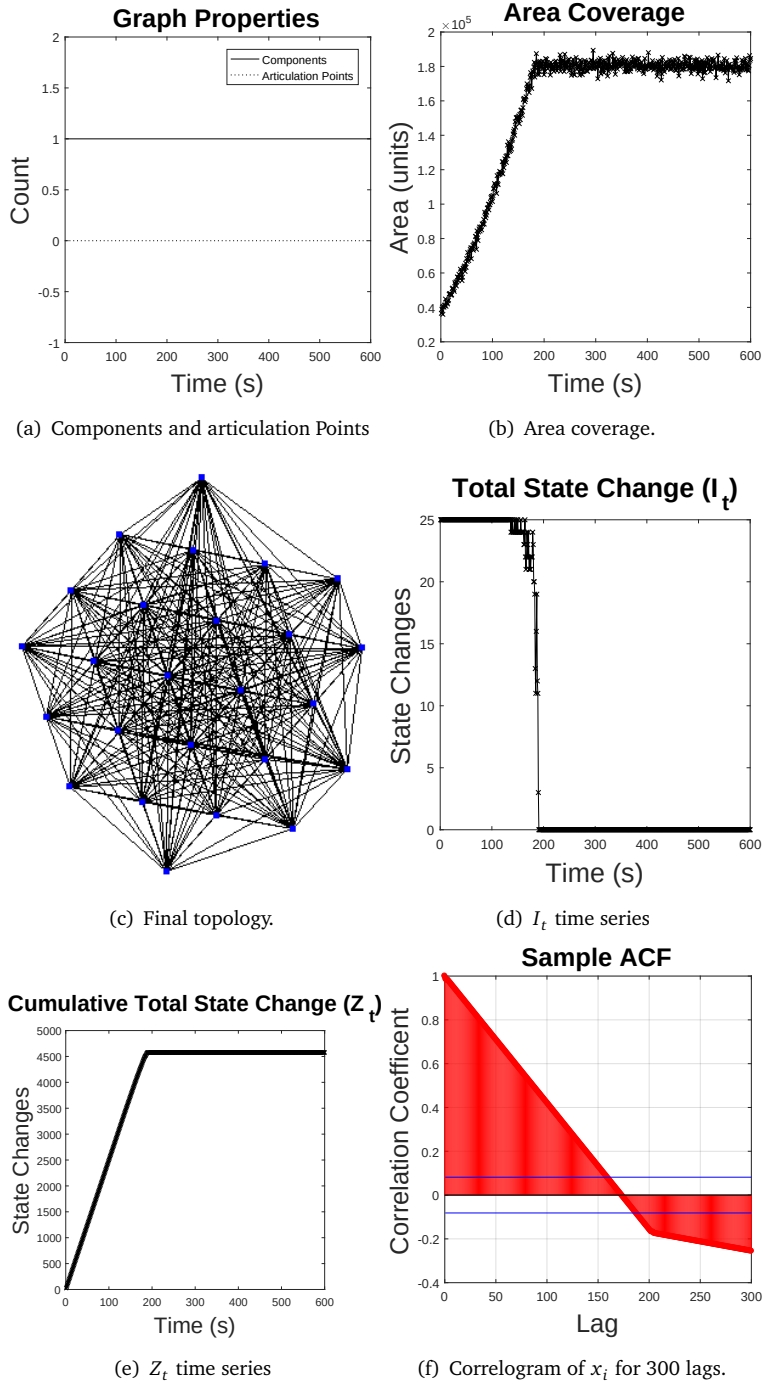


FIGURE 8.1: Lossless experiment results for a particular run.

TABLE 8.3: Comparison of end-of-run metric values for the lossless and two lossy cases with user traffic in KBs^{-1} and protocol updates in Hz.

	Experiment					
	Lossless		10KBs ⁻¹ , 10Hz		50KBs ⁻¹ , 1Hz	
Metric	μ	σ	μ	σ	μ	σ
Area (units) $\times 10^5$	1.80	0.04	1.82	0.04	1.52	0.09
First lag of ACF	0.97	0.01	0.97	0.01	0.55	0.01

uration (some variation is due to Monte-Carlo area estimation). Figures 8.1(d) and 8.1(e) show that there are initially 25 state changes per time step. This is a consequence of all nodes undergoing an effective interaction in order to space themselves apart. The amount of state changes dramatically decreases once all nodes near the desired spacing. More interestingly, the *correlogram* of the time-series state change, Fig. 8.1(f), shows a high degree of correlation between time-steps, even beyond the first lag. The bounds in Fig. 8.1(f) represent a region within which no correlation is present. Note that the first data point in any correlogram (which represents a lag of zero) is always one since any value is maximally correlated with itself. Table 8.3 summarises the lossless and lossy experiments over 10 differently-seeded runs.

Lossy Communications

To better understand the performance of the algorithm in realistic communication scenarios, we next disrupt data dissemination within the algorithm through the addition of user traffic. We expect to see poorer performance of the metrics as compared to the lossless case. However, we also attempt to mitigate this decrease in performance by increasing the protocol update rate.

The experiment varies the protocol update rate over 1, 5 and 10Hz and the user traffic rate over 0, 10, 25 and 50 KBs^{-1} per node using 25 nodes with a movement speed of 1ms^{-1} . The simulations ran for 600 steps and results aggregated over 10 runs.

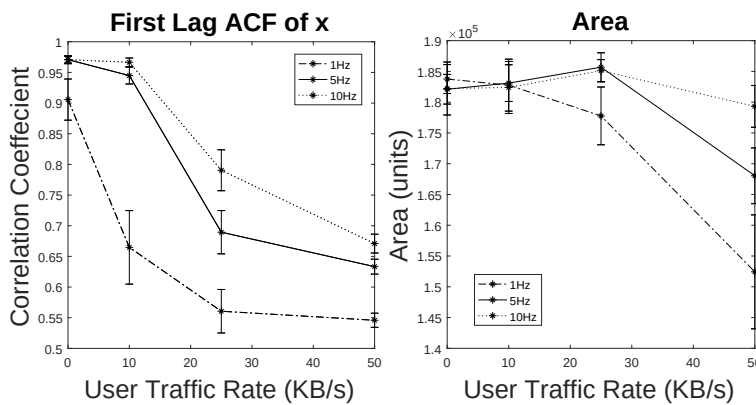
Figure 8.2(a) shows the results for the experiment which reveals several interesting features. The first thing to note is that as the user traffic rate increases, the first-lag ACF coefficient decreases since the user traffic is disrupting the regular protocol traffic. This is mitigated to some extent by increasing the rate at which protocol traffic is transmitted. Now let us examine this reduction in coefficient value with our purpose-driven metrics. In Fig. 8.2(b) we notice that in many cases the area achieved is slightly better than the lossless case above. How can this be? It is postulated that a slight disruption in protocol traffic permits less well-connected topologies such as in Fig. 8.2(d). However given the error bounds on the area estimation, a more precise area measure is required to confirm this result. As the disruption becomes too great, protocol traffic begins to hit the two second cache cleaning limit and nodes have no neighbours to use to position themselves. They therefore remain stationary and produce a more ‘cluttered’ topology like that in Fig. 8.2(e). Again, this is mitigated by an increased protocol

traffic rate. These topologies show a less structured version of the lossless case, Fig. 8.1(c).

Figure 8.2(c) shows the mean number of articulation points formed throughout the experiments. In general, few articulation points are created during the experiments and as user traffic increases and the topologies become more cluttered, fewer points are created since nodes are closer together. It is likely that the few articulation points that are formed are due to the random variable associated with the LNPLM and only exist for a short time.

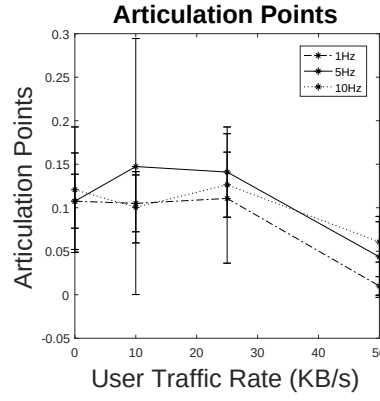
Another noteworthy feature is the very low lag 1 autocorrelation coefficient for the 1Hz update rate and 10KBs^{-1} user traffic rate in Fig. 8.2(a), yet it performs equivalently with 5Hz and 10Hz update cases in the area metric. This is explained by efficiency. While all cases reach near optimal area coverage, the 1Hz update case does so much slower, as is shown in Fig. 8.2(f). The 1Hz update case permits ‘enough’ emergence for the algorithm to work, but not enough to work very efficiently.

For comparison, Figs. 8.2(g), 8.2(h) and 8.2(i) show the most challenged case of an update of 1Hz and user traffic of 50KBs^{-1} for a particular run. The correlogram shows much less correlation than the lossless case in Fig. 8.1(f). The total state change graph initially has 25 state changes as all nodes begin spreading out. However, it also contains times where zero state changes occur since the user traffic blocked the protocol traffic completely. It mildly improves once nodes have spread out (and medium contention has lessened) but the number of state changes still contains a large amount of noise. At this point, the system has converged, albeit to a noisy state. This is easier to discern from the total cumulative state change graph Fig. 8.2(i). Here, we can see two distinct gradients with the knee occurring near time step 150 indicating the convergence point. This can easily be seen for the lossless case, Fig. 8.1(e), as well.

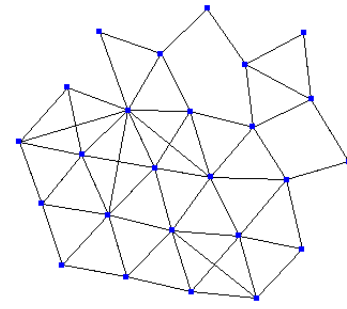
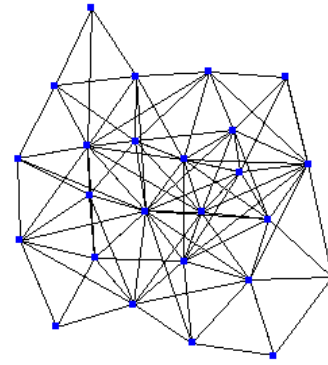
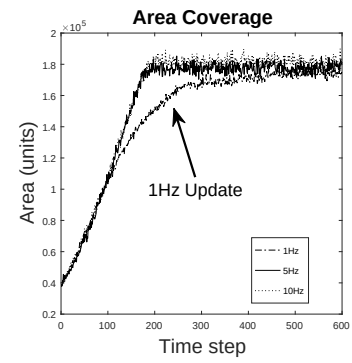
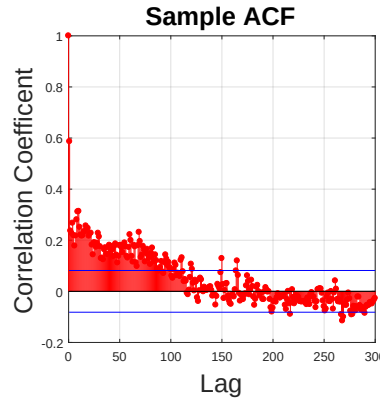
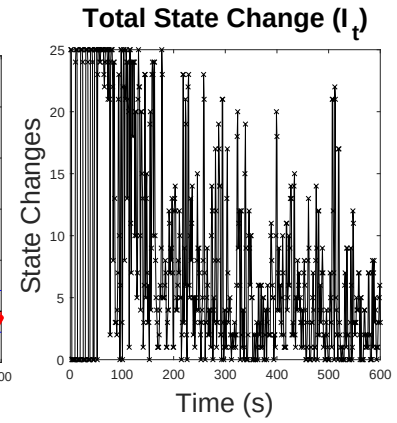


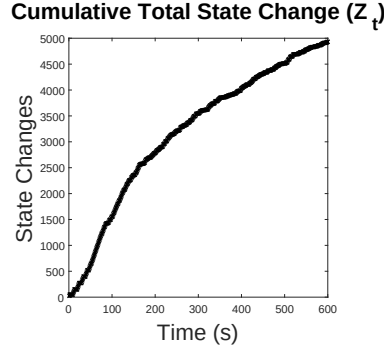
(a) First lag of the ACF of x_i vs user traffic

(b) Area coverage vs user traffic



(c) Articulation points vs user traffic

(d) Final topology for a 5Hz update and user traffic of 25KBs^{-1} (e) Final topology for a 1Hz update and user traffic of 50KBs^{-1} (f) Area coverage evolution for user traffic of 10KBs^{-1} (g) Correlogram of x_t for an update rate of 1Hz and user traffic of 50KBs^{-1} (h) Total state change for a 1Hz update and user traffic of 50KBs^{-1}



(i) Cumulative total state change for a 1Hz update and user traffic of 50KBs^{-1}

FIGURE 8.2: Lossy experiment results. Error bars indicate one standard deviation.

8.2.2 Application to Static Data Ferrying

To further examine the use of the **ACF** in swarming and emergent algorithms, it is applied to the static data-ferrying algorithm of **Chapter 4**. Firstly, an indicator function akin to Eqn. (8.1) must be defined. Given that the data-ferrying algorithm relies on speed changes, speed itself can be considered as a state, and therefore the indicator functions is

$$\delta_{i,t} = \begin{cases} 1, & |s(t-1) - s(t)| > 0 \\ 0, & \text{otherwise} \end{cases} \quad (8.2)$$

where $s(t)$ is ferry i 's speed at time t .

Figure 8.3 shows the correlogram of the first 2000 lags for the equilateral triangle case of 3 user-nodes and 3 ferries. There is an obvious pattern in the **ACF** coefficients where observations that are approximately 317 time steps apart are correlated with each other. This pattern arises since the final speed ferries approach is approximately 1.365 units per step and $1.365 \times 317 = 432.705 \approx 433$ which is approximately equal to the sidelength of the triangle. These high correlations indicate that speed (state) changes occur at regular (non-random) times. Another interesting feature of the correlogram is the 'build-up' of data points around the highly correlated points. These points are generated mid-way between the initial randomness of the system and the final converged state. If there were no correlation, the times at which speed changes occur would be random and therefore no structure within the ferry movement would be observed. This example also demonstrates that state changes may be correlated with far future state changes, not just the state change of the next time step.

Emergent behaviour has been interpreted as the outcome of a process with decreasing entropy, and it has been observed that there is a strong formal similarity between the definitions of thermodynamic and information entropy. This has led to the appropriation of entropy as a measure of macro-level disorder in multi-agent systems [VDPB01]. Given that the **ACF** is typically used to find non-random patterns in noisy data [Box+15], we expect that a system with correlated state changes will exhibit decreasing entropy. This entropy is estimated below.

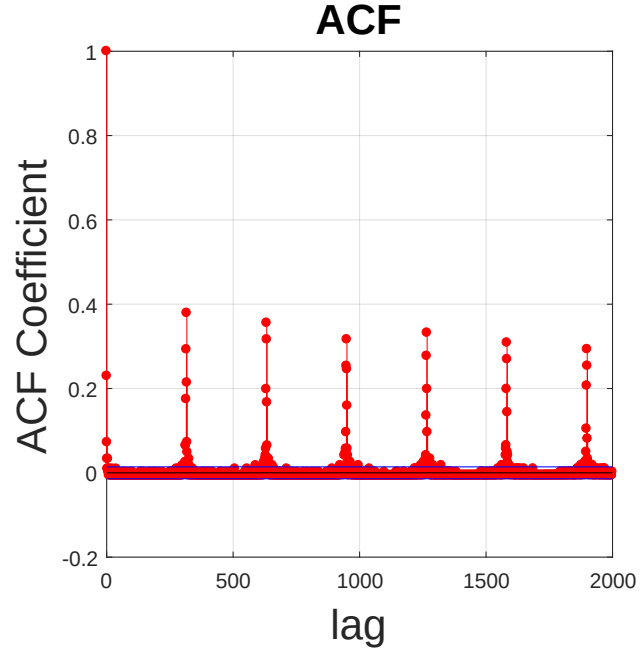


FIGURE 8.3: The correlogram of the stigmergic data-ferry algorithm of Chapter 4. Regular correlated changes in state (speed) indicate structure in the ferry movement.

The information (Shannon) entropy is defined as

$$H = - \sum_i^n p_i \log p_i \quad (8.3)$$

where n is the number of possible system states and p_i is the probability of finding the system in that state [Sha97]. Discrete entropy can also be calculated from the use of histograms [Wal06], where

$$H = - \sum_{i=1}^n p(x_i) \log_2 \left(\frac{p(x_i)}{w_i} \right), \quad (8.4)$$

with $p(x_i)$ being the probability of the system having the value x_i (calculated from the histogram counts) and w_i is the width of the bin i .

The equilateral triangle scenario with 3 ferries from Section 4.2 was run 1,000 times and the inter-arrival time between data-ferries was recorded for each step of a single simulation. The system entropy at each time step can then be estimated from histogram of the inter-arrival times over all runs.

Figure 8.4 shows the estimated entropy. Initially the entropy is zero. This is an edge case of the problem; it occurs because at the start of the simulation, there are many time steps when the inter-arrival time is zero, as the ferries have not yet made contact with any users. A much larger sample size would prevent this artifact. As the ferries do make contact at different times over different runs, the inter-arrival time becomes less certain, increasing system entropy. From there, as the system converges, the overall trend is an entropy decrease; consistent with the previous discussion.

It is noted that the entropy does not reach zero, meaning that the algorithm has not converged completely in the allowed time, and there is still some uncertainty in the ferry inter-arrival time.

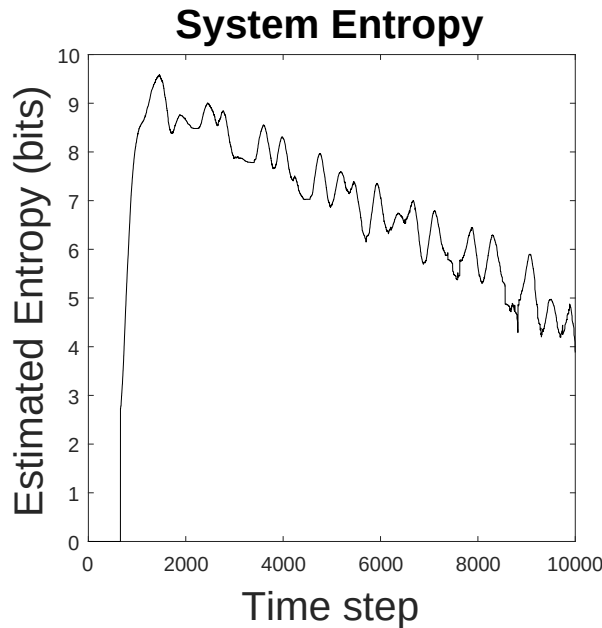


FIGURE 8.4: Decreasing entropy in the stigmergic data-ferry algorithm of Chapter 4.

8.2.3 Discussion

From examining the correlogram for the lossless communications case in Fig. 8.1(f), we can see the first lag coefficient is close to 1, meaning a near maximum correlation between the current state and the next. This is interpreted to mean that a high level of emergence is present in this case. This is visually validated by examining the evolution of the network (Fig. 8.1(c) shows the final topology). In the lossy scenario shown in Fig. 8.2(g), communications are disrupted (by increasing user traffic from 0 to 50 KBs^{-1}) in such a way that the state change time-series appears to be random, or at least contains a large noise signal within it. Consequently, the first lag coefficient has dropped to approximately 0.6. This trend of decreasing autocorrelation value with increasing user traffic can be seen in Fig. 8.2(a) and is interpreted to be the *breakdown* of emergence. This breakdown of emergence then has significant and negative effects on the outcome of the algorithm as measured by the purpose-driven metrics. Thus, for this algorithm, the ACF function provides a *continuous* indicator of inherent emergence. Furthermore, its use on the state change time-series of individual nodes means that the metric is also *local* and therefore does not require global node knowledge to be useful. Further, when examining the state changes in the static data ferrying algorithm, we see that the ACF is also useful in identifying states changes that are highly correlated with each other, even when state changes are highly separated in time.

8.3 SUMMARY

Communication is assumed to be perfect by most swarming mechanisms, however this is not the case in real-world environments. The effect of degraded communications on the production of emergent behaviour is demon-

strated in the attraction-repulsion swarm algorithm. The analysis employs metrics that measure the effectiveness of the swarm for a specific purpose, as well as a metric that indicates if emergence is present. Node interaction statistics pertaining to state changes as a metric is selected for this purpose and introduces the use of the **ACF**. Analysis shows that under perfect communication conditions, the attraction-repulsion algorithm built a well-connected mesh network where state changes in individual nodes were highly correlated with each other. When challenged with user traffic, emergence broke down resulting in a reduction of the autocorrelation lag coefficient values. Therefore an increase in randomness in the time-series of state changes of individual nodes was shown. This affected the purpose-driven metrics in unexpected ways.

This study has demonstrated the need to consider the impact of communication load and protocol contention on the production of a desired emergent property. In this case, efficiency and resultant network topologies differed substantially to the ideal, lossless communications scenario. Practically, this highlights the need for protocol-related traffic prioritization through **QoS**-based approaches to ensure the realization of the desired emergent behaviour. From an analysis perspective, the study also shows that the use of the **ACF** in examining the state change time-series of individual nodes for emergence can be beneficial. This is an important result since global knowledge of node statistics is not required.

9

Conclusion and Future Directions

Since the development of modern **Uninhabited Aerial Systems (UASs)**, the uptake and use of such systems has exponentially increased. In particular, as cost-per-unit has decreased and ease-of-operation has increased, mass deployments of **UASs** have become feasible in the logistical sense. We are today, after many years of improvements in hardware design and real-time control, able to deploy impressive displays of **UAS** swarms that are a technical marvel to behold [BBC18; Int]. However, in order to have these machines work together and remove the control burden from human operators, they must have agency to determine their own actions. Leveraging the well-established and understood fields of **Multi-Agent System (MAS)**, **Artificial Intelligence (AI)**, wireless communications and network modelling, while drawing inspiration from nature, is a promising avenue for such development. Larger and larger physical swarms with traditional control systems, while impressive, are not scalable and exacerbate the need for human control, both online during deployment and offline during mission planning. This thesis presents a step in the direction of adaptable and scalable autonomous control for swarming **UAS** deployments. In particular, the conclusions and insights of the thesis are given below.

9.1 ON THE EQUIDISTANT FERRY SPACING FOR STATIC PATH DATA FERRY ROUTES

A monumental challenge facing current day usage of **UASs** is that of public safety given the risk associated with air traffic accidents. With autonomous air traffic control very much still in its infancy [BW19], static-path routes are an attractive approach for data-ferrying applications due to their known trajectories. Our approach in **Chapter 4** to equidistant ferry spacing on static-path routes maximises throughput and is a distributed solution that is extremely resilient to timing perturbations. This can be seen in the time evolution of the captured solution (see **Figure 4.4**) and results of simulated (see **Figure 4.7**) and physical experiments (see **Table 4.7**). This would likely make it suitable to the ferrying of physical materials as well, where the time taken to load and unload the aircraft would introduce significant perturbations into the system.

In the current form, the final ferry speeds are dependent on the number of ferries in the system (see Eqn. 4.6). Thus, there is a trade-off between using fewer faster-moving ferries at the expense of **UAS** battery life, or multiple

slower-moving ferries at the expense of service delivery. There is scope for future work to overcome this problem to ensure final ferry speeds reach their maximum once synchronisation is achieved, via online manipulation of the pheromone decay rates. However, given that this is also a distributed problem, it is unclear how unsynchronised decay rates across user-nodes would affect the favourable convergence properties of the dynamical system.

Some future work on this algorithm has already been undertaken through the introduction of counter-rotating ferry teams [Hie18]. To decrease message latency of the system, two teams of ferries can be employed to traverse the user-node perimeter with one team moving in a counter-rotating direction. The stigmergic algorithm described in this chapter can be used to equally-space nodes of the individual teams with each team having their own pheromone. However, in order to coordinate across teams so that user-nodes experience equal service times from both teams, there must be a mechanism for these pheromones to interact. The work in [Hie18]¹ describes such a method in which a controlling user-node makes small adjustments to each team's pheromone so that when one is maximal, the other is minimal. In this way, not only do the ferries of the same team equally-space themselves, but the teams themselves are coordinated for uniform service time at that node. Extending this to all user-nodes, if possible, would be a significant development.

¹This work was undertaken as part of Major (MAJ) Andrew Hietpas' United States Marine Corps (USMC) Master's Thesis program at the Naval Postgraduate School (NPS). See Section 1.4 for a statement of involvement in this work.

9.2 ADAPTABLE DATA TRANSFER UNDER VARIABLE GRAPH CONNECTIVITY CONDITIONS

Assuming that aircraft are safe to fly anywhere in an **Area-of-Operations (AO)**, dynamic-path ferrying can be used to further improve data-ferrying throughput given that resource-nodes are no longer confined to specific paths. In Chapter 5 we devised a swarming-based dynamic-path algorithm that adapted its behaviour based on the underlying graph connectivity of the user-node network. The approach outperforms exact **Travelling Salesman Problem (TSP)**-based solutions across a continuum of expected node degrees in **Random Geometric Graphs (RGGs)** (see Figure 5.5). Under particular assumptions, modelling results are reasonably accurate (see Figure 5.15 and Figure 5.16). However, results rely heavily on the properties afforded by randomness through the use of **RGGs**, and similarly to the algorithm in Chapter 6, this work would benefit from investigations into more realistic topologies. Before this, future work should investigate topologies with heterogeneous graph connectivities, considering that networks which cover large geographical distances are unlikely to have homogeneous connectivity.

Additional work in Chapter 5 considered the use of a well-known flocking algorithm to create a collective flock buffer when individual agent buffers were limited in size. However, it only considered a single flock consisting of all agents in the system. As was expected, the usefulness of the collective flocking buffer in terms of data latency decreases with increasing individual agent buffer size (see Figure 5.26). For example, with 5 ferries, a collective flock buffer was favourable when the individual buffer size was 1500 units or less. However, with 10 ferries, a collective flock buffer only outperformed individual buffers when the individual buffer size was 700 units or less. An interesting future direction should investigate heuristic metrics for dynamic

flock creation and deletion with dynamic flock sizes. This would allow the data-ferrying service to be more reactive to network traffic flows and enhance overall throughput. Knowledge generated here could also likely be applied to other applications that require multi-flocking capability. For example, the **Radio-Frequency (RF)** emitter search algorithm of [Fra18b] would benefit from flock splitting and recreation in the search for multiple simultaneous transmitters.

9.3 EXTENDING COORDINATION ACROSS HETEROGENEOUSLY-CAPABLE NODES FOR MULTI-ROLE MISSIONS

Chapters 4 and 5 only consider the coordination of resource-nodes in space, however Chapter 6 introduced the further complication of heterogeneous swarms of multi-capable nodes. To overcome this problem and to provide a mechanism for human oversight of role importance, we extended the algorithm of Chapter 5 using a multi-criteria decision-making architecture. The algorithm was shown to outperform dedicated teams of equivalent size when tested in an environment with randomly placed target-nodes (see Figure 6.3 and Figure 6.4). For example, with 6 resource-nodes and equal role priorities, service time was improved by approximately 15% when communications ranges were low and 47% when high. Interestingly, the approach was shown to exhibit adaptive behaviour in role selection depending on need (see Figure 6.7). To further benchmark the algorithm, future work should explore its performance with more realistic topologies.

When considering the extension of the algorithm posed in Chapter 5 to the multi-role problem, we considered an alternate solution where team sizes were optimised. This provides a better method of selecting team sizes based on the marginal benefit of adding more resources, rather than the naïve linear proportioning taken in Chapter 6. This would provide a stronger comparison algorithm and may yield a viable approach to the multi-role problem itself. However, considerable effort was expended in accurately deriving a model to describe the algorithm’s performance with respect to the number of swarm nodes for the communications role. This effort would need to be repeated for each different role involved and reconciled into a singular utility function. This represents a viable piece of future work.

The approaches mentioned above coordinate multiple systems in achieving a high level goal, be it for communications, surveillance or logistics operations. Any operator now need only activate the swarm to do so as opposed to directly controlling any of the assets individually. Qualitatively, one can see how this substantially reduces cognitive load on the operator.

9.4 TOWARDS SCALABLE POLICY TRANSPLANTATION

The main result of Chapter 7 pertains to **Multi-Agent Deep Reinforcement Learning (MADRL)**. Specifically, the conditions which ensure that learned agent policies can be copied into multiple agents without loss of performance due to policies being specific to particular agents. This thesis conjectures (see Conjecture 1), and demonstrates via empirical study, that relative world-views are required (or are at least somehow related to the underlying

mechanism). It is shown that when the observation vector of agents is structured in such a way as to make positional information relative to the agent's own position, policies can be copied into other agents. However, if the observation vector contains absolute positional information, learned policies are not generic but rather specific to the agent on which they were trained and copying the policies results in failure. This result allows the deployment of large swarms without the need for training all its members and adds scalability to the design process.

Conjecture 1 was formulated from empirical evidence and so the theoretical constraints required on either the reinforcement learning architecture, reward function or observation vector to enforce this type of scalability is an open question. To the author's knowledge, there are no works that explicitly consider this problem, making it a truly unique contribution as well as a prime target of future work.

The physical experiments of **Chapter 7** demonstrate that control policies that were learned in simulation are readily deployable in the real world despite the use of low fidelity platform models. However, as part of future work, improved platform models should be used for policy training to better generalise control to the real world. Trajectory tracks seen in these experiments show a drifting, and somewhat inefficient movement, due to the way the agent actions were defined. This could be improved by moving the coordination logic further up the autonomy stack. For example, rather than having the **Deep Neural Network (DNN)** output motion vectors for the agent, it could instead output a desired target or a general **AO** region; leaving the actual path-planning to traditional algorithms such as A^* [RN10]. However, care must be taken to ensure that agent control and observation is posed in a swarm-safe manner.

Where possible, algorithms were implemented in real systems in order to characterise the sim-to-real problem and validate conclusions in the real world. Like pre-planned mass **UAS** deployments, it is believed that through this implementation, this thesis has contributed to the study of Swarm Robotics via the development of novel **Command, Control and Communications (C3)** architectures for swarm experimentation using **Commercial-Off-The-Shelf (COTS)** devices (see **Figure 7.6**). Standardised **C3** software that facilitates physical (**UAS**) swarming research and development is currently sorely missing from the open-source toolkit, making it a readily available avenue for technical, if not research, future work.

9.5 QUANTIFYING THE BREAKDOWN OF EMERGENT BEHAVIOUR USING STATE AUTOCORRELATION

Chapter 8 introduced the novel use of the **Autocorrelation Function (ACF)** to search for non-random structure in agent state changes, with the assumption that agent state changes are correlated with each other in some unknown way in emergent systems. Despite the underlying simple idea, this approach is shown to identify structure within agent states as the system evolves in time, at least for the two tested algorithms (see **Figure 8.1** and **Figure 8.3**). For example, in a flocking-like emergent system with perfect communication, agent state changes were highly correlated with future state changes.

However, correlation strength dropped rapidly as the system devolved into randomness. Unsurprisingly, the appropriated notion of thermodynamic entropy to MAS shows a decrease in system entropy when the ACF shows non-random structure. While the ACF measure may not necessarily indicate that the system structure is caused by an emergent process, its continuous nature is convenient to measure the ‘amount’ of structure within the system. This construct was used to characterise the decrease in performance of an emergent algorithm as its protocol information is disrupted. An important practical consideration for scalability when communicating via RF. Given that entropy is also a measure related to system randomness, future work could investigate the relationship between the two metrics, likely being a similar descriptor of the same property.

There are numerous evolutionary-based approaches to developing swarm behaviour, for example, using fuzzy logic [Ewe+13] or MADRL such as is used in Chapter 7. In these approaches, an optimiser typically fits agent and/or swarm behaviour to a defined reward or objective function. It may be possible then to incorporate randomness measures such as entropy or the ACF into the objective function to help explore the optimisation space and guide the solution towards structured behaviour. At the very least, monitoring these measures as the optimisation progresses may yield insights into the process as structure evolves.

9.6 CONCLUDING STATEMENT

UASs are becoming the go-to platform for the commercial and military sectors to improve safety, enhance efficiency and save money in their operations. To mitigate the human burden as UAS deployments become large, research and development needs to be invested into the autonomy and self-governance of individual platforms, as well as the swarm as a whole. This thesis is seen by the author and his advisors as step in this direction of adaptable, scalable and effective control of autonomous drone swarms by providing the following contributions:

1. A novel algorithm for equally-spaced data-ferries;
2. A connectivity-aware, hybrid ferrying and bridging algorithm for disconnected wireless communications;
3. An approach to multi-role swarm coordination based on multi-criteria decision making for positional-based UAS;
4. Empirical evidence for the required conditions under which copying of learning agent policies is permissible; and
5. A novel metric of emergent behaviour strength based on the ACF of agent state changes, and its use in investigating the breakdown of emergent behaviour in poor wireless communication environments.

A

Statements of Authorship

See below signed authorship statements for each of the publications listed below. Note that [Hun+18] is missing several signatures due to some authors becoming uncontactable. It does however contain the principle author signature.

Statement of Authorship

Title	Data ferrying in tactical networks using swarm intelligence and stigmergic coordination
Status	Published
Citation	B. Fraser and R. Hunjet, "Data ferrying in tactical networks using swarm intelligence and stigmergic coordination," in 2016 26th International Telecommunication Networks and Applications Conference (ITNAC), Dunedin, New Zealand, Dec. 2016, pp. 1–6.

Principal Author

Name	Bradley Fraser		
Contribution	Original concept. Implementation and simulation. Analysis. Manuscript generation.		
Overall percentage	90		
Certification	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Signature		Date	18/09/21

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

- i. the candidate's stated contribution to the publication is accurate (as detailed above);
- ii. permission is granted for the candidate to include the publication in the thesis; and
- iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name	Robert Hunjet		
Contribution	Project guidance (design, research context) Manuscript		
Signature		Date	16/09/21

Statement of Authorship

Title	Data Ferrying with Swarming UAS in Tactical Defence Networks
Status	Published
Citation	R. Hunjet et al., "Data Ferrying with Swarming UAS in Tactical Defence Networks," in 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, May 2018, pp. 6381–6388.

Principal Author

Name	Robert Hunjet
Contribution	Project management. Guidance of DST contribution. Manuscript generation (introduction, context)
Overall percentage	N/A
Certification	N/A
Signature	Date 16/09/21

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

- i. the candidate's stated contribution to the publication is accurate (as detailed above);
- ii. permission is granted for the candidate to include the publication in the thesis; and
- iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name	Bradley Fraser
Contribution	Algorithm design (of DST contribution). Implementation (of DST contribution). Data collection (simulation, physical) (of DST contribution). Data analysis (of DST contribution). Manuscript generation (of DST contribution).
Overall percentage	25
Certification	This paper <u>partly</u> reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis.
Signature	Date 19/09/21

Name	Thomas Stevens
Contribution	Implementation (of DST contribution). Data collection (physical) (of DST contribution). Manuscript generation (of DST contribution).
Overall percentage	N/A
Signature	Date

Name	Laura Hodges		
Contribution	Implementation (of DST contribution). Data collection (physical) (of DST contribution).		
Overall percentage	N/A		
Signature		Date	

Name	Karina Mayen		
Contribution	Algorithm design (of Monash contribution). Implementation (of Monash contribution). Data collection (simulation, physical) (of Monash contribution). Manuscript generation.		
Overall percentage	N/A		
Signature		Date	

Name	Jan Carlo Barca		
Contribution	Project management. Guidance of Monash contribution. Manuscript generation (introduction, context).		
Overall percentage	N/A		
Signature		Date	

Name	Madeleine Cochrane		
Contribution	Data collection (physical) (of DST contribution).		
Overall percentage	N/A		
Signature		Date	

Name	Ricardo Cannizzaro		
Contribution	Data collection (physical) (of DST contribution).		
Overall percentage	N/A		
Signature		Date	

Name	Jennifer Palmer		
Contribution	Project management. Guidance of DST contribution. Manuscript generation (introduction, context, final editing).		
Overall percentage	N/A		
Signature		Date	

Statement of Authorship

Title	Adaptive Swarm Control for Mobile Resource Placement in Wireless Ad-Hoc Networks
Status	Published
Citation	B. Fraser, A. Coyle, C. Szabo, and R. Hunjet, "Adaptive Swarm Control for Mobile Resource Placement in Wireless Ad-Hoc Networks," in 2019 IEEE 20th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), Jun. 2019, pp. 1–6.

Principal Author

Name	Bradley Fraser		
Contribution	Algorithm design. Implementation and simulation. Analysis. Manuscript generation.		
Overall percentage	80		
Certification	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Signature		Date	18/09/21

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

- i. the candidate's stated contribution to the publication is accurate (as detailed above);
- ii. permission is granted for the candidate to include the publication in the thesis; and
- iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name	Andrew Coyle		
Contribution	Algorithm design. Manuscript editing.		
Signature		Date	10/9/2021

Name	Claudia Szabo		
Contribution	Guidance. Manuscript editing.		
Signature		Date	09/09/21

Name	Robert Hunjet		
Contribution	Guidance. Manuscript editing.		
Signature		Date	16/09/21

Statement of Authorship

Title	Data Ferry Flocking for Bulk Information Transfer Under Ferry Buffer Constraints
Status	Published
Citation	B. Fraser, "Data Ferry Flocking for Bulk Information Transfer Under Ferry Buffer Constraints," in 2018 28th International Telecommunication Networks and Applications Conference (ITNAC), Nov. 2018, pp. 1–6.

Principal Author

Name	Bradley Fraser		
Contribution	Algorithm design. Implementation and simulation. Analysis. Manuscript generation.		
Overall percentage	100		
Certification	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Signature		Date	18/09/21

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

- i. the candidate's stated contribution to the publication is accurate (as detailed above);
- ii. permission is granted for the candidate to include the publication in the thesis; and
- iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Statement of Authorship

Title	Leveraging Emergent Specialization in a Heterogeneous Multi-Role Swarm Control Architecture for Positional-based UAS Missions
Status	Published
Citation	B. Fraser, C. Szabo, A. Coyle, and R. Hunjet, "Leveraging Emergent Specialization in a Heterogeneous Multi-Role Swarm Control Architecture for Positional-based UAS Missions," in 2020 IEEE Symposium Series on Computational Intelligence (SSCI), Canberra, Australia, Dec. 2020, pp. 2584–2591.

Principal Author

Name	Bradley Fraser		
Contribution	Algorithm design. Implementation and simulation. Analysis. Manuscript generation.		
Overall percentage	80		
Certification	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Signature		Date	18/09/21

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

- i. the candidate's stated contribution to the publication is accurate (as detailed above);
- ii. permission is granted for the candidate to include the publication in the thesis; and
- iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name	Claudia Szabo		
Contribution	Experiment design. Manuscript editing		
Signature		Date	09/09/21

Name	Andrew Coyle		
Contribution	Guidance. Manuscript editing.		
Signature		Date	10/9/2021

Name	Robert Hunjet		
Contribution	Guidance. Manu'		
Signature		Date	16/09/21

Statement of Authorship

Title	Train Small, Deploy Big: Do Relative World Views Permit Swarm-Safety During Policy Transplantation for Multi-Agent Reinforcement Learning Problems
Status	Published
Citation	B. Fraser and G. Laurito, "Train Small, Deploy Big: Do Relative World Views Permit Swarm-Safety During Policy Transplantation for Multi-Agent Reinforcement Learning Problems?," in AI 2020: Advances in Artificial Intelligence, 2020, pp. 269–280.

Principal Author

Name	Bradley Fraser		
Contribution	Theoretical conjecture. Implementation and simulation. Analysis. Manuscript generation.		
Overall percentage	95		
Certification	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Signature		Date	18/09/21

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

- i. the candidate's stated contribution to the publication is accurate (as detailed above);
- ii. permission is granted for the candidate to include the publication in the thesis; and
- iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name	Giuseppe Laurito		
Contribution	Manuscript preparation/editing.		
Signature		Date	08/09/2021

Statement of Authorship

Title	Simulating the effect of degraded wireless communications on emergent behavior
Status	Published
Citation	B. Fraser, R. Hunjet, and C. Szabo, "Simulating the effect of degraded wireless communications on emergent behavior," in 2017 Winter Simulation Conference (WSC), Dec. 2017, pp. 4081–4092.

Principal Author

Name	Bradley Fraser		
Contribution	Theoretical formulation. Algorithm design. Implementation and simulation. Analysis. Manuscript generation.		
Overall percentage	85		
Certification	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Signature		Date	18/09/21

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

- i. the candidate's stated contribution to the publication is accurate (as detailed above);
- ii. permission is granted for the candidate to include the publication in the thesis; and
- iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name	Claudia Szabo		
Contribution	Theoretical formulation. Analysis Manuscript preparation and editing.		
Signature		Date	09/09/21

Name	Robert Hunjet		
Contribution	Guidance. Manuscript editing		
Signature		Date	16/09/21

Statement of Authorship

Title	A swarm intelligent approach to data ferrying in sparse disconnected networks
Status	Published
Citation	B. Fraser, R. Hunjet, and A. Coyle, "A swarm intelligent approach to data ferrying in sparse disconnected networks," Int. J. Parallel Emergent Distrib. Syst., pp. 1–22, Oct. 2017.

Principal Author

Name	Bradley Fraser		
Contribution	Algorithm design. Theoretical formulation. Implementation and simulation. Analysis. Manuscript generation.		
Overall percentage	80		
Certification	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Signature		Date	18/09/21

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

- i. the candidate's stated contribution to the publication is accurate (as detailed above);
- ii. permission is granted for the candidate to include the publication in the thesis; and
- iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name	Andrew Coyle		
Contribution	Theoretical formulation. Analysis. Manuscript preparation and editing.		
Signature		Date	10/9/2021

Name	Robert Hunjet		
Contribution	Guidance.		
Signature		Date	16/09/21

Statement of Authorship

Title	An Analytic Latency Model for a Next-Hop Data-Ferrying Swarm on Random Geometric Graphs
Status	Published
Citation	B. Fraser, A. Coyle, R. Hunjet, and C. Szabo, "An Analytic Latency Model for a Next-Hop Data-Ferrying Swarm on Random Geometric Graphs," IEEE Access, vol. 8, pp. 48929–48942, 2020.

Principal Author

Name	Bradley Fraser
Contribution	Algorithm design. Theoretical formulation. Implementation and simulation. Analysis. Manuscript generation.
Overall percentage	80
Certification	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.
Signature	
Date	18/09/21

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

- i. the candidate's stated contribution to the publication is accurate (as detailed above);
- ii. permission is granted for the candidate to include the publication in the thesis; and
- iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name	Andrew Coyle
Contribution	Theoretical formulation. Analysis. Manuscript preparation and editing.
Signature	
Date	10/9/2021

Name	Robert Hunjet
Contribution	Guidance. M
Signature	
Date	16/09/21

Name	Claudia Szabo
Contribution	Guidance. Manuscript editing.
Signature	
Date	09/09/21

Bibliography

- [3D 20] 3D Robotics. *dronekit-python*. <https://github.com/dronekit/dronekit-python>. Accessed: 2020-7-20. 2020 (cit. on p. 136).
- [ASR18] Hussein A Abbass, Jason Scholz, and Darryn J Reid, eds. *Foundations of Trusted Autonomy*. Springer, Cham, 2018 (cit. on p. 74).
- [ACM05] M Abramson, W Chao, and R Mittu. “Design and evaluation of distributed role allocation algorithms in open environments”. In: *Proceedings of the International Conference on Artificial intelligence*. 2005, pp. 565–571 (cit. on p. 32).
- [Abr05] Myriam Abramson. “Three myths about roles”. In: *Workshop on roles and coordination, AAAI Fall Symposium*. 2005 (cit. on p. 31).
- [AOS08] M Abuelela, S Olariu, and I Stojmenovic. “OPERA: Opportunistic packet relaying in disconnected Vehicular Ad Hoc Networks”. In: *2008 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems*. 2008, pp. 285–294 (cit. on p. 13).
- [AM02] William Agassounon and Alcherio Martinoli. “Efficiency and robustness of threshold-based distributed allocation algorithms in multi-agent systems”. In: *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3. AAMAS '02*. Bologna, Italy: Association for Computing Machinery, July 2002, pp. 1090–1097 (cit. on p. 31).
- [AO+15] Omar Ait Oualhaj, Abdellatif Kobbane, Essaid Sabir, Jalel Ben-othman, and Mohammed Erradi. “A ferry-assisted solution for forwarding function in Wireless Sensor Networks”. en. In: *Pervasive Mob. Comput.* 22.SI (Sept. 2015), pp. 126–135 (cit. on p. 17).
- [Aln+15] M Alnuaimi, K Shuaib, K Alnuaimi, and M Abed-Hafez. “Data gathering in wireless sensor networks with ferry nodes”. In: *2015 IEEE 12th International Conference on Networking, Sensing and Control*. Apr. 2015, pp. 221–225 (cit. on p. 14).
- [Amo+16] Dario Amodi, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. “Concrete Problems in AI Safety”. In: (June 2016). arXiv: 1606.06565 [cs.AI] (cit. on p. 141).
- [And79] A M Andrew. “Another efficient algorithm for convex hulls in two dimensions”. In: *Inf. Process. Lett.* 9.5 (Dec. 1979), pp. 216–219 (cit. on p. 54).
- [AP12] T B Apker and M A Potter. “Robotic swarms as solids, liquids and gasses”. In: *2012 AAAI Fall Symposium Series* (2012) (cit. on p. 90).
- [App06] David L Applegate. *The Traveling Salesman Problem: A Computational Study*. en. Vol. 4. Princeton University Press, 2006, p. 606 (cit. on pp. 12, 54, 80, 87).
- [Ard19] ArduPilot Dev Team. *MAVProxy — MAVProxy documentation*. <https://ardupilot.org/mavproxy/>. Accessed: 2020-7-20. 2019 (cit. on p. 137).
- [Ard20] ArduPilot Dev Team. *ArduPilot*. <https://ardupilot.org/ardupilot/index.html>. Accessed: 2020-7-20. 2020 (cit. on p. 134).
- [AYT18] Ross D Arnold, Hiroyuki Yamaguchi, and Toshiyuki Tanaka. “Search and rescue with autonomous flying robots through behavior-based cooperative intelligence”. In: *Journal of International Humanitarian Action* 3.1 (Dec. 2018), p. 18 (cit. on p. 111).
- [ALV13] Martin Aruldoss, T Miranda Lakshmi, and V Prasanna Venkatesan. “A survey on multi criteria decision making methods and its applications”. In: *Healthc. Inf. Manage.* 1.1 (2013), pp. 31–43 (cit. on p. 49).
- [AP11] Peter Atkins and Julio de Paula. *Physical Chemistry for the Life Sciences*. en. OUP Oxford, 2011 (cit. on p. 90).
- [ACG11] Anjali Awasthi, S S Chauhan, and S K Goyal. “A multi-criteria decision making approach for location planning for urban distribution centers under uncertainty”. In: *Math. Comput. Model.* 53.1-2 (Jan. 2011), pp. 98–109 (cit. on p. 49).
- [AC15] Allan M Axelrod and Girish Chowdhary. “Adaptive Algorithms for Autonomous Data-Ferrying in Nonstationary Environments”. In: *AIAA Infotech @ Aerospace*. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, Jan. 2015 (cit. on pp. 21, 26).
- [BBC18] BBC. *Drone light show in China breaks world record*. <https://www.bbc.com/news/av/technology-43981145>. Accessed: 2021-1-26. May 2018 (cit. on pp. 4, 157).

- [Bak+19] Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. “Emergent Tool Use From Multi-Agent Autocurricula”. In: (Sept. 2019). arXiv: [1909.07528 \[cs.LG\]](#) (cit. on p. 126).
- [BPZ19] D Baldazo, J Parras, and S Zazo. “Decentralized Multi-Agent Deep Reinforcement Learning in Swarms of Drones for Flood Monitoring”. In: *2019 27th European Signal Processing Conference (EUSIPCO)*. Sept. 2019, pp. 1–5 (cit. on p. 126).
- [Bar64] Paul Baran. *On Distributed Communications*. Tech. rep. RM-3420-PR. Rand Corporation, Aug. 1964 (cit. on p. 27).
- [Bar11] Maria Barbarossa. *Stability of discrete dynamical systems*. 2011 (cit. on p. 60).
- [BAS13] Jan Carlo Barca and Y Ahmet Sekercioglu. “Swarm robotics reviewed”. In: *Robotica* 31.3 (May 2013), pp. 345–359 (cit. on p. 28).
- [BH01] C R Baugh and J Huang. *Traffic Model for IEEE 802.16. 3 TG3 MAC/PHY Simulations*. Tech. rep. IEEE 802.16 Broadband Wireless Access Working Group, 2001 (cit. on pp. 147, 148).
- [Bed08] Mark A Bedau. “Weak Emergence”. In: *Nous* 31 (June 2008), pp. 375–399 (cit. on pp. 4, 32, 58, 126).
- [BW93] Gerardo Beni and Jing Wang. “Swarm Intelligence in Cellular Robotic Systems”. en. In: *Robots and Biological Systems: Towards a New Bionics? NATO ASI Series Volume 102*. Springer, Berlin, Heidelberg, 1993, pp. 703–712 (cit. on p. 32).
- [Ben92] Jon Jouis Bentley. “Fast Algorithms for Geometric Traveling Salesman Problems”. In: *ORSA Journal on Computing* 4.4 (Nov. 1992), pp. 387–411 (cit. on p. 18).
- [BT+14] Oded Berger-Tal, Jonathan Nathan, Ehud Meron, and David Saltz. “The Exploration-Exploitation Dilemma: A Multidisciplinary Framework”. In: *PLoS One* 9.4 (2014), e95693 (cit. on p. 2).
- [BTI11] D Bhadauria, O Tekdas, and V Isler. “Robotic data mules for collecting data over sparse sensor fields”. In: *Journal of Field Robotics* (2011) (cit. on pp. 19, 26).
- [BGSC20] Sushrut Bhalla, Sriram Ganapathi Subramanian, and Mark Crowley. “Deep Multi Agent Reinforcement Learning for Autonomous Driving”. In: *Advances in Artificial Intelligence*. Springer International Publishing, 2020, pp. 67–78 (cit. on p. 126).
- [Bie+93] Daniel Bienstock, Michel X Goemans, David Simchi-Levi, and David Williamson. “A note on the prize collecting traveling salesman problem”. en. In: *Math. Program.* 59.1-3 (Mar. 1993), pp. 413–420 (cit. on p. 21).
- [BMP18] Daniel Bienstock, Gonzalo Muñoz, and Sebastian Pokutta. “Principled Deep Neural Network Training through Linear Programming”. In: (Oct. 2018). arXiv: [1810.03218 \[cs.LG\]](#) (cit. on p. 126).
- [BTAZ06] Muhammad Mukarram Bin Tariq, Mostafa Ammar, and Ellen Zegura. “Message Ferry Route Design for Sparse Ad Hoc Networks with Mobile Nodes”. In: *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. MobiHoc ’06. Florence, Italy: ACM, 2006, pp. 37–48 (cit. on pp. 19, 26).
- [BDT99] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. en. OUP USA, Oct. 1999, pp. 187–219 (cit. on pp. 2, 36, 126, 143).
- [Bon+16] Marco Bonola, Lorenzo Bracciale, Pierpaolo Loreti, Raul Amici, Antonello Rabuffi, and Giuseppe Bianchi. “Opportunistic communication in smart city: Experimental insight with small-scale taxi fleets as data carriers”. en. In: *Ad Hoc Networks* 43.SI (June 2016), pp. 43–55 (cit. on pp. 13, 14).
- [Bor20] Stefan Borg. “Below the radar. Examining a small state’s usage of tactical unmanned aerial vehicles”. In: *Defence Studies* 20.3 (July 2020), pp. 185–201 (cit. on p. 1).
- [Bos+05] Fabio Boschetti, Mikhail Prokopenko, Ian Macreadie, and Anne-Marie Grisogono. “Defining and Detecting Emergence in Complex Networks”. en. In: *Knowledge-Based Intelligent Information and Engineering Systems*. Springer, Berlin, Heidelberg, 2005, pp. 573–580 (cit. on p. 32).
- [Box+15] George E P Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time Series Analysis: Forecasting and Control*. en. John Wiley & Sons, 2015 (cit. on pp. 146, 153).
- [BS15] T R Bretschneider and K Shetti. “UAV-based gas pipeline leak detection”. In: *Proc. of ARCS* (2015) (cit. on p. 1).
- [BE16] Mariana Madruga de Brito and Mariele Evers. “Multi-criteria decision-making for flood risk management: a survey of the current state of the art”. en. In: *Nat. Hazards Earth Syst. Sci.* 16.4 (Apr. 2016), pp. 1019–1033 (cit. on p. 49).

- [BW19] Marc Brittain and Peng Wei. “Autonomous Air Traffic Controller: A Deep Multi-Agent Reinforcement Learning Approach”. In: (May 2019). arXiv: [1905.01303 \[cs.LG\]](#) (cit. on p. 157).
- [Bro+16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. “OpenAI Gym”. In: (June 2016). arXiv: [1606.01540 \[cs.LG\]](#) (cit. on p. 42).
- [BP18] Christian Bunse and Sebastian Plotz. “Security Analysis of Drone Communication Protocols”. In: *Engineering Secure Software and Systems*. Springer International Publishing, 2018, pp. 96–107 (cit. on p. 1).
- [Bur+06] J Burgess, B Gallagher, D Jensen, and B N Levine. “MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks”. In: *INFOCOM* (2006) (cit. on p. 12).
- [CP88] Nicholas W Calderone and Robert E Page. “Genotypic variability in age polyethism and task specialization in the honey bee, *Apis mellifera* (Hymenoptera: Apidae)”. In: *Behav. Ecol. Sociobiol.* 22.1 (Jan. 1988), pp. 17–25 (cit. on p. 123).
- [CW11] Adam Campbell and Annie S Wu. “Multi-agent Role Allocation: Issues, Approaches, and Multiple Perspectives”. In: *Auton. Agent. Multi. Agent. Syst.* 22.2 (Mar. 2011), pp. 317–355 (cit. on pp. 31, 129).
- [Cam+18] B Campbell, A Perry, R Hunjet, G Wang, and B Northcote. “Minimising RF Detectability for Low Probability of Detection Communication”. In: *2018 Military Communications and Information Systems Conference (MilCIS)*. Nov. 2018, pp. 1–6 (cit. on p. 52).
- [CWF14] A J Carfang, N Wagle, and E W Frew. “Improving data ferrying by iteratively learning the radio frequency environment”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. [ieeexplore.ieee.org](#), 2014, pp. 1182–1188 (cit. on pp. 21, 26).
- [Cha11] W K V Chan. “Interaction metric of emergent behaviors in agent-based simulation”. In: *Proceedings of the 2011 Winter Simulation Conference (WSC)*. Dec. 2011, pp. 357–368 (cit. on pp. 6, 33, 97, 145, 146).
- [Cha+11] C Y Chang, C Y Lin, C Y Hsieh, and Y J Ho. “Patrolling Mechanisms for Disconnected Targets in Wireless Mobile Data Mules Networks”. In: *2011 International Conference on Parallel Processing*. 2011, pp. 93–98 (cit. on pp. 20, 26, 35, 80).
- [Cha+14] C Y Chang, G J Yu, T L Wang, and C Y Lin. “Path Construction and Visit Scheduling for Targets by Using Data Mules”. In: *IEEE Trans. Syst. Man Cybern.* 44.10 (Oct. 2014), pp. 1289–1300 (cit. on pp. 20, 26, 35).
- [CJ12] Jyh-Huei Chang and Rong-Hong Jan. “Message Ferry Routing Algorithm for Data Collection in Partitioned and Buffer-Limited Wireless Sensor Networks”. en. In: *J. Inf. Sci. Eng.* 28.4 (July 2012), pp. 655–669 (cit. on pp. 20, 26).
- [Che19] Gang Chen. “A New Framework for Multi-Agent Reinforcement Learning – Centralized Training and Exploration with Decentralized Execution via Policy Distillation”. In: (Oct. 2019). arXiv: [1910.09152 \[cs.LG\]](#) (cit. on p. 126).
- [CTL20] Xi Chen, Jun Tang, and Songyang Lao. “Review of Unmanned Aerial Vehicle Swarm Communication Architectures and Routing Protocols”. en. In: *NATO Adv. Sci. Inst. Ser. E Appl. Sci.* 10.10 (May 2020), p. 3661 (cit. on p. 1).
- [Che+07] C M Cheng, P H Hsiao, H T Kung, and D Vlah. “Maximizing Throughput of UAV-Relaying Networks with the Load-Carry-and-Deliver Paradigm”. In: *2007 IEEE Wireless Communications and Networking Conference*. Mar. 2007, pp. 4417–4424 (cit. on p. 14).
- [CY07] M C Chuah and P Yang. “Performance Evaluations of Various Message Ferry Scheduling Schemes with Two Traffic Classes”. In: *2007 4th IEEE Consumer Communications and Networking Conference*. Jan. 2007, pp. 227–233 (cit. on pp. 23, 76).
- [CY05] Mooi Choo Chuah and Pena Yang. “A message ferrying scheme with differentiated services”. en. In: *Milcom 2005 - 2005 Ieee Military Communications Conference, Vols 1-5*. IEEE Military Communications Conference. 345 E 47th St, New York, Ny 10017 USA: Ieee, 2005, pp. 1521–1527 (cit. on pp. 23, 26, 76).
- [Chu+18] Soon-Jo Chung, Aditya Avinash Paranjape, Philip Dames, Shaojie Shen, and Vijay Kumar. “A survey on aerial swarm robotics”. In: *IEEE Trans. Robot.* 34.4 (Aug. 2018), pp. 837–855 (cit. on p. 2).
- [Chu+16] T H Chung, M R Clement, M A Day, K D Jones, D Davis, and M Jones. “Live-fly, Large-Scale Field Experimentation for Large Numbers of Fixed-Wing UAVs”. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. May 2016, pp. 1255–1262 (cit. on pp. 4, 29).
- [CD01] E von Collani and Klaus Dräger. *Binomial Distribution Handbook for Scientists and Engineers*. en. Springer Science & Business Media, June 2001 (cit. on p. 42).
- [Coo15] William Cook. *Concorde TSP Solver*. <http://www.math.uwaterloo.ca/tsp/concorde.html>. Accessed: 2018-8-16. Mar. 2015 (cit. on p. 80).

- [Coy18] A Coyle. “Using Directional Antenna in UAVs to Enhance Tactical Communications”. In: *2018 Military Communications and Information Systems Conference (MilCIS)*. ieeexplore.ieee.org, Nov. 2018, pp. 1–6 (cit. on p. 27).
- [CH12] Andrew T Crooks and Alison J Heppenstall. “Introduction to Agent-Based Modelling”. In: *Agent-Based Models of Geographical Systems*. Ed. by Alison J Heppenstall, Andrew T Crooks, Linda M See, and Michael Batty. Dordrecht: Springer Netherlands, 2012, pp. 85–105 (cit. on p. 43).
- [DNP04] Shirshanka Das, Alok Nandan, and Giovanni Pau. “SPAWN: A Swarming Protocol for Vehicular Ad-hoc Wireless Networks”. In: *Proceedings of the 1st ACM International Workshop on Vehicular Ad Hoc Networks. VANET '04*. New York, NY, USA: ACM, 2004, pp. 93–94 (cit. on p. 13).
- [Dav+16] D T Davis, T H Chung, M R Clement, and M A Day. “Consensus-based data sharing for large-scale aerial swarm coordination in lossy communications environments”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ieeexplore.ieee.org, Oct. 2016, pp. 3801–3808 (cit. on p. 29).
- [DFL01] J A Davis, A H Fagg, and B N Levine. “Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks”. In: *Proceedings Fifth International Symposium on Wearable Computers*. 2001, pp. 141–148 (cit. on p. 13).
- [DG16] Carl P Dettmann and Orestis Georgiou. “Random geometric graphs with general connection functions”. en. In: *Phys Rev E* 93.3 (Mar. 2016), p. 032313 (cit. on p. 43).
- [DFDA11] Mario Di Francesco, Sajal K Das, and Giuseppe Anastasi. “Data Collection in Wireless Sensor Networks with Mobile Elements: A Survey”. In: *ACM Trans. Sen. Netw.* 8.1 (Aug. 2011), 7:1–7:31 (cit. on p. 14).
- [Dia+06] M B Dias, R Zlot, N Kalra, and A Stentz. “Market-Based Multirobot Coordination: A Survey and Analysis”. In: *Proc. IEEE* 94.7 (July 2006), pp. 1257–1270 (cit. on p. 31).
- [DO59] Edsger W Dijkstra and Others. “A note on two problems in connexion with graphs”. In: *Numer. Math.* 1.1 (1959), pp. 269–271 (cit. on p. 42).
- [DBS06] M Dorigo, M Birattari, and T Stutzle. “Ant colony optimization”. In: *IEEE Comput. Intell. Mag.* 1.4 (Nov. 2006), pp. 28–39 (cit. on pp. 2, 29).
- [DBT00] Marco Dorigo, Eric Bonabeau, and Guy Theraulaz. “Ant algorithms and stigmergy”. In: *Future Gener. Comput. Syst.* 16.8 (June 2000), pp. 851–871 (cit. on pp. 2, 24, 29, 52, 78, 112, 126, 143).
- [DL14] Haibin Duan and Pei Li. *Bio-inspired Computation in Unmanned Aerial Vehicles*. Springer, Berlin, Heidelberg, 2014 (cit. on p. 112).
- [DM03] Adrian Dumitrescu and Joseph S B Mitchell. “Approximation Algorithms for TSP with Neighborhoods in the Plane”. In: *J. Algorithms* 48.1 (Aug. 2003), pp. 135–159 (cit. on pp. 18, 80).
- [Ela96] Saber N Elaydi. *An Introduction to Difference Equations*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1996 (cit. on p. 60).
- [Elb19] Elbit Systems. *Hermes™ 900 - Elbit Systems*. <https://elbitsystems.com/product/hermes-900-5/>. Accessed: 2019-12-12. 2019 (cit. on p. 111).
- [ES16] M Elliot and T Stevens. “Dynamic range extension using HARLEQUIN and HAIL”. In: *MILCOM 2016 - 2016 IEEE Military Communications Conference*. ieeexplore.ieee.org, Nov. 2016, pp. 835–841 (cit. on pp. 8, 64, 68).
- [Erd+17] M Erdelj, E Natalizio, K R Chowdhury, and I F Akyildiz. “Help from the Sky: Leveraging UAVs for Disaster Management”. In: *IEEE Pervasive Comput.* 16.1 (Jan. 2017), pp. 24–32 (cit. on p. 1).
- [Ewe+13] W Ewert, R J Marks, B B Thompson, and A Yu. “Evolutionary Inversion of Swarm Emergence Using Disjunctive Combs Control”. In: *IEEE Trans. Syst. Man Cybern.* 43.5 (2013), pp. 1063–1076 (cit. on p. 161).
- [Fal03] Kevin Fall. “A Delay-tolerant Network Architecture for Challenged Internets”. In: *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. SIGCOMM '03*. Karlsruhe, Germany: ACM, 2003, pp. 27–34 (cit. on pp. 11, 116).
- [Far+05] Alessandro Farinelli, Luca Iocchi, Daniele Nardi, and Fabio Patrizi. “Task Assignment with Dynamic Token Generation”. In: *Monitoring, Security, and Rescue Techniques in Multiagent Systems*. Ed. by Barbara Dunin-Keplicz, Andrzej Jankowski, Andrzej Skowron, and Marcin Szczuka. Springer Berlin Heidelberg, 2005, pp. 467–477 (cit. on p. 31).
- [Faw+17] W Fawaz, R Atallah, C Assi, and M Khabbaz. “Unmanned Aerial Vehicles as Store-Carry-Forward Nodes for Vehicular Networks”. In: *IEEE Access* 5 (2017), pp. 23710–23718 (cit. on p. 14).
- [Fed20] Federal Aviation Administration. *FAA Aerospace Forecast Fiscal Years 2020-2040*. Tech. rep. TC20-0011. Federal Aviation Administration, 2020 (cit. on p. 1).

- [FBB07] Paulo R Ferreira Jr., Felipe S Boffo, and Ana L C Bazzan. “A swarm based approximated algorithm to the extended generalized assignment problem (E-GAP)”. In: *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems - AAMAS '07*. Honolulu, Hawaii: ACM Press, 2007, p. 1 (cit. on p. 32).
- [Fer+10] Paulo Roberto Ferreira, Fernando dos Santos, Ana L C Bazzan, Daniel Epstein, and Samuel J Waskow. “RoboCup Rescue as multiagent task allocation among teams: experiments with task interdependencies”. In: *Auton. Agent. Multi. Agent. Syst.* 20.3 (May 2010), pp. 421–443 (cit. on p. 32).
- [FM08] Dario Floreano and Claudio Mattiussi. *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. en. MIT Press, 2008 (cit. on p. 143).
- [Fra18a] **Bradley Fraser**. “Data Ferry Flocking for Bulk Information Transfer Under Ferry Buffer Constraints”. In: *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*. Nov. 2018, pp. 1–6 (cit. on pp. 4, 7, 29, 75, 126).
- [Fra18b] **Bradley Fraser**. “Radio-Frequency Emitter Localisation Using a Swarm of Search Agents”. In: *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*. Nov. 2018, pp. 1–6 (cit. on pp. 4, 126, 159).
- [Fra+20a] **Bradley Fraser**, Andrew Coyle, Robert Hunjet, and Claudia Szabo. “An Analytic Latency Model for a Next-Hop Data-Ferrying Swarm on Random Geometric Graphs”. In: *IEEE Access* 8 (2020), pp. 48929–48942 (cit. on pp. 7, 75, 123).
- [Fra+19] **Bradley Fraser**, Andrew Coyle, Claudia Szabo, and Robert Hunjet. “Adaptive Swarm Control for Mobile Resource Placement in Wireless Ad-Hoc Networks”. In: *2019 IEEE 20th International Symposium on “A World of Wireless, Mobile and Multimedia Networks” (WoWMoM)*. June 2019, pp. 1–6 (cit. on pp. 7, 75, 123).
- [FH16] **Bradley Fraser** and Robert Hunjet. “Data ferrying in tactical networks using swarm intelligence and stigmergic coordination”. In: *2016 26th International Telecommunication Networks and Applications Conference (ITNAC)*. Dunedin, New Zealand, Dec. 2016, pp. 1–6 (cit. on pp. 6, 51).
- [FHC17] **Bradley Fraser**, Robert Hunjet, and Andrew Coyle. “A swarm intelligent approach to data ferrying in sparse disconnected networks”. In: *Int. J. Parallel Emergent Distrib. Syst.* (Oct. 2017), pp. 1–22 (cit. on pp. 7, 51, 80).
- [FL20] **Bradley Fraser** and Giuseppe Laurito. “Train Small, Deploy Big: Do Relative World Views Permit Swarm-Safety During Policy Transplantation for Multi-Agent Reinforcement Learning Problems?”. In: *AI 2020: Advances in Artificial Intelligence*. Springer International Publishing, 2020, pp. 269–280 (cit. on pp. 7, 125).
- [FMR20] **Bradley Fraser**, Philip Moss, and Kent Rosser. “4DNav: A Hybrid Dynamically-Planned Optimisation-based Approach to Mission Planning for Spatio-Temporal UAS Rendezvous”. In: *Australasian Conference on Robotics and Automation*. 2020 (cit. on p. 136).
- [Fra+20b] **Bradley Fraser**, Claudia Szabo, Andrew Coyle, and Robert Hunjet. “Leveraging emergent specialization in a heterogeneous multi-role swarm control architecture for positional-based UAS missions”. In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. Canberra, Australia: IEEE, Dec. 2020 (cit. on pp. 7, 111).
- [FSH17] **Bradley Fraser**, Claudia Szabo, and Robert Hunjet. “Simulating the effect of degraded wireless communications on emergent behavior”. In: *2017 Winter Simulation Conference (WSC)*. Dec. 2017, pp. 4081–4092 (cit. on pp. 7, 29, 97, 143).
- [FPP07] David Freedman, Robert Pisani, and Roger Purves. *Statistics: Fourth International Student Edition*. en. W.W. Norton & Company, Feb. 2007 (cit. on p. 96).
- [GCT15] N Ganganath, C T Cheng, and C K Tse. “Distributed Anti-Flocking Control for Mobile Surveillance Systems”. In: *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*. May 2015, pp. 1726–1729 (cit. on p. 143).
- [GZD09] S Gao, H Zhang, and S Das. “Efficient data collection in wireless sensor networks with path-constrained mobile sinks”. In: *2009 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks Workshops*. June 2009, pp. 1–9 (cit. on pp. 16, 17, 26).
- [Gao+16] X Gao, X Zhu, Y Feng, F Wu, and G Chen. “Data Ferry Trajectory Planning for Sweep Coverage Problem with Multiple Mobile Sensors”. In: *2016 13th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. June 2016, pp. 1–9 (cit. on pp. 14, 20).
- [GGT07] Simon Garnier, Jacques Gautrais, and Guy Theraulaz. “The biological principles of swarm intelligence”. In: *Swarm Intelligence* 1.1 (June 2007), pp. 3–31 (cit. on pp. 85, 112).

- [Gau+03] Paolo Gaudiano, Benjamin Shargel, Eric Bonabeau, and Bruce T Clough. *Swarm intelligence: A new C2 paradigm with an application to control swarms of UAVs*. Tech. rep. ICOSYSTEM CORP CAMBRIDGE MA, 2003 (cit. on p. 29).
- [GFC13] J A Gay-Fernández and I Cuiñas. “Peer to Peer Wireless Propagation Measurements and Path-Loss Modeling in Vegetated Environments”. In: *IEEE Transactions on Antennas and Propagation* 61.6 (June 2013), pp. 3302–3311 (cit. on p. 147).
- [Gen+18] Mingyang Geng, Xing Zhou, Bo Ding, Huaimin Wang, and Lei Zhang. “Learning to Cooperate in Decentralized Multi-robot Exploration of Dynamic Environments: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13–16, 2018, Proceedings, Part VII”. In: *Neural Information Processing*. Ed. by Long Cheng, Andrew Chi Sing Leung, and Seiichi Ozawa. Vol. 11307. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 40–51 (cit. on p. 126).
- [GPM19] James B Gibson, John Page, and Faqihza Mukhlis. “Simulation of an Unmanned Aerial Vehicle Search and Rescue Swarm for Observation of Emergent Behaviour”. In: *Intersections in Simulation and Gaming: Disruption and Balance*. Springer Singapore, 2019, pp. 95–105 (cit. on p. 1).
- [Gil60] E N Gilbert. “Capacity of a burst-noise channel”. In: *The Bell System Technical Journal* 39.5 (1960), pp. 1253–1265 (cit. on p. 42).
- [Gil61] E Gilbert. “Random Plane Networks”. In: *Journal of the Society for Industrial and Applied Mathematics* 9.4 (Dec. 1961), pp. 533–543 (cit. on p. 43).
- [GCT18] Christopher K Gilmore, Michael Chaykowsky, and Brent Thomas. *Autonomous Unmanned Aerial Vehicles for Blood Delivery: A UAV Fleet Design Tool and Case Study*. Tech. rep. RR3047. RAND Corporation, 2018 (cit. on p. 111).
- [Gol16] Goldman Sachs. *Drones: Reporting for Work*. <https://www.goldmansachs.com/insights/technology-driving-innovation/drones/>. Accessed: 2021-1-25. 2016 (cit. on p. 1).
- [Goo20] Google Developers. *Protocol Buffers* | Google Developers. <https://developers.google.com/protocol-buffers>. Accessed: 2020-7-21. 2020 (cit. on p. 137).
- [Gos+89] S Goss, S Aron, J L Deneubourg, and J M Pasteels. “Self-organized shortcuts in the Argentine ant”. en: *Naturwissenschaften* 76.12 (Dec. 1989), pp. 579–581 (cit. on p. 29).
- [GLQ08] Hui Guo, Jiang Li, and Yi Qian. “HoP: Pigeon-Assisted Forwarding in Partitioned Wireless Networks”. In: *Proceedings of the Third International Conference on Wireless Algorithms, Systems, and Applications*. WASA '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 72–83 (cit. on pp. 14, 22, 26).
- [Guo+06] Shimin Guo, Majid Ghaderi, Aaditeshwar Seth, and Srinivasan Keshav. “Opportunistic scheduling in ferry based networks”. In: *Proc. WNEPT* (2006) (cit. on p. 13).
- [GK07] Shimin Guo and Srinivasan Keshav. “Fair and Efficient Scheduling in Data Ferrying Networks”. In: *International Conference On Emerging Networking Experiments And Technologies* (2007), pp. 1–12 (cit. on p. 13).
- [GEK17] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. “Cooperative Multi-agent Control Using Deep Reinforcement Learning”. In: *Autonomous Agents and Multiagent Systems*. Springer International Publishing, 2017, pp. 66–83 (cit. on p. 126).
- [HSS19] Aric Hagberg, Dan Schult, and Pieter Swart. *NetworkX — NetworkX*. <https://networkx.github.io/>. Accessed: 2019-12-12. 2019 (cit. on pp. 42, 117).
- [HS18] R N Haksar and M Schwager. “Distributed Deep Reinforcement Learning for Fighting Forest Fires with a Network of Aerial Robots”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2018, pp. 1067–1074 (cit. on pp. 1, 126).
- [Ham20] David Hambling. *China Releases Video Of New Barrage Swarm Drone Launcher*. <https://www.forbes.com/sites/davidhambling/2020/10/14/china-releases-video-of-new-barrage-swarm-drone-launcher/?sh=1218d4e42ad7>. Accessed: 2021-1-26. Oct. 2020 (cit. on p. 4).
- [HW16] Hamdani and Retantyo Wardoyo. “The complexity calculation for group decision making using TOPSIS algorithm”. In: *AIP Conf. Proc.* 1755.1 (July 2016), p. 070007 (cit. on p. 115).
- [HO20a] Thomas Hamilton and David Ochmanek. *Operating Low-Cost, Reusable Unmanned Aerial Vehicles in Contested Environments: Preliminary Evaluation of Operational Concepts*. en. RAND Corporation, May 2020 (cit. on p. 1).
- [HO20b] Thomas Hamilton and David Ochmanek. *Operating Low-Cost, Reusable Unmanned Aerial Vehicles in Contested Environments: Preliminary Evaluation of Operational Concepts*. Tech. rep. RR-4407-AF. RAND Corporation, 2020 (cit. on p. 3).

- [HN08] Samer Hanoun and Saeid Nahavandi. “Dynamic Route Construction for Mobile Collectors in Wireless Sensor Networks”. en. In: *Intelligent Robotics and Applications*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, Oct. 2008, pp. 824–833 (cit. on p. 14).
- [HMT18a] M Harounabadi and A Mitschele-Thiel. “Trajectory and Buffer Aware Message Forwarding for Multiple Cooperating UAVs in Message Ferry Networks”. en. In: *Ad Hoc Networks*. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 223 (2018). Ed. by Yifeng Zhou and Thomas Kunz, pp. 156–165 (cit. on p. 24).
- [HMT17] Mehdi Harounabadi and Andreas Mitschele-Thiel. “Stigmergic Communication for Self-Organized Multi Ferry Delay Tolerant Networks”. en. In: *Mobile Netw Appl* (Jan. 2017), pp. 1–10 (cit. on pp. 24, 26).
- [HMT18b] Mehdi Harounabadi and Andreas Mitschele-Thiel. “Applying Message Forwarding and Replication to Multi-UAV Message Ferry Networks”. In: *Mobile Networks and Applications* (Mar. 2018) (cit. on p. 24).
- [HSL11] Ting He, Ananthram Swami, and Kang-Won Lee. “Dispatch-and-Search: Dynamic Multi-Ferry Control in Partitioned Mobile Networks”. en. In: *Proceedings of the Twelfth Acm International Symposium on Mobile Ad Hoc Networking and Computing (mobihoc’ 11)*. 1515 Broadway, New York, Ny 10036-9998 USA: Assoc Computing Machinery, 2011 (cit. on pp. 23, 26).
- [Heb49] D O Hebb. *The Organization of Behavior: A Neuropsychological Theory*. New York: Wiley, 1949 (cit. on p. 47).
- [HA14] T Heimfarth and J P Araujo. “Using Unmanned Aerial Vehicle to Connect Disjoint Segments of Wireless Sensor Network”. In: *2014 IEEE 28th International Conference on Advanced Information Networking and Applications*. May 2014, pp. 907–914 (cit. on pp. 19, 26).
- [HAG14] T Heimfarth, J P Araujo, and J C Giacomini. “Unmanned Aerial Vehicle as Data Mule for Connecting Disjoint Segments of Wireless Sensor Network with Unbalanced Traffic”. In: *2014 IEEE 17th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing*. June 2014, pp. 246–252 (cit. on pp. 22, 26).
- [HB08] D Henkel and T X Brown. “Towards autonomous data ferry route design through reinforcement learning”. In: *2008 International Symposium on a World of Wireless, Mobile and Multimedia Networks*. ieeexplore.ieee.org, June 2008, pp. 1–6 (cit. on pp. 21, 26).
- [HB06] Daniel Henkel and Timothy X Brown. “On Controlled Node Mobility in DelayTolerant Networks of Unmanned Aerial Vehicles”. In: *International Symposium on Advance Radio Technologies (ISART)*. 2006 (cit. on pp. 54, 85, 99, 100).
- [Hie18] Andrew Hietpas. *Stigmergic Control of Dual Direction Communication Ferry Nodes for Denied Communication Environments*. Tech. rep. Naval Postgraduate School Monterey United States, 2018 (cit. on pp. 8, 158).
- [HSD04] M W Hirsch, S Smale, and R L Devaney. *Differential Equations, Dynamical Systems, and an Introduction to Chaos*. Differential equations, dynamical systems, and an introduction to chaos. Academic Press, 2004 (cit. on p. 60).
- [HN17] Amy Hocraffer and Chang S Nam. “A meta-analysis of human-system interfaces in unmanned aerial vehicle (UAV) swarm management”. en. In: *Appl. Ergon.* 58 (Jan. 2017), pp. 66–80 (cit. on p. 2).
- [Hol07] O Thomas Holland. “Taxonomy for the Modeling and Simulation of Emergent Behavior Systems”. In: *Proceedings of the 2007 Spring Simulation Multiconference - Volume 2*. SpringSim ’07. Norfolk, Virginia: Society for Computer Simulation International, 2007, pp. 28–35 (cit. on p. 86).
- [Hsu+20] Christopher D Hsu, Heejin Jeong, George J Pappas, and Pratik Chaudhari. “Scalable Reinforcement Learning Policies for Multi-Agent Control”. In: (Nov. 2020). arXiv: 2011.08055 [cs.MA] (cit. on p. 33).
- [Hui+11] K P Hui, P Pourbeik, P George, D Phillips, S Magrath, and M Kwiatkowski. “OPAL - A survivability-oriented approach to management of tactical military networks”. In: *2011 - MILCOM 2011 Military Communications Conference*. Nov. 2011, pp. 1127–1132 (cit. on pp. 1, 27).
- [HPK17] K Hui, D Phillips, and A Kekirigoda. “Beyond Line-of-Sight range extension with OPAL using autonomous unmanned aerial vehicles”. In: *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*. Oct. 2017, pp. 279–284 (cit. on p. 1).
- [Hun15a] R Hunjet. “Autonomy and self-organisation for tactical communications and range extension”. In: *2015 Military Communications and Information Systems Conference (MilCIS)*. Nov. 2015, pp. 1–5 (cit. on pp. 4, 126).
- [Hun15b] R Hunjet. “Autonomy and self-organisation for tactical communications and range extension”. In: *2015 Military Communications and Information Systems Conference (MilCIS)*. Nov. 2015, pp. 1–5 (cit. on pp. 27, 29, 143, 144).
- [HH11] R Hunjet and P Hui. “Maintaining connectivity in mobile adhoc networks using distributed optimisation”. In: *2011 Military Communications and Information Systems Conference*. Nov. 2011, pp. 1–6 (cit. on p. 1).

- [Hun+18] Robert Hunjet, **Bradley Fraser**, Thomas Stevens, Laura Hodges, Karina Mayen, Jan Carlo Barca, Madeleine Cochrane, Ricardo Cannizzaro, and Jennifer L Palmer. “**Data Ferrying with Swarming UAS in Tactical Defence Networks**”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Australia: IEEE, May 2018, pp. 6381–6388 (cit. on pp. 7, 8, 51, 163).
- [Hun+17] Robert Hunjet, Thomas Stevens, Mathew Elliot, **Bradley Fraser**, and Peter George. “**Survivable communications and autonomous delivery service a generic swarming framework enabling communications in contested environments**”. In: *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*. Baltimore, USA, Oct. 2017, pp. 788–793 (cit. on p. 26).
- [Hus+17] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. “Imitation Learning: A Survey of Learning Methods”. In: *ACM Comput. Surv.* 50.2 (Apr. 2017), pp. 1–35 (cit. on p. 141).
- [HŠN18] Maximilian Hüttenrauch, Adrian Šošić, and Gerhard Neumann. “Local Communication Protocols for Learning Complex Swarm Behaviors with Deep Reinforcement Learning”. In: *International Conference on Swarm Intelligence*. computational-learning.net, 2018 (cit. on pp. 33, 130, 131).
- [HŠN19] Maximilian Hüttenrauch, Adrian Šošić, and Gerhard Neumann. “Deep Reinforcement Learning for Swarm Systems”. In: *J. Mach. Learn. Res.* 20.54 (2019), pp. 1–31 (cit. on p. 126).
- [Hwa17] Kai Hwang. *Cloud Computing for Machine Learning and Cognitive Applications*. en. MIT Press, June 2017 (cit. on p. 127).
- [HK06] Rob J Hyndman and Anne B Koehler. “Another look at measures of forecast accuracy”. In: *Int. J. Forecast.* 22.4 (Oct. 2006), pp. 679–688 (cit. on p. 95).
- [IEE89] IEEE. *IEEE Standard for Local Area Networks: Token Ring Access Method and Physical Layer Specifications*. Tech. rep. Dec. 1989, pp. 1–98 (cit. on p. 31).
- [IG18] Mauro Sebastián Innocente and Paolo Grasso. “Swarm of autonomous drones self-organised to fight the spread of wildfires”. en. In: *Proceedings of the GEOSAFE Workshop on Robust Solutions for Fire Fighting*. CEUR, July 2018 (cit. on p. 1).
- [Int] Intel Corp. *Drone Light Shows Powered by Intel*. <https://www.intel.com.au/content/www/au/en/technology-innovation/aerial-technology-light-show.html>. Accessed: 2021-1-26 (cit. on pp. 4, 157).
- [Isa+12] J T Isaacs, S Venkateswaran, J Hespanha, U Madhow, J Burman, and T Pham. “Multiple event localization in a sparse acoustic sensor network using UAVs as data mules”. In: *2012 IEEE Globecom Workshops*. Dec. 2012, pp. 1562–1567 (cit. on pp. 18, 26).
- [Jaw+13] I Jawhar, M Ammar, S Zhang, J Wu, and N Mohamed. “Ferry-based linear wireless sensor networks”. In: *2013 IEEE Global Communications Conference (GLOBECOM)*. Dec. 2013, pp. 304–309 (cit. on p. 17).
- [JSS05] David Jea, Arun Somasundara, and Mani Srivastava. “Multiple Controlled Mobile Elements (Data Mules) for Data Collection in Sensor Networks”. en. In: *Distributed Computing in Sensor Systems*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, June 2005, pp. 244–257 (cit. on pp. 16, 26).
- [Joh06] Christopher W Johnson. “What Are Emergent Properties and How Do They Affect the Engineering of Complex Systems?” In: *Reliable Engineering and System Safety*. 12 (2006), pp. 1475–1481 (cit. on p. 32).
- [Jua+02] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li Shiuan Peh, and Daniel Rubenstein. “Energy-efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet”. In: *SIGARCH Comput. Archit. News* 30.5 (Oct. 2002), pp. 96–107 (cit. on p. 13).
- [JK18a] Kyle D Julian and Mykel J Kochenderfer. “Autonomous Distributed Wildfire Surveillance using Deep Reinforcement Learning”. In: *2018 AIAA Guidance, Navigation, and Control Conference*. unknown, Jan. 2018 (cit. on p. 126).
- [JK18b] Kyle D Julian and Mykel J Kochenderfer. “Distributed Wildfire Surveillance with Autonomous Aircraft using Deep Reinforcement Learning”. In: (Oct. 2018). arXiv: 1810.04244 [cs.R0] (cit. on p. 1).
- [KY13] B Kalyanasundaram and M Younis. “Using mobile data collectors to federate clusters of disjoint sensor network segments”. In: *2013 IEEE International Conference on Communications (ICC)*. June 2013, pp. 1496–1500 (cit. on pp. 19, 26).
- [KKZ11] Shyam Kapadia, Bhaskar Krishnamachari, and Lin Zhang. “Data delivery in delay tolerant networks: A survey”. In: *Mobile Ad-Hoc Networks: Protocol Design*. InTech, 2011 (cit. on p. 14).
- [KB07] Dervis Karaboga and Bahriye Basturk. “A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm”. en. In: *Journal of Global Optimization* 39.3 (2007), pp. 459–471 (cit. on pp. 2, 143).

- [KS16] Marcin Kawecki and Radosaw Olgierd Schoeneich. “Mobility-based routing algorithm in delay tolerant networks”. en. In: *Eurasip J. Wirel. Commun. Network*. (2016), pp. 1–9 (cit. on p. 12).
- [KMC20] Jacqueline Kazil, David Masad, and Andrew Crooks. “Utilizing Python for Agent-Based Modeling: The Mesa Framework”. In: *2020 International Conference on Social Computing, Behavioral-Cultural Modeling & Prediction and Behavior Representation in Modeling and Simulation*. unknown, Oct. 2020 (cit. on p. 42).
- [Kim+14] D Kim, R N Uma, B H Abay, W Wu, W Wang, and A O Tokuta. “Minimum Latency Multiple Data MULE Trajectory Planning in Wireless Sensor Networks”. In: *IEEE Trans. Mob. Comput.* 13.4 (Apr. 2014), pp. 838–851 (cit. on pp. 19, 26).
- [Kim+16] Donghyun Kim, Wei Wang, Deying Li, Joong-Lyul Lee, Weili Wu, and Alade O Tokuta. “A joint optimization of data ferry trajectories and communication powers of ground sensors for long-term environmental monitoring”. en. In: *J. Comb. Optim.* 31.4 (May 2016), pp. 1550–1568 (cit. on pp. 19, 26).
- [Kin16] Katharine Hall Kindervater. “The emergence of lethal surveillance: Watching and killing in the history of drone technology”. In: *Security Dialogue* 47.3 (Jan. 2016) (cit. on p. 1).
- [KBJ13a] Abdullah Konak, George E Buchert, and James Juro. “A flocking-based approach to maintain connectivity in mobile wireless ad hoc networks”. In: *Appl. Soft Comput.* 13.2 (Feb. 2013), pp. 1284–1291 (cit. on pp. 4, 126).
- [KBJ13b] Abdullah Konak, George E Buchert, and James Juro. “A flocking-based approach to maintain connectivity in mobile wireless ad hoc networks”. In: *Appl. Soft Comput.* 13.2 (Feb. 2013), pp. 1284–1291 (cit. on pp. 27, 29, 143).
- [Kou+19] A Koubâa, A Allouch, M Alajlan, Y Javed, A Belghith, and M Khalgui. “Micro Air Vehicle Link (MAVlink) in a Nutshell: A Survey”. In: *IEEE Access* 7 (2019), pp. 87658–87680 (cit. on p. 136).
- [KWB01] B Krishnamachari, S B Wicker, and R Bejar. “Phase transition phenomena in wireless ad hoc networks”. In: *GLOBECOM’01. IEEE Global Telecommunications Conference (Cat. No.01CH37270)*. Vol. 5. Nov. 2001, 2921–2925 vol.5 (cit. on p. 98).
- [Kus+13] Alex Kushleyev, Daniel Mellinger, Caitlin Powers, and Vijay Kumar. “Towards a swarm of agile micro quadrotors”. en. In: *Auton. Robots* 35.4 (Nov. 2013), pp. 287–300 (cit. on p. 28).
- [LW08] Kun-Chan Lan and Ze Ming Wu. “On the feasibility of using public transport as data mules for traffic monitoring”. In: *2008 IEEE Intelligent Vehicles Symposium*. June 2008, pp. 979–984 (cit. on p. 13).
- [Lau15] Dylan Z Lau. “Investigation of coordination algorithms for swarm robotics conducting area search”. PhD thesis. Monterey, California: Naval Postgraduate School, 2015 (cit. on p. 28).
- [LFR20] G Laurito, B Fraser, and K Rosser. “Airborne Localisation of Small UAS using Visual Detection: A Field Experiment”. In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. Dec. 2020, pp. 1435–1443 (cit. on pp. 129, 142).
- [Leg+08] François Legras, Arnaud Glad, Olivier Simonin, and François Charpillet. “Authority Sharing in a Swarm of UAVs: Simulation and Experiments with Operators”. en. In: *Simulation, Modeling, and Programming for Autonomous Robots*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, Nov. 2008, pp. 293–304 (cit. on p. 28).
- [LES14] Liron Levin, Alon Efrat, and Michael Segal. “Collecting data in ad-hoc networks with reduced uncertainty”. In: *Ad Hoc Networks* 17 (June 2014), pp. 71–81 (cit. on pp. 21, 26).
- [LMAM02] Ling Li, Alcherio Martinoli, and Yaser S Abu-Mostafa. “Emergent Specialization in Swarm Systems”. In: *Intelligent Data Engineering and Automated Learning — IDEAL 2002*. Springer Berlin Heidelberg, 2002, pp. 261–266 (cit. on p. 123).
- [Li+17] Qiyang Li, Xintong Du, Yizhou Huang, Quinlan Sykora, and Angela P Schoellig. “Learning of Coordination Policies for Robotic Swarms”. In: (Sept. 2017). arXiv: 1709.06620 [cs.R0] (cit. on pp. 34, 128, 130).
- [LR00] Qun Li and Daniela Rus. “Sending Messages to Mobile Users in Disconnected Ad-hoc Wireless Networks”. In: *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*. MobiCom ’00. New York, NY, USA: ACM, 2000, pp. 44–55 (cit. on p. 14).
- [Li+11] Y Li, B Weng, Q Liu, L Tang, and M Daneshmand. “Multiple Ferry Routing for the Opportunistic Networks”. In: *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*. Dec. 2011, pp. 1–5 (cit. on pp. 18, 26).
- [LT19] Yanjiang Li and Chong Tan. “A survey of the consensus for multi-agent systems”. In: *Systems Science & Control Engineering* 7.1 (Jan. 2019), pp. 468–482 (cit. on p. 129).

- [LJ11] Zhi-Jun Li and Shou-Xu Jiang. “Planning the Mobility of Routing Ferries for Intermittently Connected Mobile Networks”. en. In: *2011 6th International Ict Conference on Communications and Networking in China (chinacom)*. 345 E 47th St, New York, Ny 10017 USA: Ieee, Aug. 2011, pp. 816–821 (cit. on pp. 14, 19).
- [Lin+12] M Lindeberg, J E Haavet, Sergio Cabrero Barros, V Goebel, and T Plagemann. “Message lost or message taken - on message ferry selection in DTNs -”. In: *2012 IFIP Wireless Days*. Nov. 2012, pp. 1–7 (cit. on p. 12).
- [LDS04] Anders Lindgren, Avri Doria, and Olov Schelén. “Probabilistic Routing in Intermittently Connected Networks”. en. In: *Service Assurance with Partial and Intermittent Resources*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2004, pp. 239–254 (cit. on p. 12).
- [Liu+10] C H Liu, T He, K w. Lee, K K Leung, and A Swami. “Dynamic Control of Data Ferries under Partial Observations”. In: *2010 IEEE Wireless Communication and Networking Conference*. ieeexplore.ieee.org, Apr. 2010, pp. 1–6 (cit. on pp. 23, 26).
- [LWC13] J S Liu, S Y Wu, and K M Chiu. “Path planning of a data mule in wireless sensor network using an improved implementation of clustering-based genetic algorithm”. In: *2013 IEEE Symposium on Computational Intelligence in Control and Automation (CICA)*. Apr. 2013, pp. 30–37 (cit. on pp. 18, 26).
- [Liu+15] M Liu, T Johnson, R Agarwal, A Efrat, A Richa, and M M Coutinho. “Robust data mule networks with remote healthcare applications in the Amazon region: A fountain code approach”. In: *2015 17th International Conference on E-health Networking, Application Services (HealthCom)*. Oct. 2015, pp. 546–551 (cit. on p. 13).
- [Liu+17] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. “A survey of deep neural network architectures and their applications”. In: *Neurocomputing* 234 (Apr. 2017), pp. 11–26 (cit. on p. 47).
- [Liu+20] Yuxuan Liu, Hu Liu, Yongliang Tian, and Cong Sun. “Reinforcement learning based two-level control framework of UAV swarm for cooperative persistent surveillance in an unknown urban area”. In: *Aerosp. Sci. Technol.* 98 (Mar. 2020), p. 105671 (cit. on p. 126).
- [Lon+17] Pinxin Long, Tingxiang Fan, Xinyi Liao, Wenxi Liu, Hao Zhang, and Jia Pan. “Towards Optimally Decentralized Multi-Robot Collision Avoidance via Deep Reinforcement Learning”. In: (Sept. 2017). [arXiv: 1709.10082 \[cs.R0\]](https://arxiv.org/abs/1709.10082) (cit. on p. 126).
- [Lot+17] P Lottes, R Khanna, J Pfeifer, R Siegwart, and C Stachniss. “UAV-based crop and weed classification for smart farming”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. ieeexplore.ieee.org, May 2017, pp. 3024–3031 (cit. on p. 1).
- [Low+17] Ryan Lowe, Y I Wu, Aviv Tamar, Jean Harb, Openai Pieter Abbeel, and Igor Mordatch. “Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments”. In: *Advances in Neural Information Processing Systems* 30. Ed. by I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett. Curran Associates, Inc., 2017, pp. 6379–6390 (cit. on pp. 48, 126, 128, 129, 131, 134, 138).
- [LSS13] D E Lucani, P B Sujit, and J B Sousa. “Spare the mule, help your neighbors: Robot route planning for data retrieval on large scale sensor networks”. In: *2013 IEEE International Conference on Robotics and Automation*. May 2013, pp. 2534–2541 (cit. on pp. 17, 26).
- [Luk+05] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, Keith Sullivan, and Gabriel Balan. “MASON: A Multiagent Simulation Environment”. In: *Simulation* 81.7 (July 2005), pp. 517–527 (cit. on pp. 42, 55).
- [Ma+11] Liang Ma, Ting He, Ananthram Swami, K Won Lee, and Kin K Leung. “Switch-and-navigate: Controlling data ferry mobility under bounded message delays”. In: *Proc. of the IEEE Military Communications Conference*. 2011 (cit. on pp. 23, 26).
- [MAZ11] A Mansy, M Ammar, and E Zegura. “Deficit Round-Robin Based Message Ferry Routing”. In: *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*. Dec. 2011, pp. 1–5 (cit. on pp. 24, 26).
- [Mar+11] X Marjou, A Braud, R Khanafer, N Omnes, and A Quenot. “Using hawkers to ferry audiovisual data”. In: *2011 The 14th International Symposium on Wireless Personal Multimedia Communications (WPMC)*. Oct. 2011, pp. 1–5 (cit. on pp. 13, 14).
- [MV+16] Ruben Martinez-Vidal, Ramon Marti, Cormac J Sreenan, and Joan Borrell. “Measuring QoS in an Aeronautical Opportunistic Network Architecture with Limited Access to a Satellite Communications Backhaul”. In: *Mobile Information Systems* 2016 (2016), 7601316:1–12 (cit. on p. 147).
- [MA03] E Martinson and R C Arkin. “Learning to role-switch in multi-robot systems”. In: *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*. Vol. 2. Sept. 2003, 2727–2734 vol.2 (cit. on p. 32).
- [MK15] David Masad and Jacqueline Kazil. “Mesa: An Agent-Based Modeling Framework”. In: *Python in Science Conference*. unknown, Jan. 2015, pp. 51–58 (cit. on p. 42).

- [MK21] Mohammad Masdari and Hemn Khezri. “Service selection using fuzzy multi-criteria decision making: a comprehensive review”. en. In: *J. Ambient Intell. Humaniz. Comput.* 12.2 (Feb. 2021), pp. 2803–2834 (cit. on p. 49).
- [Mas51] Frank J Massey. “The Kolmogorov-Smirnov Test for Goodness of Fit”. In: *J. Am. Stat. Assoc.* 46.253 (Mar. 1951), pp. 68–78 (cit. on p. 95).
- [MWH01] M Mauve, J Widmer, and H Hartenstein. “A survey on position-based routing in mobile ad hoc networks”. In: *IEEE Netw.* 15.6 (Nov. 2001), pp. 30–39 (cit. on p. 77).
- [MLY17] Michalis Mavrovouniotis, Changhe Li, and Shengxiang Yang. “A survey of swarm intelligence for dynamic optimization: Algorithms and applications”. In: *Swarm and Evolutionary Computation* 33 (2017), pp. 1–17 (cit. on p. 2).
- [MP43] Warren S McCulloch and Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. In: *Bull. Math. Biophys.* 5.4 (Dec. 1943), pp. 115–133 (cit. on p. 47).
- [Mei+13] L Meier, J Camacho, B Godbolt, J Goppert, L Heng, and others. “Mavlink: Micro air vehicle communication protocol”. In: *[Online]* (2013) (cit. on p. 136).
- [MKK10] Y Q Miao, A Khamis, and M S Kamel. “Applying Anti-Flocking Model in Mobile Surveillance Systems”. In: *2010 International Conference on Autonomous and Intelligent Systems, AIS 2010*. June 2010, pp. 1–6 (cit. on p. 143).
- [Mic+18] Matias Micheletto, Vinicius Petrucci, Rodrigo Santos, Javier Orozco, Daniel Mosse, Sergio F Ochoa, and Roc Meseguer. “Flying Real-Time Network to Coordinate Disaster Relief Activities in Urban Areas”. en. In: *Sensors* 18.5 (May 2018), p. 1662 (cit. on p. 1).
- [ML16] Eliza J T Middleton and Tanya Latty. “Resilience in social insect infrastructure systems”. en. In: *J. R. Soc. Interface* 13.116 (Mar. 2016) (cit. on p. 112).
- [MML14] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. “Grey Wolf Optimizer”. In: *Adv. Eng. Softw.* 69 (2014), pp. 46–61 (cit. on p. 2).
- [MCA05] Ranjeev Mittu, William Chao, and Myriam Abramson. “Design and Evaluation of Distributed Role Allocation Algorithms in Open Environments”. In: *Proceedings of the 2005 International Conference on Artificial Intelligence, ICAI 2005, Las Vegas, Nevada, USA, June 27-30, 2005, Volume 2*. Vol. 2. unknown, Jan. 2005, pp. 565–571 (cit. on p. 31).
- [Mni+13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. “Playing Atari with Deep Reinforcement Learning”. In: (Dec. 2013). arXiv: [1312.5602 \[cs.LG\]](#) (cit. on p. 126).
- [Mni+15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dhharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. “Human-level control through deep reinforcement learning”. en. In: *Nature* 518.7540 (Feb. 2015), pp. 529–533 (cit. on pp. 5, 46, 126).
- [MEP12] R Moazzez-Estanjini and I Ch Paschalidis. “On delay-minimized data harvesting with mobile elements in wireless sensor networks”. In: *Ad Hoc Networks* 10.7 (Sept. 2012), pp. 1191–1203 (cit. on pp. 20, 26).
- [MEWP13] Reza Moazzez-Estanjini, Jing Wang, and Ioannis Ch Paschalidis. “Scheduling Mobile Nodes for Cooperative Data Transport in Sensor Networks”. en. In: *Ieee-Acm Transactions on Networking* 21.3 (June 2013), pp. 974–989 (cit. on pp. 18, 26).
- [Mog06] Jeffrey C Mogul. “Emergent (Mis)Behavior vs. Complex Software Systems”. In: *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006*. EuroSys ’06. Leuven, Belgium: ACM, 2006, pp. 293–304 (cit. on p. 98).
- [Mor+04] David L Moreno, Carlos V Regueiro, Roberto Iglesias, and Senén Barro. “Using prior knowledge to improve reinforcement learning in mobile robotics”. In: *Proc. Towards Autonomous Robotics Systems. Univ. of Essex, UK* (2004) (cit. on p. 33).
- [NNN18] Thanh Thi Nguyen, Ngoc Duy Nguyen, and Saeid Nahavandi. “Deep Reinforcement Learning for Multi-Agent Systems: A Review of Challenges, Solutions and Applications”. In: (Dec. 2018). arXiv: [1812.11794 \[cs.LG\]](#) (cit. on pp. 5, 36, 126).
- [NSE08] G S Nitschke, M C Schut, and A E Eiben. “Emergent Specialization in Biologically Inspired Collective Behavior Systems”. In: *Intelligent Complex Adaptive Systems*. Ed. by Ang Yang and Yin Shan. IGI Global, 2008, pp. 215–253 (cit. on p. 123).
- [NRG14] Erik Nordström, Christian Rohner, and Per Gunningberg. “Haggle: Opportunistic mobile content sharing using search”. In: *Comput. Commun.* 48 (July 2014), pp. 121–132 (cit. on p. 15).

- [OS06] R Olfati-Saber. “Flocking for multi-agent dynamic systems: algorithms and theory”. In: *IEEE Trans. Automat. Contr.* 51.3 (Mar. 2006), pp. 401–420 (cit. on pp. 2, 29, 103, 104, 143).
- [PX420] PX4 Dev Team. *Hex Cube Black (FMUV3) · PX4 User Guide*. https://docs.px4.io/master/en/flight_controller/pixhawk-2.html. Accessed: 2020-7-20. 2020 (cit. on p. 134).
- [Paj15] Gonzalo Pajares. “Overview and Current Status of Remote Sensing Applications Based on Unmanned Aerial Vehicles (UAVs)”. In: *Photogrammetric Engineering & Remote Sensing* 81.4 (2015), pp. 281–330 (cit. on p. 1).
- [PY10] Sinno Jialin Pan and Qiang Yang. “A Survey on Transfer Learning”. In: *IEEE Trans. Knowl. Data Eng.* 22.10 (Oct. 2010), pp. 1345–1359 (cit. on p. 141).
- [PL05] Liviu Panait and Sean Luke. “Cooperative multi-agent learning: The state of the art”. en. In: *Auton. Agent. Multi. Agent. Syst.* 11.3 (Nov. 2005), pp. 387–434 (cit. on p. 125).
- [Par+18] Seongjoon Park, Kangho Kim, Hyunsoon Kim, and Hwangnam Kim. “Formation Control Algorithm of Multi-UAV-Based Network Infrastructure”. en. In: *NATO Adv. Sci. Inst. Ser. E Appl. Sci.* 8.10 (Sept. 2018), p. 1740 (cit. on p. 27).
- [Par98] L E Parker. “ALLIANCE: an architecture for fault tolerant multirobot cooperation”. In: *IEEE Trans. Rob. Autom.* 14.2 (Apr. 1998), pp. 220–240 (cit. on p. 32).
- [Par03] H Van Dyke Parunak. “Making swarming happen”. In: *Proceedings of Swarming and Network-Enabled C4ISR*. 2003 (cit. on pp. 29, 30).
- [PB11] B Pearre and T X Brown. “Fast, Scalable, Model-Free Trajectory Optimization for Wireless Data Ferries”. In: *2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN)*. July 2011, pp. 1–8 (cit. on p. 26).
- [PB12] B Pearre and T X Brown. “Energy conservation in sensor network data ferrying: A reinforcement metalearning approach”. In: *2012 IEEE Global Communications Conference (GLOBECOM)*. Dec. 2012, pp. 79–85 (cit. on pp. 21, 26).
- [Pea10] B Pearre. “Model-free trajectory optimisation for wireless data ferries”. In: *IEEE Local Computer Network Conference*. Oct. 2010, pp. 777–784 (cit. on pp. 21, 26).
- [PB10] B Pearre and T X Brown. “Model-free trajectory optimization for wireless data ferries among multiple sources”. In: *2010 IEEE Globecom Workshops*. Dec. 2010, pp. 1793–1798 (cit. on pp. 21, 26).
- [PW10] Ofir Pele and Michael Werman. “The Quadratic-Chi Histogram Distance Family”. In: *Computer Vision – ECCV 2010*. Springer Berlin Heidelberg, 2010, pp. 749–762 (cit. on p. 83).
- [PPC06] L Pelusi, A Passarella, and M Conti. “Opportunistic networking: data forwarding in disconnected mobile ad hoc networks”. In: *IEEE Commun. Mag.* 44.11 (Nov. 2006), pp. 134–141 (cit. on p. 14).
- [PLX12a] J Peng, J Liu, and L Xue. “A Multi-Constraint Ferry Routing Scheme for Delay Tolerant Networks”. In: *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing*. 2012, pp. 1–4 (cit. on pp. 22, 26).
- [PLX12b] Juan Peng, Jiakang Liu, and Liqin Xue. “A Multi-Constraint Ferry Routing Scheme for Delay Tolerant Networks”. en. In: *2012 International Conference on Wireless Communications, Networking and Mobile Computing (wicom)*. International Conference on Wireless Communications Networking and Mobile Computing. 345 E 47th St, New York, Ny 10017 USA: Ieee, 2012 (cit. on pp. 76, 113).
- [Pen03] Mathew Penrose. *Random Geometric Graphs*. en. Oxford University Press, 2003 (cit. on pp. 43, 44).
- [PFH04] A Pentland, R Fletcher, and A Hasson. “DakNet: rethinking connectivity in developing nations”. In: *Computer* 37.1 (Jan. 2004), pp. 78–83 (cit. on p. 13).
- [Pim+13] L C A Pimenta, G A S Pereira, N Michael, R C Mesquita, M M Bosque, L Chaimowicz, and V Kumar. “Swarm Coordination Based on Smoothed Particle Hydrodynamics Technique”. In: *IEEE Trans. Rob.* 29.2 (Apr. 2013), pp. 383–399 (cit. on p. 90).
- [Pol+11] Bahadır K Polat, Pushkar Sachdeva, Mostafa H Ammar, and Ellen W Zegura. “Message ferries as generalized dominating sets in intermittently connected mobile networks”. en. In: *Pervasive Mob. Comput.* 7.2 (Apr. 2011), pp. 189–205 (cit. on p. 14).
- [Pou+18] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and S S Iyengar. “A Survey on Deep Learning: Algorithms, Techniques, and Applications”. In: *ACM Comput. Surv.* 51.5 (Sept. 2018), pp. 1–36 (cit. on p. 5).

- [PG16] Bhaskar Prince and Shruti Gupta. "Load Balanced and Energy Efficient Data Collection Scheme using Data Mule for WSNs". en. In: *2016 Ieee International Conference on Recent Trends in Electronics, Information & Communication Technology (rteict)*. 345 E 47th St, New York, Ny 10017 USA: Ieee, 2016, pp. 1651–1655 (cit. on pp. 16, 17, 26).
- [Qui+09] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. "ROS: an open-source Robot Operating System". In: *ICRA workshop on open source software*. Vol. 3. 2009, p. 5 (cit. on p. 70).
- [RG+20] Panagiotis Radoglou-Grammatikis, Panagiotis Sarigiannidis, Thomas Lagkas, and Ioannis Moscholios. "A compilation of UAV applications for precision agriculture". In: *Computer Networks* 172 (May 2020), p. 107148 (cit. on p. 1).
- [Răd+19] Roxana Rădulescu, Manon Legrand, Kyriakos Efthymiadis, Diederik M Roijers, and Ann Nowé. "Deep Multi-agent Reinforcement Learning in a Homogeneous Open Population". In: *Artificial Intelligence*. Springer International Publishing, 2019, pp. 90–105 (cit. on pp. 34, 126, 128).
- [RTO08] Rao Naveed Bin Rais, Thierry Turetli, and Katia Obraczka. "Coping with Episodic Connectivity in Heterogeneous Networks". In: *Proceedings of the 11th International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. MSWiM '08. New York, NY, USA: ACM, 2008, pp. 211–219 (cit. on p. 13).
- [Rap02] Theodore S Rappaport. *Wireless Communications: Principles and Practice*. en. 2nd. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2002 (cit. on pp. 40, 41).
- [RF18] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement". In: *arXiv* (2018) (cit. on p. 142).
- [Res20] Research Markets LTD. *The Drone Market Report 2020-2025 - Research and Markets*. https://www.researchandmarkets.com/reports/5117908/the-drone-market-report-2020-2025?utm_source=GNOM&utm_medium=PressRelease&utm_code=7rwzgr&utm_campaign=1416129+-+The+Drone+Market+Report+2020-2025&utm_exec=joca220prd. Accessed: 2021-1-25. 2020 (cit. on p. 1).
- [Rey87a] Craig W Reynolds. "Flocks, herds and schools: A distributed behavioral model". In: *ACM SIGGRAPH Computer Graphics* 21.4 (1987), pp. 25–34 (cit. on pp. 2, 126, 138).
- [Rey87b] Craig W Reynolds. "Flocks, herds and schools: A distributed behavioral model". In: *ACM SIGGRAPH Computer Graphics* 21.4 (1987), pp. 25–34 (cit. on pp. 29, 103, 143, 144).
- [RG12] M Riordan and D Grigoros. "Data Mule Service for Mobile Ad-Hoc Networks". In: *2012 11th International Symposium on Parallel and Distributed Computing*. June 2012, pp. 227–234 (cit. on p. 13).
- [Rom88] L J Romkey. "Nonstandard for Transmission of IP Datagrams over Serial Lines : SLIP". In: *RFC 1055* (1988) (cit. on p. 134).
- [RK18] Pol Rosello and Mykel J Kochenderfer. "Multi-agent reinforcement learning for multi-object tracking". In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. 2018, pp. 1397–1404 (cit. on p. 126).
- [Rum+95] David E Rumelhart, Richard Durbin, Richard Golden, and Yves Chauvin. "Backpropagation: The basic theory". In: *Backpropagation: Theory, architectures, and applications*. 561 (1995), pp. 1–34 (cit. on p. 47).
- [Rus99] R A Russell. "Ant trails - an example for robots to follow?". In: *Proceedings 1999 IEEE International Conference on Robotics and Automation*. Vol. 4. 1999, 2698–2703 vol.4 (cit. on p. 30).
- [RN10] Stuart Jonathan Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. en. Prentice Hall, 2010 (cit. on pp. 138, 160).
- [SHK21] Fabrice Saffre, Hanno Hildmann, and Hannu Karvonen. "The Design Challenges of Drone Swarm Control". In: *Engineering Psychology and Cognitive Ergonomics*. Springer International Publishing, 2021, pp. 408–426 (cit. on p. 1).
- [SP07] Massoud Salehi and John Proakis. *Digital Communications*. en. McGraw-Hill Education, Nov. 2007 (cit. on p. 42).
- [San+17] Rodrigo M Santos, Javier Orozco, Daniel Mosse, Vinicius Petrucci, Sergio F Ochoa, and Roc Meseguer. "Flying Real-Time Network for Disaster Assistance". en. In: *Ubiquitous Computing and Ambient Intelligence*. Lecture Notes in Computer Science. Springer, Cham, Nov. 2017, pp. 591–602 (cit. on p. 14).
- [Sau+09] John Sauter, Robert Matthews, Joshua Robinson, John Moody, and Stephanie Riddle. "Swarming Unmanned Air and Ground Systems for Surveillance and Base Protection". In: *AIAA Infotech@Aerospace Conference*. Infotech@Aerospace Conferences. American Institute of Aeronautics and Astronautics, Apr. 2009 (cit. on pp. 28–30).

- [Sce+04] Paul Scerri, Alessandro Farinelli, Stephen Okamoto, and Milind Tambe. "Allocating roles in extreme teams". In: *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*. 2004, pp. 1502–1503 (cit. on p. 31).
- [SS87] F W Scholz and M A Stephens. "K-Sample Anderson–Darling Tests". In: *J. Am. Stat. Assoc.* 82.399 (Sept. 1987), pp. 918–924 (cit. on p. 95).
- [Sch+20] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. "Mastering Atari, Go, chess and shogi by planning with a learned model". en. In: *Nature* 588.7839 (Dec. 2020), pp. 604–609 (cit. on p. 5).
- [Sch+18] Janaína Schwarzrock, Iulislou Zacarias, Ana L C Bazzan, Ricardo Queiroz de Araujo Fernandes, Leonardo Henrique Moreira, and Edison Pignaton de Freitas. "Solving task allocation problem in multi Unmanned Aerial Vehicles systems using Swarm intelligence". In: *Eng. Appl. Artif. Intell.* 72 (June 2018), pp. 10–20 (cit. on p. 32).
- [See+04] P Seem, A Farinelli, S Okamoto, and M Tambe. "Token approach for role allocation in extreme teams: analysis and experimental evaluation". In: *13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*. June 2004, pp. 397–402 (cit. on p. 31).
- [Sej18] Terrence J Sejnowski. *The Deep Learning Revolution*. en. MIT Press, Oct. 2018 (cit. on pp. 5, 127).
- [SB15a] Samaneh Hosseini Semnani and Otman A Basir. "Semi-flocking algorithm for motion control of mobile sensors in large-scale surveillance systems". en. In: *IEEE Trans Cybern* 45.1 (Jan. 2015), pp. 129–137 (cit. on pp. 4, 126).
- [SB15b] Samaneh Hosseini Semnani and Otman A Basir. "Semi-flocking algorithm for motion control of mobile sensors in large-scale surveillance systems". en. In: *IEEE Trans Cybern* 45.1 (Jan. 2015), pp. 129–137 (cit. on pp. 29, 103, 111, 143).
- [Sha+03] R C Shah, S Roy, S Jain, and W Brunette. "Data MULEs: modeling a three-tier architecture for sparse sensor networks". In: *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, 2003*. May 2003, pp. 30–41 (cit. on p. 14).
- [Sha97] C E Shannon. "The Mathematical Theory of Communication. 1963". en. In: *MD Comput.* 14.4 (July 1997), pp. 306–317 (cit. on p. 154).
- [Shi+06] T Shinkawa, T Terauchi, T Kitani, N Shibata, K Yasumoto, M Ito, and T Higashino. "A Technique for Information Sharing using Inter-Vehicle Communication with Message Ferrying". In: *7th International Conference on Mobile Data Management (MDM'06)*. Vol. 2006. May 2006, pp. 130–130 (cit. on p. 13).
- [ST93] David B Shmoys and Éva Tardos. "An approximation algorithm for the generalized assignment problem". In: *Math. Program.* 62.1 (Feb. 1993), pp. 461–474 (cit. on pp. 31, 35).
- [Sil+16] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. "Mastering the game of Go with deep neural networks and tree search". en. In: *Nature* 529.7587 (Jan. 2016), pp. 484–489 (cit. on p. 126).
- [SMT13] T Simon and A Mitschele-Thiel. "A self-organized message ferrying algorithm". In: *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*. June 2013, pp. 1–6 (cit. on pp. 23, 24, 26, 30, 76, 84, 113).
- [SKS16] S K Singh, P Kumar, and J P Singh. "An energy efficient Odd-Even round number based data collection using mules in WSNs". In: *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*. Mar. 2016, pp. 1255–1259 (cit. on pp. 16, 17, 26).
- [ST10] Achudhan Sivakumar and Colin Keng-Yan Tan. "UAV Swarm Coordination Using Cooperative Control for Establishing a Wireless Communications Backbone". In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 3 - Volume 3*. AAMAS '10. Toronto, Canada: International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 1157–1164 (cit. on pp. 1, 14, 24, 26, 30).
- [SH03] Tara Small and Zygmunt J Haas. "The Shared Wireless Infostation Model: A New Ad Hoc Networking Paradigm (or Where There is a Whale, There is a Way)". In: *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing*. MobiHoc '03. New York, NY, USA: ACM, 2003, pp. 233–244 (cit. on p. 13).
- [Sma15] David Smalley. *News: Autonomous, swarming UAVs fly into the future - Office of Naval Research*. <https://www.onr.navy.mil/Media-Center/Press-Releases/2015/LOCUST-low-cost-UAV-swarm-ONR.aspx>. Accessed: 2017-3-24. 2015 (cit. on p. 4).

- [Smi+17] P Smith, R Hunjed, A Aleti, and J C Barca. “Adaptive data transfer methods via policy evolution for UAV swarms”. In: *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*. ieeexplore.ieee.org, Nov. 2017, pp. 1–8 (cit. on p. 141).
- [Smi+18] Phillip Smith, Robert Hunjet, Aldeida Aleti, and Jan Carlo Barca. “Data transfer via UAV swarm behaviours: Rule generation, evolution and learning”. In: *Journal of Telecommunications and the Digital Economy* 6.2 (2018), pp. 35–57 (cit. on p. 141).
- [SRS04] A A Somasundara, A Ramamoorthy, and M B Srivastava. “Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines”. In: *25th IEEE International Real-Time Systems Symposium*. Dec. 2004, pp. 296–305 (cit. on p. 20).
- [Sta16] Stanford University. *ImageNet*. <http://www.image-net.org/>. Accessed: 2020-7-24. 2016 (cit. on p. 142).
- [Ste15] Stefan Steinerberger. “New bounds for the traveling salesman constant”. en. In: *Adv. Appl. Probab.* 47.1 (Mar. 2015), pp. 27–36 (cit. on p. 88).
- [Sto+19] Petr Stodola, Jan Drozd, Jan Mazal, Jan Hodický, and Dalibor Procházka. “Cooperative Unmanned Aerial System Reconnaissance in a Complex Urban Environment and Uneven Terrain”. en. In: *Sensors* 19.17 (Aug. 2019) (cit. on p. 1).
- [Sto02] I Stojmenovic. “Position-based routing in ad hoc networks”. In: *IEEE Commun. Mag.* 40.7 (July 2002), pp. 128–134 (cit. on p. 77).
- [SG11] Ryo Sugihara and Rajesh K Gupta. “Path Planning of Data Mules in Sensor Networks”. en. In: *Acm Transactions on Sensor Networks* 8.1 (Aug. 2011) (cit. on pp. 18, 26).
- [SB17a] R S Sutton and A G Barto. “Reinforcement learning: An introduction (Complete 2nd Edition Draft)”. 2017 (cit. on pp. 46, 47, 128).
- [SB17b] Claudia Szabo and Lachlan Birdsey. “Validating Emergent Behavior in Complex Systems”. en. In: *Advances in Modeling and Simulation*. Simulation Foundations, Methods and Applications. Springer, Cham, 2017, pp. 47–62 (cit. on p. 32).
- [STC14] Claudia Szabo, Yong Meng Teo, and Gautam K Chengleput. “Understanding Complex Systems: Using Interaction as a Measure of Emergence”. In: *Proceedings of the 2014 Winter Simulation Conference*. 2014, edited by A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, 207–218. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc. (Cit. on p. 145).
- [Tam+15] Ardi Tampuu, Tambet Matiisen, Dorian Kodolja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. “Multiagent Cooperation and Competition with Deep Reinforcement Learning”. In: (Nov. 2015). arXiv: 1511.08779 [cs.AI] (cit. on p. 126).
- [TK10] Sunil Taneja and Ashwani Kush. “A survey of routing protocols in mobile ad hoc networks”. In: *Int. J. Innov. Technol. Manage.* 1.3 (2010), p. 279 (cit. on p. 42).
- [TB99] G Theraulaz and E Bonabeau. “A Brief History of Stigmergy”. In: *Artif. Life* 5.2 (Apr. 1999), pp. 97–116 (cit. on p. 2).
- [Toh02] Chai-Keong Toh. *Ad Hoc Mobile Wireless Networks: Protocols and Systems*. en. 1st. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2002 (cit. on pp. 3, 11).
- [TH11] Gwo-Hshiung Tzeng and Jih-Jeng Huang. *Multiple attribute decision making. Methods and applications*. 6000 Broken Sound Parkway NW, Suite 300, Boca Raton, FL 33487-2742: CRC Press, Jan. 2011 (cit. on pp. 6, 49, 114).
- [US 17] US Department of Defense. *Department of Defense Announces Successful Micro-Drone Demonstration*. <https://www.defense.gov/News/News-Releases/News-Release-View/Article/1044811/departement-of-defense-announces-successful-micro-drone-demonstration>. Accessed: 2017-3-24. 2017 (cit. on p. 4).
- [Usb+14] Kyle Usbeck, Matthew Gillen, Joseph Loyall, Andrew Gronosky, Joshua Sterling, Ralph Kohler, Richard Newkirk, and David Canestrare. “Data Ferrying to the Tactical Edge: A Field Experiment in Exchanging Mission Plans and Intelligence in Austere Environments”. en. In: *2014 Ieee Military Communications Conference: Affordable Mission Success: Meeting the Challenge (milcom 2014)*. IEEE Military Communications Conference. Raytheon, Baltimore, MD: Ieee Computer Soc, Oct. 2014, pp. 1311–1317 (cit. on p. 1).
- [VB00] Amin Vahdat and David Becker. *Epidemic Routing for Partially-Connected Ad Hoc Networks*. Tech. rep. CS-2000-06. Duke University, Apr. 2000 (cit. on p. 12).
- [VJ97] Christine L Valenzuela and Antonia J Jones. “Estimating the Held-Karp lower bound for the geometric TSP”. In: *Eur. J. Oper. Res.* 102.1 (Oct. 1997), pp. 157–175 (cit. on p. 88).

- [VDPB01] H Van Dyke Parunak and Sven Brueckner. “Entropy and Self-organization in Multi-agent Systems”. In: *Proceedings of the Fifth International Conference on Autonomous Agents*. AGENTS ’01. New York, NY, USA: ACM, 2001, pp. 124–130 (cit. on p. 153).
- [VKK16] G C Vanarotti, U M Kulkarni, and H H Kenchannavar. “Ferry based data gathering in Wireless Sensor Networks”. In: *2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATcT)*. July 2016, pp. 165–170 (cit. on pp. 19, 26).
- [Vás+14] G Vászrhelyi, C Virágh, G Somorjai, N Tarcai, T Szörényi, T Nepusz, and T Vicsek. “Outdoor flocking and formation flight with autonomous aerial robots”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2014, pp. 3866–3873 (cit. on p. 29).
- [VY13] N Vastardis and K Yang. “Mobile Social Networks: Architectures, Social Properties, and Key Research Challenges”. In: *IEEE Communications Surveys Tutorials* 15.3 (2013), pp. 1355–1371 (cit. on p. 12).
- [Ver45] P F Verhulst. “Recherches mathématiques sur la loi d’accroissement de la population”. In: *Nouveaux mémoires de l’Académie Royale des Sciences et Belles-Lettres de Bruxelles* 18 (1845), pp. 14–54 (cit. on p. 61).
- [VMP14] Matthias Vigelius, Bernd Meyer, and Geoffrey Pascoe. “Multiscale modelling and analysis of collective decision making in swarm robotics”. en. In: *PLoS One* 9.11 (Nov. 2014), e111542 (cit. on p. 90).
- [VLC05] R Viswanathan, Jing Li, and Mooi Choo Chuah. “Message ferrying for constrained scenarios”. In: *Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks*. June 2005, pp. 487–489 (cit. on pp. 20, 26).
- [WNMA15] Mohd Nadhir Ab Wahab, Samia Nefti-Meziani, and Adham Atyabi. “A Comprehensive Review of Swarm Optimization Algorithms”. In: *PLoS One* 10.5 (2015), e0122827 (cit. on p. 2).
- [Wal06] Kenneth Wallis. *A note on the calculation of entropy from histograms*. Tech. rep. University of Warwick, 2006 (cit. on p. 154).
- [Wal+06] Bryan Walter, Adrian Sannier, Dirk Reiners, and James Oliver. “UAV Swarm Control: Calculating Digital Pheromone Fields with the GPU”. In: *Journal of Defense Modeling & Simulation* 3.3 (July 2006), pp. 167–176 (cit. on p. 30).
- [Wan09] Jing Wang. “Trajectory Optimization of Mobile Element for Sparse Wireless Sensor Networks”. en. In: *Proceedings of the International Conference on Communication Software and Networks*. 10662 Los Vaqueros Circle, Po Box 3014, Los Alamitos, Ca 90720-1264 USA: Ieee Computer Soc, 2009, pp. 829–833 (cit. on pp. 17, 26).
- [WL10] T Wang and C P Low. “Dynamic Message Ferry Route (dMFR) for Partitioned MANETs”. In: *2010 International Conference on Communications and Mobile Computing*. Vol. 3. Apr. 2010, pp. 447–451 (cit. on pp. 22, 26, 80, 82).
- [WL11] T Wang and C P Low. “Reducing message delay with the general Message Ferry Route (MFR*) problem”. In: *2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. Oct. 2011, pp. 380–387 (cit. on pp. 20, 26).
- [Wan+20] Tonghan Wang, Heng Dong, Victor Lesser, and Chongjie Zhang. “ROMA: Multi-Agent Reinforcement Learning with Emergent Roles”. In: (Mar. 2020). arXiv: 2003.08039 [cs.MA] (cit. on p. 141).
- [WD92] Christopher J C H Watkins and Peter Dayan. “Q-learning”. In: *Mach. Learn.* 8.3 (May 1992), pp. 279–292 (cit. on p. 46).
- [Wei] Eric W Weisstein. *Erfc*. en. <http://mathworld.wolfram.com/Erfc.html>. Accessed: 2018-5-29 (cit. on p. 42).
- [Wei04] Eric W Weisstein. *Square line picking*. <http://mathworld.wolfram.com/SquareLinePicking.html>. 2004 (cit. on p. 108).
- [Win18] A F Winfield. “Experiments in artificial theory of mind: From safety to story-telling”. en. In: *Frontiers in Robotics and AI* 5.75 (June 2018) (cit. on p. 132).
- [Wu+12] X Wu, Y Chen, M Xu, and W Peng. “TAFR: A TTL-aware Message Ferry Scheme in DTN”. In: *2012 Fourth International Conference on Computational and Information Sciences*. Aug. 2012, pp. 1380–1383 (cit. on pp. 23, 26, 76).
- [Xue+14] L Xue, D Kim, Y Zhu, D Li, W Wang, and A O Tokuta. “Multiple heterogeneous data ferry trajectory planning in wireless sensor networks”. In: *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*. Apr. 2014, pp. 2274–2282 (cit. on pp. 19, 26).
- [XLP12] Liqin Xue, Jiakang Liu, and Juan Peng. “An adaptive message ferry routing algorithm for Delay tolerant Networks”. In: *2012 IEEE 14th International Conference on Communication Technology*. Nov. 2012, pp. 699–703 (cit. on p. 14).

- [Yan+05] Jeonghwa Yang, Yang Chen, M Ammar, and Chungkee Lee. “Ferry replacement protocols in sparse MANET message ferrying systems”. In: *IEEE Wireless Communications and Networking Conference, 2005*. Vol. 4. Mar. 2005, 2038–2044 Vol. 4 (cit. on p. 14).
- [YTQ08] Mao Yang, Yantao Tian, and Xinyue Qi. *Behavior Analysis of Swarm Robot Systems Based on Vicsek Model*. 2008 (cit. on p. 90).
- [YK09] Danlei Yu and Young-Bae Ko. “FFRDV: Fastest-Ferry Routing in DTN-enabled Vehicular Ad Hoc Networks”. In: *2009 11th International Conference on Advanced Communication Technology*. Vol. 02. Feb. 2009, pp. 1410–1414 (cit. on p. 13).
- [Yua+18] W Yuan, N Ganganath, C Cheng, G Qing, and F C M Lau. “Path Planning for Semi-Flocking-Controlled Mobile Sensor Networks on Mobility Maps”. In: *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. May 2018, pp. 1–5 (cit. on pp. 4, 126).
- [Yuc+13] Baris Yuce, Michael S Packianather, Ernesto Mastrocinque, Duc Truong Pham, and Alfredo Lambiase. “Honey Bees Inspired Optimization Method: The Bees Algorithm”. en. In: *Insects* 4.4 (Nov. 2013), pp. 646–662 (cit. on p. 3).
- [ZF07] Z Zhang and Z Fei. “Route Design for Multiple Ferries in Delay Tolerant Networks”. In: *2007 IEEE Wireless Communications and Networking Conference*. Mar. 2007, pp. 3460–3465 (cit. on pp. 18, 26).
- [ZC08] J Zhao and G Cao. “VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks”. In: *IEEE Trans. Veh. Technol.* 57.3 (May 2008), pp. 1910–1922 (cit. on p. 13).
- [ZAZ04] W Zhao, M Ammar, and E Zegura. “A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks”. In: *Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. MobiHoc '04. Roppongi Hills, Tokyo, Japan: ACM, 2004, pp. 187–198 (cit. on p. 17).
- [ZAZ05] W Zhao, M Ammar, and E Zegura. “Controlling the mobility of multiple data transport ferries in a delay-tolerant network”. In: *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies*. Vol. 2. Mar. 2005, 1407–1418 vol. 2 (cit. on pp. 18, 26).
- [ZA03] Wenrui Zhao and M H Ammar. “Message ferrying: proactive routing in highly-partitioned wireless ad hoc networks”. In: *The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems, 2003. FTDCS 2003. Proceedings*. ieeexplore.ieee.org, May 2003, pp. 308–314 (cit. on pp. 14, 17, 22, 26).
- [ZSS17] Tsvetan Zhivkov, Eric Schneider, and Elizabeth I Sklar. “Measuring the Effects of Communication Quality on Multi-robot Team Performance”. In: *Towards Autonomous Robotic Systems*. Springer International Publishing, 2017, pp. 408–420 (cit. on p. 33).
- [Zho+17] S Zhou, Q Zhong, B Ou, and Y Liu. “Intelligent compressive data gathering using data ferries for wireless sensor networks”. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2017, pp. 6015–6019 (cit. on p. 21).
- [ZRW20] Y Zhou, B Rao, and W Wang. “UAV Swarm Intelligence: Recent Advances and Future Trends”. In: *IEEE Access* 8 (2020), pp. 183856–183878 (cit. on p. 2).
- [Zhu+17] Bing Zhu, Lihua Xie, Duo Han, Xiangyu Meng, and Rodney Teo. “A survey on recent progress in control of swarm systems”. In: *Sci. China Inf. Sci.* 60.7 (June 2017), p. 070201 (cit. on p. 111).

The End.