UNIVERSITY OF VAASA

FACULTY OF TECHNOLOGY

TELECOMMUNICATIONS ENGINEERING

Ken Helin

TOPOLOGY CONTROL MULTI-OBJECTIVE OPTIMISATION IN WIRELESS SENSOR NETWORKS

Connectivity-Based Range Assignment and Node Deployment

Master's thesis for the degree of Master of Science in Technology submitted for inspection, Vaasa, 30 August, 2011.

Supervisor D.Sc. (Tech.) Mohammed Salem Elmusrati

Instructor M.Sc. Reino Virrankoski

PREFACE

In the last step, only then,
is that of the first realised.
To be so, is a blessing.
For it to be otherwise,
the first ne'er will be taken,
in fear of its consequence.

TABLE OF CONTENTS

List of Figures 6

List of Tables 8

Abbreviations 9

Glossary 11

Abstract 12

Chapter 1 Introduction 13

Chapter 2 Wireless Sensor Networks 15
2.1 Device Architecture . . . 15
2.2 Network Architecture . . . 16
2.3 Domain of Application . . . 17
2.4 Challenges for Development . . . 18
2.5 Open Issues . . . 19

Chapter 3 Topology Control 21
3.1 A Definition . . . 21
3.2 Motivations . . . 23
3.2.1 Energy Conservation . . . 23
3.2.2 Increased Network Capacity . . . 24
3.3 A Taxonomy . . . 27

Chapter 4 Centralised Transmission Power Control 29
4.1 Transmitting Range . . . 29
4.2 Problem Definition . . . 30
4.2.1 Homogeneous - Critical Transmission Range (CTR) . . . 30
4.2.2 Non-Homogeneous - Range Assignment (RA) . . . 30
4.3 Applicability of Approach to Application Domain . . . 31

Chapter 5 Optimisation Algorithms 32
5.1 MatlabBGL - A Graph Library . . . 32
5.2 Graph Representation . . . 32
5.3 A Basis for Comparison . . . 34
5.4 Objective Functions . . . 36

5.4.1 Connected Components (CC) . . . 37
5.4.2 Biconnected Components (BC) . . . 38
5.4.3 Global Energy Consumption (GEC) . . . 39
5.4.4 Local (Per-Node) Energy Consumption (LEC) . . . 40
5.4.5 Node Degree (ND) . . . 41
5.4.6 Towards A Unified Schema . . . 45
5.5 Euclidean Minimum Spanning Tree (EMST) . . . 47
5.6 Genetic and Evolutionary Algorithm (GEA) . . . 49
5.7 Decision Criteria . . . 52
5.7.1 Analytic Hierarchy Process (AHP) . . . 53
5.7.2 Minimal Range Deviation (MRD) . . . 56
5.8 Special Considerations . . . 57

Chapter 6 Simulation Scenario Design and Development 62
6.1 Mobility Model . . . 62
6.2 Node Deployment . . . 63
6.3 Simulink Model Parameters . . . 66
6.4 Communication Graph . . . 67

Chapter 7 Design of Experiments 73
7.1 Design Statement . . . 73
7.2 Data Representation . . . 73
7.2.1 Threshold of Connectivity (ToC) . . . 73
7.2.2 Connectivity-Time Ratio (CtR) . . . 74
7.3 Data Collation . . . 75
7.4 Data Visualisation . . . 77

Chapter 8 Discussion and Results 78
8.1 Relative Algorithmic Performance . . . 79
8.1.1 Minimal Range . . . 79
8.1.2 Medial Range . . . 81
8.1.3 Maximal Range . . . 82
8.2 Threshold-Scaling Coefficient . . . 87

Chapter 9 Conclusions 92

Bibliography 93

Appendices 95

Appendix A Elements of Graph Theory 96
A.1 Basic Definitions . . . 96
A.2 Advanced Concepts . . . 98
A.3 Proximity Graphs . . . 100

Appendix B PiccSIM - A Wireless Networked Control System Simulation Platform 101
B.1 Motivation . . . 101
B.2 Key Features . . . 101
B.3 Architecture . . . 102
B.4 Network Simulator Integration . . . 102
B.5 A (Modified) Systematic Approach . . . 104

Appendix C Pareto Optimality 106

Appendix D Genetic Algorithm Parameters 109
D.1 Population Options . . . 110
D.2 Selection Options . . . 112
D.3 Mutation Options . . . 112
D.4 Crossover Options . . . 113
D.5 Migration Options . . . 113
D.6 Multi-Objective Options . . . 115
D.7 Stopping Criteria Options . . . 115
D.8 Vectorize Option . . . 116

Appendix E A Steady-State Mobility File Generator 117

Appendix F Simulink Model Architecture 121
F.1 Root Block Diagram . . . 121
F.2 Subsystem Component Block Diagrams . . . 121
F.2.1 Mobility Subsystem . . . 121
F.2.2 Network Subsystem . . . 123
F.2.3 Optimiser Subsystem . . . 123
F.2.4 Range Assignment Subsystem . . . 124
F.2.5 Visualisation Subsystem . . . 124

LIST OF FIGURES

2.1 Generic architecture of a wireless sensor device 15
2.2 Sensinode NanoSensor N711 specifications 16
2.3 Generic architecture of a wireless sensor network 17

3.1 The case for multi-hop communication: energy consumption 24
3.2 Conflicting wireless transmissions 25
3.3 The case for multi-hop communication: network capacity 26
3.4 A taxonomy of topology control techniques 28

4.1 Radio coverage in two-dimensional networks 29

5.1 An example digraph (directed graph) 33
5.2 DT of a random set of 100 points in the plane 49
5.3 EMST of a random set of 25 points in the plane 50
5.4 Multi-criteria weighting vector values derived via AHP method 55
5.5 An example strongly connected graph 57
5.6 Dominator trees of the flowgraphs relative to the example graph 60

6.1 Simulink model optimiser subsystem component block mask dialog box . . . 67
6.2 State of the network visualisation for a simulated GA 68
6.3 State of the network visualisation for a simulated EMST 69

8.1 Connectivity-time ratio per run for each algorithm at minimal range 83
8.2 Connectivity-time ratio per run for each algorithm at medial range 84
8.3 Connectivity-time ratio per run for each algorithm at maximal range 85
8.4 Connectivity-time ratio kernel density estimators for GA optimisation 86

B.1 Network Simulator Emulator (NSE): Flow Diagram Internals 103
B.2 UDP port mapping of xPC Target and NS-2 machine nodes 104

C.1 Optimal solution set for a two-objective optimisation problem 108

F.1 Simulink model root block diagram for topology control optimisation 121
F.2 Mobility subsystem component block diagram 121
F.3 Mobility subsystem global sub-component block diagram 122
F.4 Mobility subsystem local sub-component block diagram instance 122
F.5 Network subsystem component block diagram 123
F.6 Network subsystem node sub-component block diagram instance 123
F.7 Optimiser subsystem component block diagram 123

F.8 Range assignment subsystem component block diagram 124
F.9 Visualisation subsystem component block diagram 124

List of Tables

5.1 Value range for the respective objective function and optimisation type 46
5.2 Pair-wise relative comparison of identified design criteria importance 55

6.1 Steady-state mobility file generator input parameters 64

ABBREVIATIONS

AHP	Analytic Hierarchy Process
BC	Biconnected Components
CC	(Strongly) Connected Components
CTR	Critical Transmission Range
DOE	Design of Experiments
DT	Delaunay Triangulation
EA	Evolutionary Algorithm
EMST	Euclidean Minimum Spanning Tree
GA	Genetic Algorithm
GADS	Genetic Algorithm and Direct Search (Toolbox)
GEA	Genetic and Evolutionary Algorithm
GEC	Global Energy Consumption
KDE	Kernel Density Estimation
LEC	Local (Per-Node) Energy Consumption
MCDM	Multi-Criteria Decision Making
MOEA	Multi-Objective Evolutionary Algorithm
MOO	Multi-Objective Optimisation
MOOP	Multi-Objective Optimisation Problem
MRD	Minimal Range Deviation
MST	Minimum Spanning Tree
ND	Node Degree
RA	Range Assignment
RWM	Random Waypoint Mobility

SOO Single-Objective Optimisation

SOOP Single-Objective Optimisation Problem

ToC Threshold of Connectivity

WSN Wireless Sensor Network

GLOSSARY

(Strongly) Connected Components The strongly connected components of a directed graph are its maximal strongly connected subgraphs.

Biconnected Components A maximal subset of edges of a connected graph such that the corresponding induced subgraph cannot be disconnected by deleting any vertex.

Digraph Directed graph.

UNIVERSITY OF VAASA
Faculty of technology
Author: Ken Helin
Topic of the Thesis: Topology Control Multi-Objective Optimisation in Wireless Sensor Networks: Connectivity-Based Range Assignment and Node Deployment
Supervisor: Mohammed Salem Elmusrati
Instructor: Reino Virrankoski
Degree: Master of Science in Technology
Department: Department of Computer Science
Degree Programme: Degree Programme in Information Technology
Major of Subject: Telecommunications Engineering
Year of Entering the University: 2006
Year of Completing the Thesis: 2011 **Pages:** 124

ABSTRACT:

The distinguishing characteristic that sets topology control apart from other methods, whose motivation is to achieve effects of energy minimisation and an increased network capacity, is its network-wide perspective. In other words, local choices made at the node-level always have the goal in mind of achieving a certain global, network-wide property, while not excluding the possibility for consideration of more localised factors. As such, our approach is marked by being a centralised computation of the available location-based data and its reduction to a set of non-homogeneous transmitting range assignments, which elicit a certain network-wide property constituted as a whole, namely, strong connectedness and/or biconnectedness.

As a means to effect, we propose a variety of GA which by design is multi-morphic, where dependent upon model parameters that can be dynamically set by the user, the algorithm, acting accordingly upon either single or multiple objective functions in response. In either case, leveraging the unique faculty of GAs for finding multiple optimal solutions in a single pass. Wherefore it is up to the designer to select the singular solution which best meets requirements.

By means of simulation, we endeavour to establish its relative performance against an optimisation typifying a standard topology control technique in the literature in terms of the proportion of time the network exhibited the property of strong connectedness.

As to which, an analysis of the results indicates that such is highly sensitive to factors of: the effective maximum transmitting range, node density, and mobility scenario under observation. We derive an estimate of the optimal constitution thereof taking into account the specific conditions within the domain of application in that of a WSN, thereby concluding that only GA optimising for the biconnected components in a network achieves the stated objective of a sustained connected status throughout the duration.

KEYWORDS: Topology control, multi-objective optimisation, genetic algorithm, wireless sensor networks.

1 INTRODUCTION

As a field of research, wireless sensor networks have engendered a great deal of interest in the community. Such can well be understood in terms of its vast potential as a means for solving real world problems in a wide array of application areas and in unique ways not possible even until recently. This does not come without ensuing issues and challenges that need be addressed from the outset in order that any implementation be successful in its endeavour.

The overwhelming tendency in approaching problems of such a nature has been to adopt a very narrow focus to the detriment of the prevailing conditions in a wider purview. Hence, the motivating factor for the present work was to take the alternate view or perspective of the same situation. In particular, recognising the primary importance of the network's connected status being uninterrupted, even in the presence of mobility or the potential for link(s) and/or node(s) failures, thereby giving it preeminence over all other objectives that may appertain to the specific problem. Otherwise, continued operation is jeopardised and human intervention in some form is necessitated upon, which may not be practicable depending on the environment in which the nodes are deployed. Therefore, being mindful of the inherent difficulty in reconfiguring deployed nodes out in the field, unless allowance is made and catered for, it was seen as an imperative to factor into the design features of adaptability and flexibility in any resulting implementation.

That being the case, it was found that topology control affords the theoretical bases in which to formulate the problem from the network-wide perspective, which is its distinguishing characteristic that sets it apart from other methods, and whose corollary is to achieve effects of energy minimisation and an increased network capacity. From such vantage point, local choices made at the node-level always have the goal in mind of achieving a certain global, network-wide property, while not excluding the possibility for consideration of more localised factors. Thus constituting its suitability and immediacy as the overall framework and context in which to pose the topic.

The remainder of the present work is structured as follows. Chapter 2 lays the groundwork for an appreciation of the unique challenges facing the network designer and planner specific to the domain of application anticipated for that of our findings in that of a WSN. In Chapter 3, we define the motivations for the use of topology control as a methodology in addressing the issues raised and the means for identifying the particular approach adopted

for our implementation, thereby placing it within the proper context of the literature on the topic. Chapter 4 formulates the problem definition in explicit terms of the taxonomical schema proposed for classification of topology control techniques and its applicability to the application domain. Chapter 5 constitutes the primary focus of the work and as such sets forth in great detail the manner in which the proposed optimisation procedures are to be composed and their theoretical bases. In particular, that of a GA implementation with its requisite specification of the fitness function(s) upon which it operates within a multi-objective optimisation paradigm. Chapter 6 details the simulation framework in which the development process was carried out and its ramifications for the design of experiments to be established in Chapter 7. The results of which are to be presented in Chapter 8. Finally, in Chapter 9, we recapitulate our findings and analyses thereof with the aim of drawing conclusions as well as their implications for further potential studies in the topic.

2 Wireless Sensor Networks

Wireless sensor networks (WSNs) are collections of compact-sized, relatively inexpensive computational nodes that measure local environmental conditions or other parameters and forward such information to a central point for further appropriate processing. Wireless sensor nodes can sense the environment, can communicate with neighbouring nodes, and can, in many cases, perform basic computations on the data being collected. WSNs are supportive of a wide range of useful applications which explains their continued interest as a field for research (Sohraby, Minoli & Znati 2007: p. 38).

2.1 Device Architecture

A wireless sensor is a limited device whose architecture is evidenced by a marked simplicity (see Figure 2.1): a processor, memory, a radio transceiver and antenna, a power source, and an input/output interface that allows the integration of external sensors. This reflects the uncomplicated and highly specialised nature of the typical tasks a wireless sensor device is called upon to perform. In addition, to it functioning within the ever-present constraint of diminutive energy reserves since wireless sensor devices are usually battery powered. As such, every element of their design must take these factors into account. For instance, the choice of micro-controller over that of the more familiar micro-processor by virtue of their considerably lower energy consumption in general and the ability to harness finer-grained processor states (Labrador & Wightman 2009: p. 4).

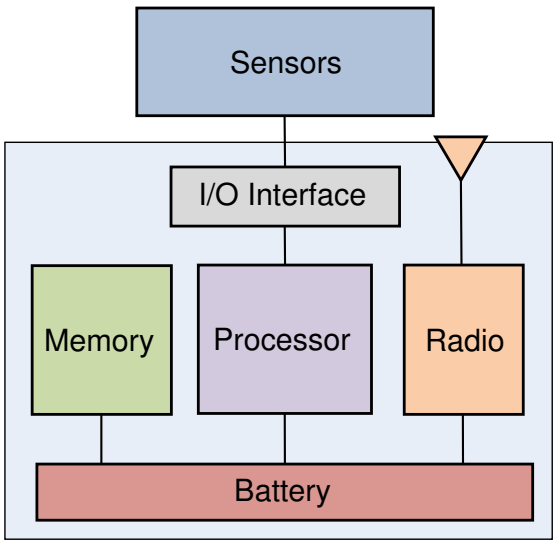

Figure 2.1: Generic architecture of a wireless sensor device.

The gaining popularity of WSNs can in large part be due to the flexibility with which the devices themselves can be interfaced with a large variety of application-specific sensors. Temperature, air quality, pressure, magnetometers, light, acoustic, and accelerometers, are just a small sample of the types of commercially available sensors. Thus serving as a general platform in which to solve practical problems in many application domains (Labrador & Wightman 2009: p. 5). The Sensinode NanoSensor N711 (see Figure 2.2) is a reference design for IP-based wireless sensor networking that furnishes a characteristic example. There are two sensors mounted upon the circuit board, a National Semiconductor LM60 analog temperature sensor and an Intersil EL7900 ambient light sensor. The board also features two LED's and two buttons. The sensors, indicator LED's, and buttons are connected to the Radiocrafts RC2301AT RF Transceiver Module IO pins (Sensinode Ltd. 2007).

Features

- AA battery holder integrated
- 2 LEDs and 2 buttons
- Temperature sensor
- Light sensor
- Open device with programming tools for user firmware and NanoStack™ support
- Programmable through the Sensinode Devboard
- Integrated RadioCrafts RC2301AT module
- Powerful TI CC2431 32 MHz single-cycle low power 8051 MCU
- 2.4 GHz IEEE 802.15.4 compliant RF-transceiver with 250kbps data rate
- 128kB programmable FLASH, 8kB SRAM
- Integrated positioning engine
- Integrated chip-antenna
- Solder pads for common signals for prototyping
- PIN-headers for programming and UARTs
- RoHS compatible
- Meets CE, FCC and ETSI requirements

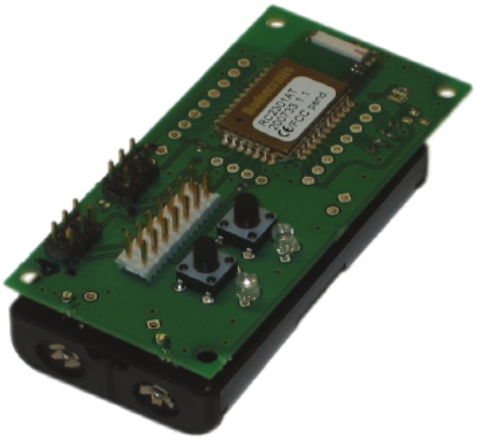

Figure 2.2: NanoSensor N711.

2.2 Network Architecture

Wireless sensor network architectures have evolved over time and continue to evolve as new devices and capabilities become available. Initially, WSNs consisted of a flat topology made up of homogeneous devices measuring a single variable. Most of these networks comprised several (not many) wireless sensor devices deployed throughout the region of interest and a single sink node acting as a gateway to other networks enabling the aggregate sensor data to be made available remotely for purposes of monitoring and analysis.

Newly discovered domains of application utilising larger scale node deployments, necessitated architectural developments for the efficient transmission of wireless sensor data. As such, layered or clustered topologies were incorporated that included multiple sinks increasing robustness amongst other benefits therewith. Further, greater flexibility in options for connectivity with the outside world became a necessity: the Internet, private networks, cellular networks, wireless ad hoc networks, and so on. As a consequence, this triggered large efforts in the design and implementation of improved communication protocols to incorporate these new capabilities.

As an example architecture, Figure 2.3 shows two small-scale wireless sensor networks of flat topology each with a single sink node connected to a wireless ad hoc network, cellular network and the Internet, which at the same time serves as a bridge interconnecting the sub-networks with each other.

(Labrador & Wightman 2009: p. 6)

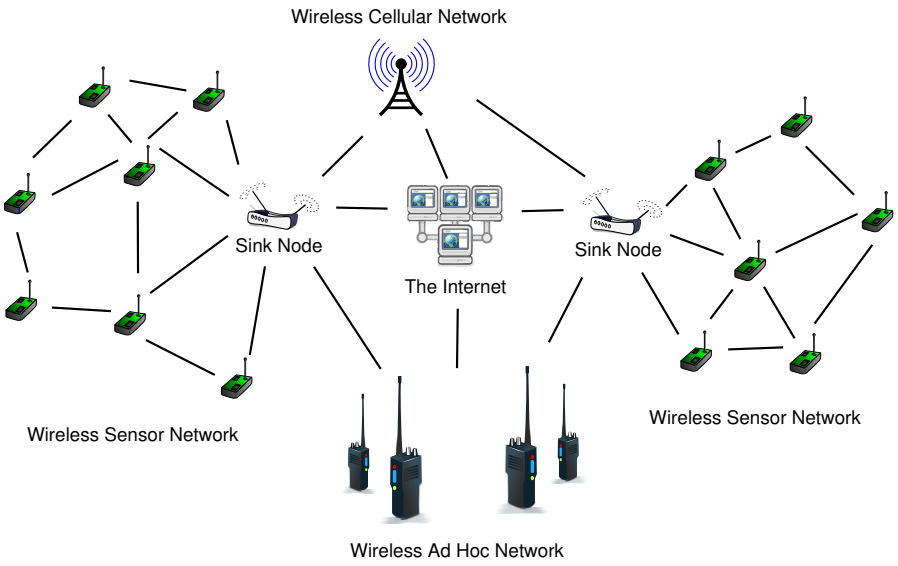

Figure 2.3: Generic architecture of a wireless sensor network.

2.3 Domain of Application

The literature abounds with myriad ways in which WSNs can be used in the field and is really only limited by one's creativity as to their possible application. By way of example,

WSNs have found successful implementation within the following application domains (Sohraby et al. 2007: pp. 10-11):

Military Monitoring inimical forces; monitoring friendly forces and equipment; military-theatre or battlefield surveillance; targeting; battle damage assessment; nuclear, biological, and chemical attack detection.

Environmental Micro-climates; forest fire detection; flood detection; precision agriculture.

Health Remote monitoring of physiological data; tracking and monitoring doctors and patients inside a hospital; drug administration; elderly assistance.

Home Home automation; instrumented environment; automated meter reading.

Commercial Environmental control in industrial and office buildings; inventory control; vehicle tracking and detection; traffic flow surveillance.

WSNs are one of the first real-world examples of *pervasive computing*, the notion that small, smart, inexpensive sensing and computing devices will soon permeate the environment. The widespread distribution and availability of small-scale sensors, actuators, and embedded processors offers an opportunity for transforming the physical world into a computing platform. Invisible computing is driven by advances in wireless technologies, WSNs, IP services, Internet, and VoIP technologies. Some claim that over the next decade, traffic from the edges of the network will be as heavy as that at present with traffic flowing from servers to clients (Sohraby et al. 2007: p. 50).

2.4 Challenges for Development

Handling such a wide variety of application types will scarcely be accomplished in any single realisation of a WSN. Nevertheless, a certain commonality is evident, especially with respect to the characteristics and the required mechanisms of such systems. Realizing these characteristics with new mechanisms is the major challenge of the vision for the evolution of wireless sensor networks. Karl & Willig (2005: pp. 7-9) has identified the following characteristic challenges as being shared among the good majority of example applications as presented in the previous section (see §2.3):

Type of Service The expectation that meaningful information and/or actions pertaining to a given task be provided not just the physical mechanism itself. In addition, the *scoping* of interactions to a specific geographic region or time interval.

Quality of Service Traditional measures become increasingly irrelevant in the context of applications tolerant to latency and minimal bandwidth usage. More pertinent is the degree to which the amount and quality of information pertaining to observed events in the region of interest are able to be extracted by the data sink.

Fault Tolerance In all likelihood, node failure, for whatever reason, is to be expected. Therefore, it is a necessity that redundancy be factored into both communications and node deployment.

Lifetime The network must operate for at least the given mission time or as long as possible. Limited energy supplies possibly augmented by renewable sources and due attention at all times to efficiency of operations.

Scalability The employed architectures and protocols must be resilient enough to scale to the requirements of a potentially large number of nodes.

Wide Range of Densities Spatio-temporal fluctuations in the presence of node failures, mobility or imperfect coverage lead to densities of considerable variation to which the network must be able to compensate.

Programmability Once deployed, nodes should be re-programmable in-the-field. Thence, being able to dynamically adapt to changes in task orientation. A fixed way of information processing is insufficient.

Maintainability As both the environment and the network itself change over time, the system has to incorporate the means to be self-monitoring as well as self-adaptable. Thus ensuring its extended operation at the required level of function.

To realize these in practice, innovative mechanisms for a communication network have to be found, as well as new architectures, and protocol concepts. A particular challenge here is the need to find mechanisms that are sufficiently specific to the idiosyncrasies of a given application to support the explicit quality of service, lifetime, and maintainability requirements (Karl & Willig 2005: p. 9).

2.5 Open Issues

Currently, we are in a phase in which the technology for implementing wireless sensor networks is relatively mature but many open-ended issues remain before they can be deployed on a large scale. Santi (2005: pp. 10-11) proposes the following as the main obstacles to such an end necessitating a more complete definition and treatment:

Energy Conservation Increasing miniaturisation, batteries of low capacity, impracticality or impossibility of replacing/recharging the energy source and the expectation of relatively long network lifetimes all serve to place considerable demands on energy efficiency.

Low-quality Communications Deployment regions are often not ideal e.g. harsh climates and extreme weather conditions. In such situations, radio communications may be extremely unreliable making the requested collective sensing task exceedingly difficult to carry out.

Operation in Hostile Environments The environment places further demands on the sensitivity of network protocols to faults and the resilience of the physical sensors themselves.

Resource-constrained Computation Protocols must aim to provide the desired QoS despite the scarce resources available to sensor networks.

Data Processing Consideration given to the data accuracy/resource consumption trade-off by offering different levels of compression/aggregation addressing the requirements of the observer in monitored events of the region of interest.

Ease-of-commercialisation Lacking Unless the mechanisms of a complete from-scratch development and implementation become more generally applicable to a wider array of applications, WSNs as a technological concept will be rendered economically infeasible.

Toward such an end, the standardisation community has been at great pains to achieve a measure of consolidation in the field of WSNs. The most notable effort in this regard is the IEEE 802.15.4 standard currently under development, which defines the PHY/MAC layer protocols for remote monitoring and control, as well as sensor network applications. The ZigBee Alliance is an industry consortium (currently comprising more than 100 members, representing 22 countries on four continents) with the goal of promoting the IEEE 802.15.4 standard and in so doing exemplifying the growing trend toward the direct means of addressing the practical issues faced (Santi 2005: p. 9).

3 TOPOLOGY CONTROL

3.1 A Definition

In order to arrive at an understanding of where the current work falls within the purview of the existing literature pertaining to the topic of 'topology control', it is seen as necessary to drill down to the specific details that are of direct relevance. As by necessity, the applied nature and thus specialised implementation of the proposed algorithm calls for a context in which to make its results meaningful and it is hoped that the following sections provide such that is sufficient for our purpose. In so doing, bridging the gap between the theoretical and that of the practical. In addition, given the relatively recent arrival of this area of research it may well be one not wholly familiar to the reader and its treatment then not entirely redundant.

Quite informally, *topology control* is the art of coordinating nodes' decisions regarding their transmitting ranges, in order to generate a network with the desired properties (e.g. connectivity) while reducing node energy consumption and/or increasing network capacity (Santi 2005: p. 30). The *topology of the network* is determined by those nodes and links that allow direct communication (Labrador & Wightman 2009: p. 66).

A more formal representation is by way of a Geometric Random Graph, $G = (V, E, r)$, where V is the set of vertices (nodes), E is the set of edges (links), and r is the radius of the transmission range of the nodes. Each and every vertex on V represents a wireless sensor device and its location determined in two-dimensional space by an associated geometric coordinate. A circle of radius r centred at that location circumscribes the subset of vertices with distance less than r respective to the node. The subset so formed constitutes those neighbouring nodes with which the node can establish a direct link (i.e. communication area). A formalised definition for the communication area is here presented in Equation (3.1):

$$C_r(x) = \{y : |x - y| < r\}, \quad x, y \in V \tag{3.1}$$

(Labrador & Wightman 2009: p. 66).

It should be noted that this is an idealised scenario where problems of environmental factors leading to possible errors and retransmission are not considered.

In order to allow for testing of the resultant graph's strongly-connected properties, links

are modelled as unidirectional meaning the existence of an edge (x,y) does in no way imply the existence of (y,x) and hence may or may not be symmetric or bidirectional. In so doing, the implication that all graphs are directed where the adjacent edge(s) of each respective vertex being independent of that of its adjacent node(s). The union of all such edge pairs comprises the set of edges E. The formal definition of which is presented in Equation (3.2):

$$E = \{(x,y,d) : y \in C_r(x) \wedge d = |x-y|\}, \quad x,y \in V \tag{3.2}$$

(Labrador & Wightman 2009: pp. 66-67).

What this signifies in practical terms is that subsequent to the localisation of the network, an associated metric is applied to each edge in that of a pair-wise Euclidean distance (irrespective of transmission range) between each respective node pair thus forming a weighted graph. This intermediate construct is then pruned taking into account the previously excluded condition of transmission range. In this instance, the pair-wise distance metric of the complete graph is symmetric (potential links) but that of the underlying range-determined subgraph (actual links), by the same logic as that above for edge-pairs, is not necessarily so.

Furthermore, it is worth noting that for the purpose of real-world applicability we assume a network deployed in a state of flux as opposed to that of a static scenario. Hence we consider the dynamic of mobility which necessitates phases of topology construction as well as that of periodic maintenance of its underlying structure to allow for pair-wise distance metric changes with ensuant modification of range-determined inter-nodal relationships.

The distinguishing characteristic that sets topology control apart from other methods whose stated objective, amongst other things, is to achieve effects of energy minimisation and/or an increased network capacity is its *network-wide perspective* (Santi 2005: p. 30). In other words, local choices made at the node-level such as the determination of a suitable transmit power level is always made with the goal in mind of achieving a certain global, network-wide property while not excluding the possibility of consideration of more localised factors (i.e. best-effort). Thus, by virtue of its limited node-wide perspective, an energy-efficient design of the wireless transceiver is not classifiable as an instance of topology control. The same can be argued of power control techniques, whose goal is the optimisation of a single wireless transmission, possibly by way of multi-hop routing, thereby constituting a channel-wide perspective (Santi 2005: pp. 30-31).

Having arrived at a definition of topology control it should be clear that it does not impose a constraint on the means by which a desired network-wide property is achieved. As a result, both centralised and distributed techniques can be classified as topology control according to our definition (Santi 2005: p. 31).

3.2 Motivations

3.2.1 Energy Conservation

It is well known that the energy profile of a typical wireless sensor node is marked by a considerable contribution by that of the radio component and its associated management processes. Once again, highlighting the paramount importance of the efficient use of the scarce energy resources available to ad hoc and sensor network nodes and the challenges and difficulties this poses for the designer.

Suppose that node u must send a packet to node v, which is at distance d (see Figure 3.1). At maximum power, node v is within u's transmitting range indicating that direct communication between u and v is possible. However, a node w in the region C circumscribed by the circle of diameter d that intersects both u and v is also in existence. Since $\delta(u,w) = d_1 < d$ and $\delta(v,w) = d_2 < d$, routing the packet using w as a relay is also possible. Which of the two alternatives is more desirable from the energy-consumption point-of-view (Santi 2005: p. 27)?

To be able to make a determination, specific reference must be made to applicable wireless channel and energy consumption models. For ease of deliberation, let us assume that the radio signal propagates according to the free-space model and that we are concerned only with the minimisation of its transmit power. With these assumptions in hand, the required power to send the message directly from u to v is proportional to d^2; in that of relaying the packet via node w, being proportional to $d_1^2 + d_2^2$ (Santi 2005: p. 27).

Consider the triangle $\widehat{uwv}$, and let γ be the angle opposite to side uv. By elementary geometry, we have:

$$d^2 = d_1^2 + d_2^2 - 2d_1 d_2 \cos\gamma \tag{3.3}$$

(Santi 2005: p. 27).

Since $w \in C$ implies that $\cos\gamma \leqslant 0$, we have that $d^2 \geqslant d_1^2 + d_2^2$. It follows that, *from the energy-consumption point-of-view, it is better to communicate using short, multi-hop paths*

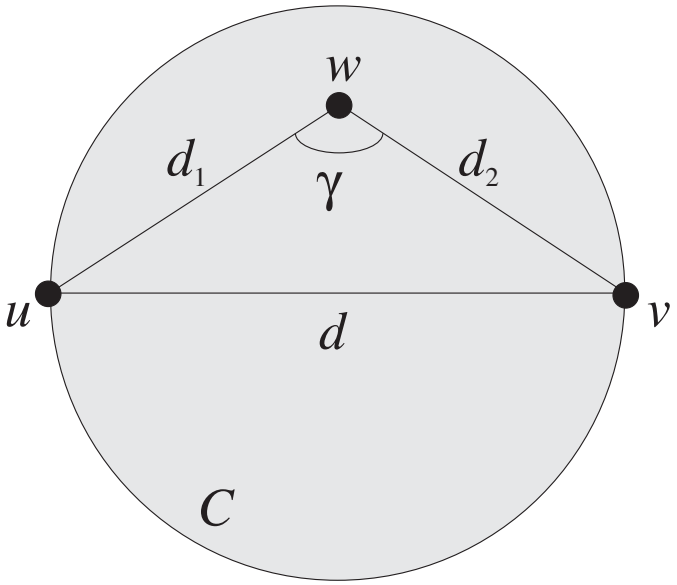

Figure 3.1: The case for multi-hop communication: node u must send a packet to v, which is at distance d; using the intermediate node w to relay u's packet is preferable from the energy consumption's point-of-view.

between the sender and the receiver (Santi 2005: p. 28).

The conclusion thus derived affords our first motivating factor in favour of topology control. Our goal then is to identify and 'remove' these energy-inefficient edges from the communication graph thus minimising their impact on the energy profile (Santi 2005: p. 28).

3.2.2 Increased Network Capacity

As opposed to wired networks distinguished by their fixed point-to-point channels, wireless communications utilise a shared medium, that of the radio channel. The use of which necessitates careful consideration be made of synchronising its access such that concurrent transmissions do not serve to corrupt each other and hence jeopardise reliable communication (Santi 2005: p. 28).

In light of this, further motivation for that of a topology control methodology is its collateral effect of a reduction in packet collisions at the data link layer and commensurate diminution in the number of retransmissions required with its undesirable additional inherent communication costs. It is somewhat counter-intuitive that the goal of a topological

construct should not be that it exhibit the property of high node degree as would be the case in a densely populated WSN with each node transmitting at maximum power (i.e. maxpower graph). Rather, that it should be tempered in light of the fact that such a scenario would be conducive to a higher probability of packet collision and a limitation of the possibility for frequency reuse (Labrador & Wightman 2009: pp. 63-64).

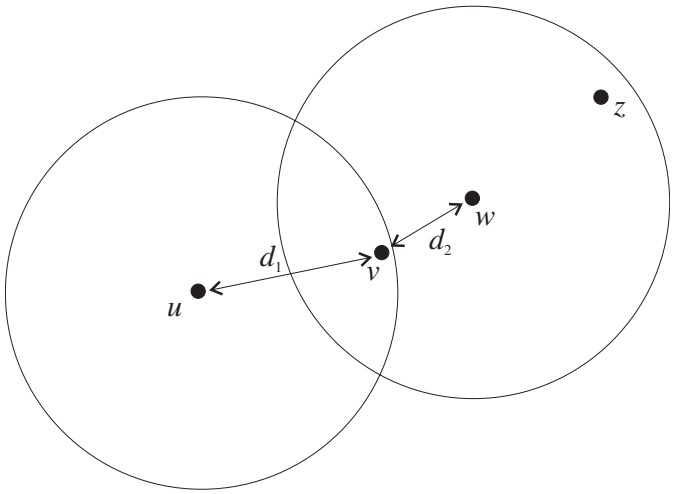

Figure 3.2: Conflicting wireless transmissions. The circles represent the radio coverage area with transmit power *P*.

The foregoing remarks, being intuitive, require refinement in order to establish a firm foundation or even to understand, in explicit terms, what is meant by the referent objective of an 'increased network capacity'. To illustrate its theoretical basis, let us consider a typical conflict scenario depicted in Figure 3.2 and elaborated upon in Santi (2005: pp. 28-30):

> *node u is transmitting a packet to node v using a certain transmit power P; at the same time, node w is sending a packet to node z using the same power P. Since* $\delta(v,w) = d_2 < \delta(v,u) = d_1$*, the power of the interfering signal received by v is higher than that of the intended transmission from u, and the reception of the packet sent by u is corrupted.*
>
> *Note that the amount of interference between concurrent transmissions is strictly related to the power used to transmit the messages. We clarify this important point with an example. Assume that node u must send a message to node v, which is experiencing a certain interference level I from other concurrent radio communications. For simplicity, we treat I as a received power level, and we assume that a packet sent to v can be correctly received only if the intensity of the received signal is at least*

$(1+\eta)I$, for some positive η. If the current transmit power P used by u is such that the received power at v is below $(1+\eta)I$, we can ensure correct message reception by increasing the transmit power to a certain value $P' > P$ such that the received power at v is above $(1+\eta)I$. This seems to indicate that increasing transmit power is a good choice to avoid packet drops due to interference. On the other hand, increasing the transmit power at u increases the level of interference experienced by the other nodes in u's surrounding. So, there is a trade-off between the 'local view' (u sending a packet to v) and the 'network view' (reduce the interference level in the whole network): in the former case, a high transmit power is desirable, while in the latter case, the transmit power should be as low as possible. The following question then arises: how should the transmit power be set, if the designer's goal is to maximise the network traffic carrying capacity?

In order to answer this question, we need an appropriate interference model. Maybe the simplest such model is the Protocol Model used in Gupta & Kumar (2000, as cited in Santi 2005: p. 29) to derive upper and lower bounds on the capacity of ad hoc networks. In this model, the packet transmitted by a certain node u to node v is correctly received if:

$$\delta(v,w) \geq (1+\eta)\delta(u,v) \tag{3.4}$$

for any other node w that is transmitting simultaneously, where $\eta > 0$ is a constant that depends on the features of the wireless transceiver. Thus, when a certain node is receiving a packet, all the nodes in its interference region *must remain silent in order for the packet to be correctly received. The interference region is a circle of radius $(1+\eta)\delta(u,v)$ (the* interference range*) centred at the receiver. In a sense, the area of the interference region measures the amount of wireless medium consumed by a certain communication; since concurrent non-conflicting communications occur only outside each other interference region, this is also a measure of the overall network capacity.*

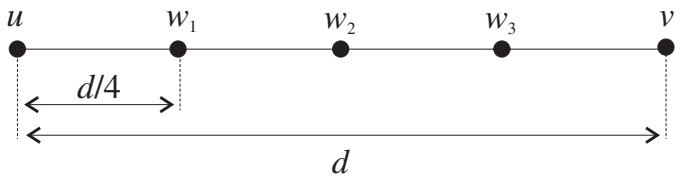

Figure 3.3: The case for multi-hop communication: node u must send a packet to v; using intermediate nodes $w_1, \ldots, w_3 = w_k$ is preferable from the network capacity point of view.

Suppose node u must transmit a packet to node v, which is at distance d. Furthermore, assume there are intermediate nodes $w_1, \ldots, w_k$ between u and v and that $\delta(u,w_1) = \delta(w_1,w_2) = \ldots = \delta(w_k,v) = \frac{d}{k+1}$ (see Figure 3.3). From the network capacity point of view, is it preferable to send the packet directly from u to v or to use the multi-hop path $w_1, w_2, \ldots, v$? This question can be easily answered by considering the interference range(s) in the two scenarios. In case of direct transmission, the interference range of

node v is $(1+\eta)d$*, corresponding to an interference region of area* $\pi d^2(1+\eta)^2$*. In case of multi-hop transmission, we have to sum the area of the interference regions of each short, single-hop transmission. The interference region for any such transmission is* $\pi(\frac{d}{k+1})^2(1+\eta)^2$*, and there are* $k+1$ *regions to consider overall. Since, by Holder's inequality, we have:*

$$\sum_{i=1}^{k+1}\left(\frac{d}{k+1}\right)^2 = (k+1)\left(\frac{d}{k+1}\right)^2 < \left(\sum_{i=1}^{k+1}\frac{d}{k+1}\right)^2 = d^2, \tag{3.5}$$

we can conclude that, from the network capacity point of view, it is better to communicate using short, multi-hop paths between the sender and the destination.

The observation above is the other motivating reason for a careful design of the network topology: instead of using long edges in the communication graph, we can use a multi-hop path composed of shorter edges that connects the endpoints of the long edge. Thus, the maxpower communication graph, that is, the graph obtained when the nodes transmit at maximum power, can be properly pruned in order to maintain only 'capacity-efficient' edges. The goal of topology control techniques is to identify and prune such edges.

3.3 A Taxonomy

As previously mentioned (see §3.1), an explicit definition of topology control does in no way limit the mechanism by which it is derived in practice. Therefore, Santi (2005: pp. 31-33) expounds a means for classification of the vying implementations; a taxonomy if you will. In like manner, affording the possibility to make a positive identification of the specific algorithm employed within the scope of the current thesis. In so doing, providing the framework in which to delineate the problem space and whose specification will be explored in the next chapter. Such formulation is a categorical imperative for each implies a difference of perspective and its concomitant determinative in arriving at a satisfactory solution.

The first-level of such a proposed hierarchy distinguishes between mechanisms on the basis of range:

- ***Homogeneous*** - where all nodes use the same transmitting range *r* known as the *critical transmitting range* (CTR), since using a range smaller than *r* would compromise the desired network-wide goal; and
- ***Non-homogeneous*** - the constraint is relaxed to allow differing ranges subject to the condition that the chosen range does not exceed the maximum range.

Non-homogeneous topology control is further sub-divided into three categories depending on the type of information utilised to compute the topology:

- Location-based;
- Direction-based; and
- Neighbour-based.

In *location-based* approaches, it is assumed that the most accurate information about node positions (the exact node location) is known. This information can be used by a centralised authority to compute a set of transmitting range assignments that optimises a certain measure. This is an instantiation of the *Range Assignment* (RA) problem and its variants as well as being indicative of the approach utilised in the current implementation; its taxonomic classification is highlighted, in *red*, within the visual representation of the nominal schema (see Figure 3.4).

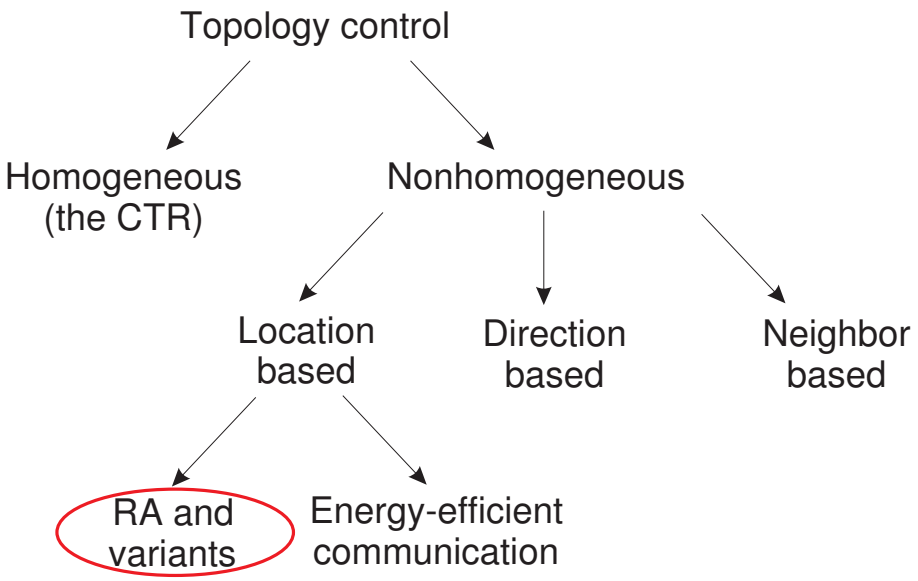

Figure 3.4: A taxonomy of topology control techniques.

4 CENTRALISED TRANSMISSION POWER CONTROL

As intimated in the preceding chapter (see §3.3), our approach is marked by being a centralised computation of the available data (i.e. location-based) and its reduction to a set of transmitting range assignments which elicit a certain network-wide property constituted as a whole e.g. strongly-connected and/or bi-connected.

4.1 Transmitting Range

The *transmitting range* of a node u denotes the range within which the data transmitted by u can be correctly received (see Figure 4.1). Given the range r, the definition of the subregion of R within which correct data reception is possible depends on the network dimension: in case of two-dimensional networks, it is the circle of radius r centred at u (Santi 2005: p. 17).

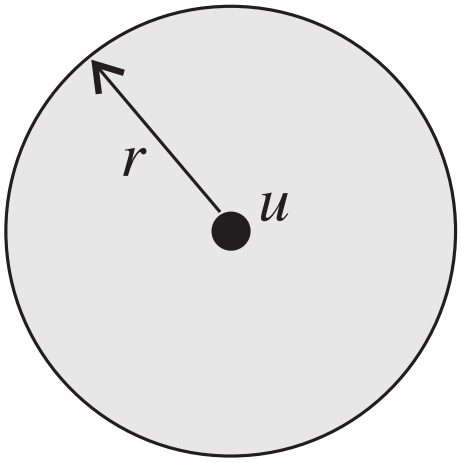

Figure 4.1: Radio coverage in two-dimensional networks. The covered region has radius r, and it is centred at the unit.

Note that, the facility for the provision of an assumption as to the nature of radio signal propagation and its reception are catered for by the possibility of further investigation in co-simulation studies (i.e. an extra degree of freedom) which are not elaborated upon here as its inclusion would inevitably result in undue expansion of the current scope envisaged of that of the present study. It is sufficient to state that, within these limitations, a probabilistic model underlies the simulation results herein later presented. Nevertheless,

a brief treatment of a suggested platform in which to conduct such an investigation is presented in the appendix (see Appendix B).

4.2 Problem Definition

A descriptive formulation of the problem space is posited now that the requisite condition of context has been met by the preceding discussion thus far as it relates to our area of interest.

4.2.1 Homogeneous - Critical Transmission Range (CTR)

The assumption that all the nodes use the same transmitting range reflects all those situations in which transceivers use the same technology and no transmit power control. This is the case, for instance, for most of the 802.11 wireless cards currently on the market. In this scenario, using the same transmitting range for all the nodes is a reasonable choice, and the only way to reduce energy consumption and increase capacity is to reduce r as much as possible (Narayanaswamy et al. 2002, as cited in Santi 2005: p. 40).

The following theorem shows that the CTR for connectivity equals the length of the longest edge of the Euclidean Minimum Spanning Tree (EMST) built on the network nodes (see §A.1 for a definition and its indicative correlate derivation):

> *Let N be a set of n nodes placed in $R = [0,l]^d$, with $d = 1, 2,$ or 3; representing the deployment region as a d-dimensional cube with side l. The CTR for connectivity r_C of the network composed of nodes in N equals the length of the longest edge of the EMST T built on the same set of nodes.*
>
> *(Santi 2005: pp. 39-40)*

This is but one variant of the many available problem formulations albeit a representative example that is easily derived and understood without recourse to extended treatment which, for our purposes, is sufficient to further the argument.

4.2.2 Non-Homogeneous - Range Assignment (RA)

> *Given the set N of network nodes, a range assignment for N is a function RA that assigns to every $u \in N$ a transmitting range $RA(u)$, with $0 < RA(u) \leqslant r_{max}$, where r_{max} is the maximum transmitting range and is dependent upon the features of the radio transceivers equipping the nodes. It is usually assumed that network nodes are equipped with transceivers having similar features, that is, r_{max} is the same for all the nodes in the network.*
>
> *(Santi 2005: pp. 17,73)*

The definition lends itself to an EMST implementation (in that it could be viewed as a special case of the CTR for connectivity problem with no restriction in its solution to a single global transmitting range) in situations where the capabilities of the nodes allow for dynamic transmission power control. In this case, however, the necessity for pruning the resulting construct is precluded. Just such an implementation is propounded later in the next chapter (see §5.5) that serves as a baseline for comparison to that of the primary focus; the application of a genetic algorithm (GA) for the selfsame purpose of deriving a solution to the RA problem.

4.3 Applicability of Approach to Application Domain

Even though the determination of range assignments, as applied here, are specifically of a non-homogeneous (RA) nature, it was worth examining the analogous case of homogeneity (CTR) in that it serves to highlight the rationale for its selection over that of its complement. In particular, the homogeneous solution, in all likelihood, may resemble a maxpower graph resulting in neither a significant change in topology nor energy savings. Hence, negating its efficacy as a means to satisfy the motivation for the use of the technique of topology control in the first instance. On the other hand, removing the constraint of homogeneity affords the prospect to a greater degree besides being much more amenable to the paradigm of multi-objective optimisation (MOO).

5 Optimisation Algorithms

As outlined in Appendix B, our basic implementation methodology, in effect abiding by 'A (Modified) Systematic Approach', is to harness the control design tools available within the Matlab/Simulink family of products in order to derive a workable solution to the RA problem (see Chapter 4). In this respect, graph theory is the fundament upon which any solution is to be based. However, it is evident that upon closer inspection there is a distinct lack of specific graph-theoretic capabilities within the basic tool-set of a Matlab/Simulink combination. Fortunately, it was discovered that a suitable library of routines addressing this deficiency was to be found.

5.1 MatlabBGL - A Graph Library

MatlabBGL is a Matlab package for working with graphs freely available on both the MathWorks Matlab Central File Exchange as well as at a dedicated website provided by the author, David Gleich (Research Fellow at Sandia National Laboratories in Livermore, CA). In essence, it implements a wrapper function for the *Boost Graph Library* (a representative member contained within the peer-reviewed and well regarded *Boost C++ Libraries* project which, as it turns out, has certain of its other libraries featured within the core framework of Matlab itself). That being the case, let us treat of its formal introduction as per Gleich (2007) in the accompanying documentation for the package.

The Boost Graph Library (Siek, Lee & Lumsdaine 2002) is a powerful graph analysis tool-kit that contains efficient algorithms implemented as generic C++ template specifications. Consequently, the primary purpose of the MatlabBGL library is to make these available via callable `mex` (MATLAB Executable) functions from within the Matlab environment. It was desirous that it be as seamless an extension to functionality as possible and therefore makes exclusive and direct use of the native Matlab sparse matrix data type to represent the adjacency matrix of a graph as its internal representation. The next section furnishes a concrete example of what this means in practice.

5.2 Graph Representation

To illustrate, consider an example digraph (see §3.1 regarding the reasoning and implication that all graphs, for our purposes, are directed) comprising six vertices and eight edges (see Figure 5.1). Note that the graph contains instances of both self-loops (at vertex x) and parallel edges (between vertices b and y). Hence, it is to be classified as a *multi-graph*.

On the contrary, a *simple graph* does not include either type of edge which will be the assumption as to the nature of all graphical representations in the course of the practical work; besides there being no provision for their handling within the library.

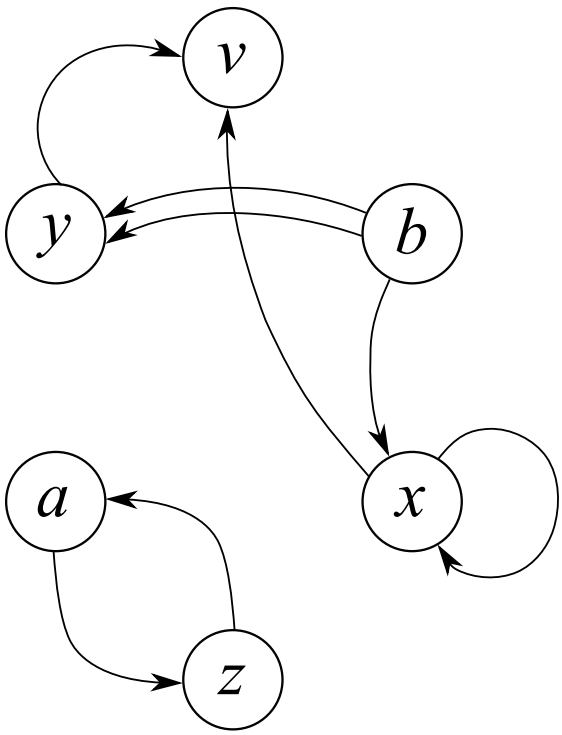

Figure 5.1: An example digraph (directed graph).

The equivalent adjacency matrix is as that presented in Equation (5.1) where the vertices are labelled: $a = 1$, $b = 2$, $v = 3$, $x = 4$, $y = 5$, $z = 6$ (i.e. row-as-source and column-as-destination node mapping in each respective edge-endpoint function). $A(i, j) = 1$ is indicative of there being an edge between vertices i and j whereas $A(2, 5) = 2$ the presence of a parallel edge. The non-zeros of the adjacency matrix define the edges and thus reflect the graphical structure thereof in an alternate form. If it should prove to be symmetric, then the graph is undirected. In general, any square sparse matrix in Matlab functions as a graph in the MatlabBGL library.

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{5.1}$$

To be further illustrative of how to construct this graph as a sparse matrix and its subsequent use within the library, the following set of commands obtain the desired result:

```
>> A = sparse(6,6);
>> A(1,6) = 1;
>> A(6,1) = 1;
>> A(2,4) = 1;
>> A(2,5) = 2;
>> A(4,3) = 1;
>> A(4,4) = 1;
>> A(5,3) = 1;
>> labels = {'a';'b';'v';'x';'y';'z'};
```

5.3 A Basis for Comparison

In a single-objective optimisation (SOO), there is one goal — the search for an optimum solution. Although the search space may contain a number of local optima, the intention is always to find the global optimum. In a SOO algorithm, as long as each iteration arrives at a better objective function value than previous, it is retained as being a viable solution (Deb 2001: p. 24). On the other hand, in multi-objective optimisation (MOO), as the name suggests, there are clearly multiple goals (i.e. more than one). Since MOO involves multiple objectives, it is intuitive to realise that single-objective optimisation is a degenerate case of multi-objective optimisation (Deb 2001: p. 1). Despite this, the trend has been in classical methodology to treat formulation of a multi-objective optimisation problem (MOOP) as a transformation and an extended application of the solution to a single-objective optimisation problem (SOOP). As a result, studies tend to concentrate on various means of converting multiple objectives into a single objective thereby avoiding the complexities of a true MOOP. It is true that theories and algorithms for SOO are applicable to the optimisation of the transformed single objective function. However, there is a fundamental difference between single and multi-objective optimisation which is ignored when using the transformation method. That being, apart from the obvious cardinality of objectives, in SOO there is but *one* solution, whereas in MOO, *several* solutions arise due to trade-offs between conflicting objectives (Deb 2001: pp. 3-4).

This also serves to highlight the fundamental difference between the focus of the present work and that of the vast majority of research as it relates to the topic of topology control. There exist many algorithms and application case studies but all from the perspective of SOO. In particular, I would like to draw attention to the application of the Euclidean Minimum Spanning Tree (EMST) algorithm as a characteristic example solution in the field (see §5.5). As a point of departure, it allows 'a basis for comparison' of its performance with that of the proposed Genetic and Evolutionary Algorithm (GEA) (see §5.6) in later simulation studies (see Chapter 8). Therefore, necessitating that a sensitivity to aforesaid differences inform the conception of the design of experiments (DOE) (see Chapter 7) in order that such a comparison be meaningful. More importantly, consideration as to the mechanism by which such a comparison is made possible, namely, by virtue of the fact that, by design, the proposed GEA is *multi-morphic*. That is to say, while still remaining true to its nature as an MOO algorithm, it is SOO/MOO-capable through consistent use of logical operators in its computation. Dependent upon model parameters that can be dynamically set by the user, the algorithm, acting accordingly upon either single or multiple objective functions in response. In either case, leveraging the unique characteristic of GEAs to find multiple optimal solutions in a single simulation run. In other words, the algorithm features capabilities of SOO when functioning upon a single objective. Yet, at the same time, it also possesses characteristics of MOO due its maintenance of an internal set of solutions.

It was desirous that the implementation be so capable for early in the investigation period, it was immediately apparent from an examination into the graph-theoretical basis of topology control techniques that there are many perspectives from which to draw a solution. Essentially, it is up to the designer to determine the desired properties to be exhibited by the resulting construct. Therefore, it was thought natural to incorporate a MOO approach to allow flexibility in choice of the functions in which to optimise for. Often, too, it is only after experimentation that a clear conception forms as to the requirements and how best to proceed. In many ways, it is very difficult to make unequivocal statements or predictions concerning 'typical' behaviour in a dynamic environment such as that in a WSN. This applies, especially so, with regard to graph-theoretical properties; quickly rendering the problem intractable. For these reasons, a strategy of multi-objective evolutionary algorithm (MOEA) computation in the solving of a MOOP is ideal for the purpose of not only the generation of a diverse set of optimal solutions but by the same token keeping one's options open accommodating promising new lines of inquiry with ease.

In that light, then, before we introduce the algorithms themselves, it is worth clarifying the

objective functions to be optimised and the decision variables upon which they act.

5.4 Objective Functions

In any optimisation, the *decision variables* should completely describe the decisions to be made. In our case, we define them as being the determination of the set of transmitting range assignments for the nodes constituting the network. In turn, the problem itself is some function of the decision variables. The function to be maximised or minimised, as the case may be, is called the *objective function* (Winston 1991: p. 52).

As such, we will now cover a complete formulation of the objective functions applicable to the operation of the proposed GEA. For convenience, we have created abbreviations for each to indicate when we are referring to a specific objective function as opposed to the general concept indicated by the non-abbreviated version. Note too, that given we are working within the context of a MOOP formulation, as per that given in the previous section (see §5.3), we will be using a similar notational convention as that in Appendix C which details the vital concept of pareto-optimality in sufficient depth for our purposes. Moreover, within each sub-section, we will adopt a fairly consistent order of presentation as well as a similarity in their respective content. First, we will treat of the theoretical bases for the objective function formulation. In particular, an examination as to the range of bounded values the graph property to be optimised is expected to assume as output by the operation of the objective function's calculation. Then, consideration as to its application as that implemented within the present work. In closing the section, subsequent to each objective function having been thoroughly explored, in turn, an explication will be given as to the unified frame of reference in which they ultimately abide.

N.B. Unless otherwise indicated, it is assumed we are given a *directed* graph $G = (V, E)$ with m edges (links) and n vertices (nodes) as input. In addition to which, given the set N of network nodes, a range assignment for N is a function RA that assigns to every $u \in N$ a transmitting range $RA(u)$, with $0 < RA(u) \leqslant r_{max}$, where r_{max} is the maximum transmitting range and is dependent upon the features of the radio transceivers equipping the nodes. For our purposes, network nodes are understood as being equipped with transceivers having similar features, that is, r_{max} is the same for all the nodes in the network. Our reader may recall that the latter condition is a reiteration of our working definition for the RA problem (see §4.2.2). Furthermore, as is typical in any SOO/MOO problem, the function to be optimised can be the subject of a number of constraints or bounds. In what follows, the implication being that the solitary bound is that conditioning the range of values $RA(u)$

may assume in accordance with the reasoning just put forward. Hence, their omission in the discussion pertaining to each respective objective function.

5.4.1 Connected Components (CC)

Fundamental to the premise of the present work, is the importance of a connectivity-based approach to the RA problem and its consequences for node deployment in the domain of application of a WSN. As such, Jungnickel (2008: p. 6) gives a good account of the relationship between the graph-theoretical properties of connectedness and connected components which help us to understand how to quantify such as an objective function:

> *Two vertices a and b of a graph G are called* connected *if there exists a walk with start vertex a and end vertex b. If all pairs of vertices of G are connected, G itself is called* connected*. For any vertex a, we consider* (a) *as a trivial walk of length* 0*, so that any vertex is connected with itself. Thus connectedness is an equivalence relation on the vertex set of G. The equivalence classes of this relation are called the connected components of G. Thus G is connected if and only if its vertex set V is its unique connected component. Components which contain only one vertex are also called* isolated vertices*.*

By a process of deduction, it is able to be established that the lower bound for the number of connected components in a graph is one. This result would be representative of a *connected* graph. At the other end of the scale, if we then assume a vertex set consisting of all isolated vertices, the upper bound on the number of connected components is n. This result is due to their consideration as being connected in and of themselves with an in-and-out degree of zero. Therefore, comprising a connected component of cardinality one. The sum of all connected components in the graph so formed is then n. In combination, then, we obtain the mapping of values from parameter-to-objective function space (viz. $F_{CC} : \Omega \rightarrow \Lambda_{CC}$) as a bounded interval:

$$\Lambda_{CC} = \{y \in \mathbb{Z}^+ | \, y \leq n, y = F_{CC}(x), x \in \Omega\}. \tag{5.2}$$

Furthermore, the formulation so derived implies the type of optimisation to be performed is a minimisation function:

$$\underset{x \in \Omega}{\text{minimise}} \, F_{CC}(x). \tag{5.3}$$

In an applied context, such is easily implemented by means of the direct application of the MatlabBGL library function `components`. Acting upon the non-zero structure of the input adjacency matrix representation, A_G, for the graph G, the function returns the number of connected components when called in the form `max(components(A))`.

5.4.2 Biconnected Components (BC)

Technically speaking, the term ‘biconnected components’ (BC), as used in the present work in its context as an objective function to optimise, is somewhat of a misnomer. In fact, we are minimising the number of strong articulation points in the graph. Nonetheless, it has been maintained as a naming convention for it highlights the intimate relationship between the concepts underlying both ‘connected components’ (see §5.4.1) and ’biconnected components’. Also too, it is perhaps better reflective of the desired effect and goal than would be the more accurate term of reference, ‘strong articulation points’, which is a somewhat more ambiguous usage in its intention without further clarification. Besides which, it is only after such explication that it becomes clear that it is a far stronger condition than would be strict optimisation for the number of biconnected components in a graph. In that concomitant to it, as a consequence, the objectives connected/biconnected components, both, are optimised. In addition, the number of strong articulation points in a graph provides a better measure and clear indicator of the desirability of a candidate solution in situations where, for instance, two graphs comprise the same number of biconnected components, but divergent in their numbers of strong articulation points. In such a case, the one exhibiting the minimal number of strong articulation points is the superior choice due to its greater tolerance to link failure even when the primary objective of *biconnectivity* (viz. by definition, there are no strong articulation points present in the corresponding graph) has not been achievable by execution of the algorithm. In short, then, what is meant, in objective function terms, by usage of the term *BC* is a minimisation in the number of strong articulation points and not, as is suggestive of at ‘face value’, its biconnected components.

Again, as done previously for ‘connected components’, it is worth making an examination as to the range of bounded values the graph property to be optimised is expected to assume as output by the operation of the objective function’s calculation. Further to the illustrative example presented in §5.8, it is readily apparent that the lower bound is zero. This result would be representative of a *biconnected* graph. Likewise, Italiano, Laura & Santaroni (2010: p. 160) specify the corresponding upper bound:

> *A directed graph can have at most n strong articulation points. This bound is realized by the graph consisting of a simple cycle: indeed in this graph each vertex is a strong articulation point.*

In combination, then, we obtain the mapping of values from parameter-to-objective func-

tion space (viz. $F_{BC} : \Omega \rightarrow \Lambda_{BC}$) as a bounded interval:

$$\Lambda_{BC} = \{y \in \mathbb{Z} \mid 0 \leq y \leq n, y = F_{BC}(x), x \in \Omega\}. \tag{5.4}$$

Furthermore, the formulation so derived implies the type of optimisation to be performed is a minimisation function:

$$\underset{x \in \Omega}{\text{minimise}}\, F_{BC}(x). \tag{5.5}$$

This, then, is the process informing the conception of the objective function for BC optimisation as detailed in §5.8, as referenced earlier, which contains its realisation in an applied context.

5.4.3 Global Energy Consumption (GEC)

The basic approach as to the modelling of energy consumption will be as that adopted in Santi (2005: p. 21):

> **Definition (Energy cost)** *Given a range assignment RA for a certain network ..., the energy cost of RA is defined as*
>
> $$c(RA) = \sum_{u \in N} RA(u)^{\alpha},$$
>
> *where α is the distance-power gradient.*

Santi further makes the point that the above definition of energy cost 'is coherent with our working assumption that the radio signal propagates according to the log-distance path model.' Such being the case, for our purposes, we will postulate a distance-power gradient exponent of two. Thus, being a value consistent with the selfsame model for a free-space environment scenario (Santi 2005: pp. 15-16).

This somewhat simplistic model is sufficient to characterise the relative merit of a particular candidate solution over that of another in terms of energy consumption. A position borne out by Santi (2005: p. 20) in that 'we are not interested in the absolute values of ... power consumption, but on [their] relative values.' It is worth bearing in mind that owing to its nature as an aggregate sum of the energy cost accorded a particular RA for the network as well as to differentiate it from the similarly derived objective function of 'local (per-node) energy consumption' (LEC) (see §5.4.4), we have nominated it as being a *global* measure of energy consumption.

It is a fairly straightforward matter to determine the extent of the objective function output values given that such a formulation for the energy cost of a RA is a direct function upon the values themselves. In the trivial circumstance where the whole of the network is comprised of nodes with their radio circuitry set to sleep mode, then each and every node has an explicit RA value of zero. Accordingly, the per-node energy cost will be zero as that of their accumulative sum. If we consider this then to be the degenerate case, we shall exclude zero as an endpoint in the interval. In so doing, we garner an intermediate result in that of a left-open unbounded interval. Given that the RA function assigns to each node comprising the network a transmitting range only on the condition that it not exceed the maximum capability of the equipped radio transceiver, r_{max}, we can surmise that the upper bound on the interval will be similarly conformed. In the event that each node is transmitting at full power, as determined by the RA function, $RA(u) = r_{max}$ for each u in N. This characteristic feature lends itself to a reduction of the energy cost function to that of nr_{max}^2 and, at the same time, an expression for its upper bound. In combination, then, we obtain the mapping of values from parameter-to-objective function space (viz. $F_{GEC} : \Omega \to \Lambda_{GEC}$) as a left-open unbounded interval:

$$\Lambda_{GEC} = \left\{ y \in \mathbb{R} \mid 0 < y \leq nr_{max}^2, y = F_{GEC}(x), x \in \Omega \right\}. \tag{5.6}$$

Furthermore, the formulation so derived implies the type of optimisation to be performed is a minimisation function:

$$\underset{x \in \Omega}{\text{minimise}}\ F_{GEC}(x). \tag{5.7}$$

As to its implementation, unlike that of the other objective functions thus far, the standard array manipulation operations within the Matlab environment suffice to bring about the desired result.

5.4.4 Local (Per-Node) Energy Consumption (LEC)

Whereas the objective function of ‘global energy consumption’ (GEC), as that treated in the previous section (see §5.4.3), has, as its perspective, a network-wide view, the present, on the other hand, adopts a local, or more precisely, node-centric view to the matter of energy consumption. Hence, its distinguishing nomenclature as that of ‘local (per-node) energy consumption’ (LEC). Despite which, there are many features common to both measures. This is in large part due to LEC being derived from the same energy cost function as that for GEC. In that capacity, it could well be considered not as entirely separate, but as a special case thereof. Although, in a way, it is characteristic of a node-centric measure, but in a network-wide sense. It is so for its stated purpose is to minimise the maximum

singular measure of energy consumption, as opposed to its aggregate. Such a mechanism is intended to render the overall spread of range assignments to be more uniform so that their extrema are less pronounced than would be otherwise. In other words, a kind of indirect means to obtaining the objective of energy expenditure minimalisation within the network. It also has the potential to effect an increase in the network lifetime because of its being able to distribute the energy cost associated with communication to more nodes, the likelihood of singular node failure is postponed. Stated differently, the intended result is that no particular node is allowed to 'bear the brunt' of the energy cost.

The situation is therefore much simplified as it pertains to an analysis of the expected range of values pertinent to LEC as an objective function considering no summation is involved. In effect, the energy cost function reduces to the form:

$$c(RA) = \max\{RA(u)^{\alpha} \mid u \in N, \alpha = 2\}. \tag{5.8}$$

The underlying basis being therefore essentially unchanged, the assumptions according, the left-open unbounded interval remains the same as before as does that of the upper bound; however, with the exception that the condition of multiplication by a factor n is not necessary for we are observant of a singular instance of range assignment only. In combination, then, we obtain the mapping of values from parameter-to-objective function space (viz. $F_{LEC} : \Omega \rightarrow \Lambda_{LEC}$) as a left-open unbounded interval:

$$\Lambda_{LEC} = \left\{ y \in \mathbb{R} \mid 0 < y \leq r_{max}^2, y = F_{LEC}(x), x \in \Omega \right\}. \tag{5.9}$$

Furthermore, the formulation so derived implies the type of optimisation to be performed is a minimisation function:

$$\underset{x \in \Omega}{\text{minimise}}\ F_{LEC}(x). \tag{5.10}$$

As to its implementation, unlike that of the other objective functions thus far, but like that immediately prior with GEC, the standard array manipulation operations within the Matlab environment suffice to bring about the desired result.

5.4.5 Node Degree (ND)

As an objective function, ND is supplemental to those precedent in their relative importance as judged by pair-wise comparison of the criteria utilising the AHP method as detailed in §5.7. Its presence then is justified by consideration of the desirability in obtaining a predetermined number of reference points in two-dimensional space (three in such a

circumstance) in order to facilitate the operation of localisation techniques upon nodes constituting a WSN.

Translating this requirement for successful triangulation into graph-theoretical terms means that the network-wide property of minimum degree, $\delta(G)$, be equal to or greater than the target value, δ^*. In particular, minimum in-degree, $\delta^-(G)$, and corresponding target value, δ^{-*}, given that, first, we are chiefly concerned with directed graphs, and, second, the successful reception of the transmitted beacon is the key requirement of localisation which necessitates that the node be within range (i.e. the recipient may or may not be able to relay a reply to the same beacon due to its own restrictions on maximum transmitting and/or current transmitting range assignment).

Such a problem definition, in terms of a goal or target value, lends itself to reformulation as an instance of the *goal attainment problem*. That being the case, although not immediately applicable without further manipulation, The MathWorks, Inc. (2009b) indicates a well-defined means of posing the problem in its general form:

> *Given a set of positive weights w_i, the* goal attainment problem *tries to find x to minimize the maximum of*
>
> $$\frac{F_i(x) - F_i^*}{w_i}. \ldots$$
>
> *If the F_i^* are positive, and you set all weights as $w_i = F_i^*$, the goal attainment problem becomes minimizing the relative difference between the functions $F_i(x)$ and the goals F_i^*.*
>
> *In other words, the goal attainment problem is to minimize a slack variable γ. ... This implies the expression that is the formal statement of the goal attainment problem:*
>
> $$\min_{x,\gamma} \gamma$$
>
> *such that $F(x) - w\gamma \leq F^*$.*

First and foremost, since the target value is a *minimum requirement*, reversal of the usual term of reference in optimisation to that of a maximisation function is a requisite condition. Of course, such would be productive of a meaningless result on its own (i.e. a complete graph at its logical conclusion), but it must be remembered that since we are working within the confines of a MOO-capable mechanism (see §5.3), conflicting objectives are acting concurrently on its operation. Thus, the concept of pareto-optimality comes to the fore in obtaining a solution (see Appendix C).

In order to establish the implications toward formulation as a goal attainment problem, in its general form as given, we must consider the range of values to be expected of the variables as parameters. In this regard, we will first make an examination as to the graph-theoretical property of minimum in-degree. As treated previously in §5.4.1, isolated vertices represent nodes disconnected from the rest of the graph. By virtue of that fact alone, we are able to deduce that the presence of any isolated vertices would be conducive of a minimum in-degree value of zero. On the other hand, a node connected with every other node would form $n-1$ such links. If all nodes within the graph are likewise connected, in addition to being a complete graph, the minimum in-degree would similarly be $n-1$. In combination, then, we thus arrive at the following range of possible values for the property of minimum in-degree:

$$\{z \in \mathbb{Z} \mid 0 \leq z \leq n-1, z = F_{ND}(x), x \in \Omega\}. \tag{5.11}$$

If we then consider the range of possible values to be expected of target values for the same parameter, we would anticipate that they should be of the same extent. However, to be deemed a reasonable choice, we would envisage that a target value of zero (viz. allowing for the presence of isolated vertices and consequent difficulties in localisation) to be invalid. Accordingly, we can express such reasoning in the following form:

$$\{\delta^{-*} \in \mathbb{Z}^{+} \mid \delta^{-*} \leq n-1\}. \tag{5.12}$$

Revisiting the goal attainment problem formulation of The MathWorks, Inc. (2009b) in light of the conclusions now reached; first, we are able to confirm that the goals (viz. δ^{-*}) are indeed positive (i.e. as per Equation (5.12)) and second, by setting the weights to be equal $w_i = F_i^* = \delta_i^{-*}$, the effect becomes the maximisation of the relative difference between actual and target values thereby. By substitution into its formal statement so constituted:

$$\begin{aligned} &\underset{x \in \Omega, \gamma}{\text{maximise}}\ \gamma \\ &\text{s.t.} \quad F_{ND}(x) - \delta_i^{-*}\gamma \leq \delta_i^{-*}. \end{aligned} \tag{5.13}$$

Less formally constituted, the corresponding output values for each possible pairing of combined input parameters (i.e. Equation (5.11) and Equation (5.12) as extrema in their respective bounded intervals) into that of our reformulated problem statement produces

the following four potential outcomes:

$$y = \frac{F_{ND}(x) - \delta^{-*}}{\delta^{-*}} = \begin{cases} -1 & \text{if } F_{ND}(x) = 0 \text{ and } \delta^{-*} = 1, \\ -1 & \text{if } F_{ND}(x) = 0 \text{ and } \delta^{-*} = n-1, \\ 0 & \text{if } F_{ND}(x) = n-1 \text{ and } \delta^{-*} = n-1, \\ n-2 & \text{if } F_{ND}(x) = n-1 \text{ and } \delta^{-*} = 1. \end{cases} \tag{5.14}$$

such that $F_{ND}(x) = [0..n-1]$ and $\delta^{-*} = [1..n-1]$ for some $x \in \Omega$. Since both $F_{ND}(x)$ and δ^{-*} are integers, their difference will also be such. If we then take the result as numerator with δ^{-*} as denominator, the quotient will always result in a rational number even for intermediate values of the parameters between the extrema of their respective bounded intervals. In combination, then, we obtain the mapping of values from parameter-to-objective function space (viz. $F_{ND} : \Omega \to \Lambda_{ND}$) as a bounded interval:

$$\Lambda_{ND} = \left\{ y \in \mathbb{Q} \mid -1 \leq y \leq n-2, y = \frac{F_{ND}(x) - \delta_i^{-*}}{\delta_i^{-*}} \right\} \tag{5.15}$$

where $F_{ND}(x) = [0..n-1]$ and $\delta_i^{-*} = [1..n-1]$ for some $x \in \Omega$. The output values of which can therefore be understood as the relative under-or-over achievement of the goal with respect to the target value set by the user according to the following interpretation:

$$y = \begin{cases} \textbf{under-target} & \text{if } y \text{ is } \textit{negative}, \\ \textbf{target achieved} & \text{if } y \text{ is } \textit{zero}, \\ \textbf{over-target} & \text{if } y \text{ is } \textit{positive}. \end{cases} \tag{5.16}$$

To be illustrative of how such would be utilised in an applied context, let us turn our attention now to the further development of the example digraph (see Figure 5.1) and its associated adjacency matrix representation, A_G (see Equation (5.1)), as that provided earlier in our discussion, in terms relative to our reformulation of the goal attainment problem. If we were to perform a summation of columns as per Equation (5.17):

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & 0 & 2 & 2 & 2 & 1 \end{pmatrix} \tag{5.17}$$

(viz. $\sum_{i=1}^{n} A_G[i,j]$, in turn, for each $j = 1,\ldots,n$), we would obtain the in-degree row vector for the graph G, $\mathbf{deg}^{-}(u) = [1\ 0\ 2\ 2\ 2\ 1]$ for each u in N. From this we can see that the minimum value is zero (viz. $\delta^{-}(A_G) = 0$). If our target value had been three (viz. $\delta^{-*} = 3$), then by substitution into $F_{ND}(x) - \delta^{-*}/\delta^{-*}$, we obtain $0 - 3/3 = -1$. Thus, by interpretation of the said result utilising Equation (5.16), the target has been under-achieved to the maximum extent possible (in accordance with Equation (5.15)) in this particular example. Obviously, this would be a result in which there is definite room for improvement. The optimisation attempts to bring this about by arriving at a combination of edges that satisfy the constraint that the goal be met or exceeded to the greatest degree achievable.

A further modification predicated upon our reformulation of the problem is as that caveat given in The MathWorks, Inc. (2009a):

> *Genetic Algorithm and Direct Search Toolbox optimization functions minimize the objective or fitness function. That is, they solve problems of the form*
>
> $$\min_x f(x).$$
>
> *If you want to maximize $f(x)$, you can do so by minimizing $-f(x)$, because the point at which the minimum of $-f(x)$ occurs is the same as the point at which the maximum of $f(x)$ occurs.*

The additive inverse for this particular example would therefore be, $-(-1) = 1$. Such would be the objective function value, as it appears, for an individual candidate solution that the proposed GEA endeavours to minimise. A corollary effect, too, is necessitated upon reversal of the relationships between output values and their interpretation as presented in Equation (5.16).

5.4.6 Towards A Unified Schema

Given the divergent aims and means of each of the objective functions acting within the realm of operation for that of the proposed GEA, it was envisaged that a more unified schema prevail in order that it be flexible enough to accommodate changes as and when they arise in the course of the practical work. Besides which, their interpretation, as raw output values, would otherwise be difficult at best when confronted with the complexities owing to formulation as a MOOP. The stated aim of this section, then, is to explicate the process by which this has been accomplished.

We begin with a summarisation of the range of values for the respective objective function and type of optimisation performed in Table 5.1. $F(x)$ is the vector of objectives, and,

x in Ω, the parameter-spaced values as input. As per the preceding discussion, n is the number of vertices and r_{max} is the maximum transmitting range for the nodes comprising the network. The shorthand convention utilised in the table is to indicate integer/rational-valued intervals as $[a..b]$, and real-valued ones by $[a,b]$, as per standard, for some a and b as endpoints in objective function space (viz. $a,b \in \Lambda_i : i = 1,\ldots,k$ — for the i^{th} objective in k objectives).

Table 5.1: Range of values for the respective objective function and type of optimisation.

$F_i(x)$	$y = F_i(x)$	$G_i(F_i(x))$
CC	$[1..n]$	min
BC	$[0..n]$	min
GEC	$(0, nr_{max}^2]$	min
LEC	$(0, r_{max}^2]$	min
ND	$[-1..n-2]$	max

An additional qualification upon the output values, as summarised in the table, is that of their transformation into an *unscaled* version of the goal attainment problem. We have already treated of it in its general form as that applicable to the objective function of ND in the previous section (see §5.4.5). Presently, we will again make reference to The MathWorks, Inc. (2009b) in the elaboration as to the details pertaining to reformulation as an instance of the unscaled goal attainment problem whose stated objective 'is to minimize the maximum of $F_i(x) - F_i^*$.' Additionally, 'If you set all weights [in the general form of the problem] equal to 1 (or any other positive constant), the goal attainment problem is [equivalent to] the unscaled goal attainment problem.'

Taking this then as our template, if we set the goal, F_i^*, for each and every objective function, $F_i(x)$, to be equal to its respective theoretical minimum value within the interval, Λ_i, and additionally set the weight, w_i, to be equal to the maximum value within the selfsame interval, we arrive at the following instantiation of the unscaled goal attainment problem:

$$y = \frac{F_i(x) - \min \Lambda_i}{\max \Lambda_i} \tag{5.18}$$

where $i = 1,\ldots,k-1$ (with the k^{th} objective being ND which has as its formulation that

contained in Equation (5.14)) for some $x \in \Omega$. By examination of the summarised table for the intervals of those objective functions under consideration, it is clear that the maxima (as weights) are indeed exclusively that of positive valued constants dependent only upon the dimension of the network, n, and the maximum transmitting range, r_{max}, that are both held to be invariable throughout the duration of the optimisation. Further too, since a network of dimension zero is the trivial case, such is not to be regarded as a factor for consideration. Hence, altogether, satisfying the condition of positivity in The MathWorks, Inc. (2009b).

For instance, $F_{CC}(x) - 1/n$, for some $x \in \Omega$, in the case of CC, would be a formulation of the problem in its *final form*; therefore, indicative of the *interim* values as they would appear to the user during computation of the fitness values for individuals within the population in the proposed GEA. They are interim values in the sense that they are subsequently evaluated according to additional ranking schema by a post-process decision criteria methodology (see §5.7) which makes a selection from among the set of optimal solutions achieving the best possible fitness value in the final generation at the time of reaching one of the stopping criteria for the algorithm. Nevertheless, in terms of raw values, they are the means by which the respective optimisation is effected and are therefore essential.

Such a strategy makes allowance for the possibility of comparison of the relative degree to which each objective has reached its implied target on a continuum between the best and worse case scenario according to the underlying analysis specific to each objective function developed in the preceding sections of which the table provides a systematic summary. By a process of virtual normalisation of output values, a consistency in their interpretation is realisable, independent of the operant node density or transmitting range, like that in Equation (5.16) but with modification due to their being marked by exclusivity as minimisation functions and exhibitant of non-negative values:

$$y = \begin{cases} \textbf{target achieved} & \text{if } y \text{ is } \textit{zero}, \\ \textbf{under-target} & \text{if } y \text{ is } \textit{positive}. \end{cases} \tag{5.19}$$

5.5 Euclidean Minimum Spanning Tree (EMST)

We have already encountered the topic of EMST previously in our discussion (see §4.2) in the context of possible formulations of the principal problems of CTR and RA with which we are concerned. This time, however, we venture to describe details regarding the implementation. In what follows, we begin with an analysis of the optimisation problem

and its implications as given in Sedgewick (2002: pp. 261-262). Then, we consider the practical derivation of such an approach. Note that clarification of graph-theoretical terms used in the discussion can be found in Appendix A.

Suppose that we are given N vertices and we want to find the minimal set of edges connecting them together. That is to say, this is the geometric problem of EMST determination. The most obvious way to solve it is to build a complete graph with N vertices and $N(N-1)/2$ edges — one edge connecting each pair of vertices weighted with the distance between the corresponding points. Then, we can use Prim's algorithm to find the EMST in time proportional to N^2. Given that the size of the input is proportional to N, such a solution is a *quadratic* algorithm for the problem. In other words, a computationally inefficient formulation as it discounts how the geometric structure renders the majority of edges in the complete graph irrelevant to the problem. Therefore, there is no need for their inclusion prior to construction of the EMST for the graph. Indeed, research suggests that it should be possible to find the EMST of N points in time proportional to $N \log N$. This is a direct consequence of two key properties pertaining to points in the plane. First, a graph known as the Delaunay triangulation (DT) contains the EMST as a subset of edges, by definition. Second, the DT is a planar graph whose number of edges is proportional to N. In principle, then, thereby computing the DT in time proportional to $N \log N$, ensuingly, running either Kruskal's algorithm or the priority-first search method to find the EMST, likewise, in time proportional to $N \log N$.

Taking this then as our strategy, it is possible to formulate a plan of action:

Phase 1 Subsequent to localisation of the network, the Matlab in-built function `delaunay` creates a DT of a set of points in two-dimensional space.

Phase 2 Indexing the triangulated edge-set thus formed into the corresponding geometric co-ordinates of its vertices, calculate the symmetric pair-wise Euclidean distance and label such as its edge weight.

Phase 3 The resultant non-zero structure as input to the MatlabBGL library function `mst`, computes the MST for an undirected graph whose default usage is that of Kruskal's algorithm without further parameter changes.

Phase 4 Determine the longest edge in the calculated MST. If necessary, prune the tree to conform with the constraint of maximum transmitting range capability of the nodes comprising the network and indicate to the user initiation of said remedial action by the algorithm.

Phase 5 Transmit the appropriate RA value as a control signal to the respective node affected.

It is clear, that this method in no way guarantees connectivity if the maximum transmitting range is smaller than the longest edge in the EMST. As such, it is a technique of 'best-effort' in practice.

The following graphs serve to illustrate by way of a visual representation as to the effect of each functional computation: DT computation (see Figure 5.2) and the resultant EMST itself (see Figure 5.3), respectively, as they would appear, if displayed, at each stage of the process.

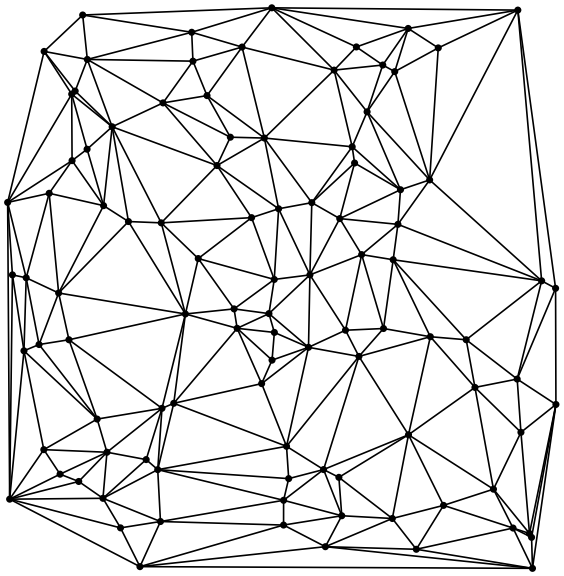

Figure 5.2: Delaunay triangulation (DT) of a random set of 100 points in the plane.

5.6 Genetic and Evolutionary Algorithm (GEA)

For all intents and purposes, the most important aspect in the proper execution of a GEA is in the articulation and formulation of the problem itself. This explains the depth with which we went to in an exploration of the objective functions applicable to its operation (see §5.4 — a section which could well be alternatively named 'Fitness Functions' instead of its present form as 'Objective Functions' and still be deemed as appropriate to its content).

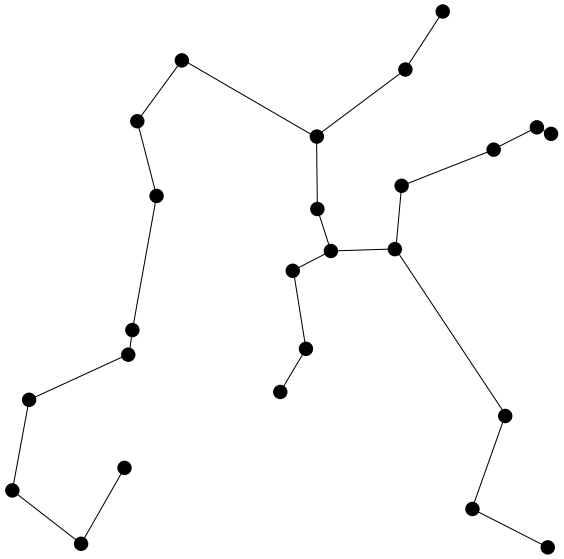

Figure 5.3: Euclidean Minimum Spanning Tree (EMST) of a random set of 25 points in the plane.

It only serves to highlight the fact that the present chapter is largely concerned with GEA in all its aspects from inception to culmination and not just those directly associated. This then is the basis in which the present section is dedicated, to wit, those components specific to GEA as a device in and of itself.

In the incipient stage of the practical component of the present work, it was thought practical to utilise the Genetic Algorithm and Direct Search (GADS) toolbox as the means to effect rather than resort to a third-party or one's own custom-versioned implementation external to the Matlab environment. Given the context in which we are working (see §5.3), the specific functional variety utilised is that of `gamultiobj`. Of which, The MathWorks, Inc. (2009a) introduces its operation in the following terms:

> *The* `gamultiobj` *solver attempts to create a set of Pareto optima for a multiobjective minimization. You may optionally set bounds and linear constraints on variables.* `gamultiobj` *uses the genetic algorithm for finding local Pareto optima. As in the* `ga` *function, you may specify an initial population, or have the solver generate one automatically.*

As we are predominantly invoking the function programmatically, we are able to set custom GEA parameters at run-time by passing an options structure as an input argument to `gamultiobj` which is created via a function call to `gaoptimset`. Otherwise, the default

values are assigned to the available options within the algorithm. Before we venture into the details as to their content, it is necessary to mention the executional logic underlying its mechanism according to our implementation.

Dependent upon model parameters that can be dynamically set by the user, the algorithm, acting accordingly upon either single or multiple objective functions in response. As a default condition, independent of other simultaneous objectives, it will automatically optimise for CC. In the event that a selection is made for BC optimisation, then the same applies but as a multi-stage computation. That is to say, the first-pass optimises for the number of connected components in the graph (see §5.4.1). Then, subsequently, the second-pass optimises for its number of strong articulation points (see §5.4.2 as well as that of §5.8). In addition to which, an intermediate stage ranks the resulting population of the first-pass achieving the best fitness value in the final generation at the time of reaching one of the stopping criteria for the GEA. It does so on the basis of a calculation of its associated number of strong articulation points. The elite individuals of which, as a subset of the final population, thereby constitute the specification for an initial population in the second-pass subject to the 'size of subpopulation' parameter (see Appendix D.1) in order that the number of solutions as such does not exceed the allowable limit. In other words, where the number of candidate solutions for an initial population is in excess of the limit imposed, we take into account only the number of individuals up to and including that limit as a truncated subset of those foremost in the intermediate ranking procedure. Otherwise, we simply utilise the whole population subset so formed as input considering the limit in such a case has not been violated.

This fundamental construct is reflective of the emphasis placed upon *connectivity-based* techniques of topology control within the present work. In the case where we are optimising for BC, the combined effect of the first-pass and intermediate ranking procedure is to act as a filter by reducing the problem to a consideration of those network configurations that exhibit connectivity in the highest degree, to wit, the number of connected components is one and its number of strong articulation points is zero — the lower bound on the range of output values expected of both measures — in the ideal case and would be thus representative of a *biconnected* graph which may or may not have been achieved in the first-pass by indirect means; hence, requiring the action of a further second-pass computational stage of the proposed GEA optimising for a different objective. Using the resulting pareto-optimal set of solutions as the nucleus for an initial population in the subsequent action of a second-pass, we thereby reduce the search space to such a degree

that the performance hit engendered by BC computation, in its present incarnation, is somewhat lessened. Primarily, then, a multi-stage GEA computation has been implemented for this very reason, as by direct observation of the impact upon the time of calculation in each iteration it was considered significant enough to warrant more careful attention as to its containment.

We have already made mention of a singular instance of a parameter applicable to the operation of the proposed GEA (i.e. size of subpopulation). As to which, and others besides, during the course of the practical work it was found that it was necessary to consider each individual stage separately with its own combination of specified parameters. This was for the reason that a single set of parameters common to both was not found to be in evidence as, more often than not, the optimal setting for a particular parameter in its respective stage tended to diverge from that of the other. In general, this is probably indicative of optimal parameter settings being highly specific to a given problem and are thus highly sensitive to any and all changes in their formulation, however slight they may be. Although not covered here, a detailed explication of those parameters utilised in each stage can be found in Appendix D which affords the possibility to make a comparison as to the essential similarities and/or differences between their usage. As presented, they are representative of values in their final form and were the result of extensive trial and error by experimentation as to the effect of each respective parameter, in isolation, upon the factors of timeliness and convergence toward a solution in particular. It should be noted too that there is a definite interplay between parameters and so careful attention must be paid to their setting in order that such not produce results contrary to that desired. That said, the GADS toolbox provides extensive facility for yet more options and parameters than that which we will treat of. Further to which, if so desired, the ability to seamlessly integrate one's own custom implementation of specific functional components within the computational process. Even so, those featured in the appendix are limited to such that seemed of particular relevance and impact to the specificities peculiar to the domain of application and problem definition with which we are concerned.

5.7 Decision Criteria

As succinctly suggested in the GADS User Guide (The MathWorks, Inc. 2009a), 'All solutions in a Pareto set are equally optimal; it is up to the designer to select a solution in the Pareto set depending on the application.' The same principle applies whether the proposed GEA is operating in either of its SOO/MOO capacities. This is the case owing to the fact that over the course of successive generations and calculations, the population

may or may not converge to an optimal solution (i.e. unique). A very likely scenario given the discontinuous nature of the properties of graphs for which we are optimising (e.g. the number of strongly connected components in a graph being 1.0 is a valid construction whereas one of a value 1.2 is meaningless and invalid). Therefore necessitating the selection of an individual that best fulfils the objective from the resulting population subset achieving the best possible fitness value in the final generation at the time of reaching one of the stopping criteria for the algorithm. In the circumstance of a true MOOP, the application of Multi-Criteria Decision Making (MCDM) methods are conducive to arriving at an informed decision as opposed to the technique of 'best guess'. This facilitates a ranking of alternatives that reflects the subjective preferences of the observer. In particular, we have made use of the Analytic Hierarchy Process (AHP) method of decision making. On the other hand, SOOPs call for a different approach: the heuristic principle of minimal range deviation (MRD). Although, both categories of decision criteria can be used in combination in the solving of MOOPs, especially, in cases where reduction via the AHP method does not yet attain the desired unique solution. In addition, once arriving at our decision, comparison is made of the individual's fitness value with that of the preceding estimation obtained in the last iteration of the algorithm which is currently in force as a range assignment for the network. If the fitness value is no better than previous, it is discarded in favour of retaining the current range assignment and awaits the next iteration of the algorithm. Let us then discuss these two techniques of decision criteria in more detail.

5.7.1 Analytic Hierarchy Process (AHP)

The AHP method allows decision makers to model a complex problem in a hierarchical structure showing the relationships between the goal, objectives (criteria), sub-objectives, and alternatives (Forman & Selly 2001: p. 43). For our purposes, we designate these components as:

Goal Obtain a single solution to the MOO-versioned RA problem.

Criteria A mapping of the simulation parameters available in the Simulink model of the system to the primary factors of connectivity, energy consumption, and node degree. Note that these are referred to as connectivity, power, and degree, respectively within that contained in Table 5.2 for the sake of brevity and space concerns. Also too, that they are indicative of the mutual exclusivity of selections and thus merged into one category in the case of connectivity and energy conservation factors: connected/bi-connected components and local/global energy consumption, respectively. Hence, to be considered as a whole relative to other decision criteria rather than as separate entities in themselves.

Alternatives The set of individuals achieving the best fitness value in the final generation at the time of reaching one of the stopping criteria for the GEA (i.e. a unique solution was not to be found).

Its purpose is to structure complexity and exercise judgement allowing for the application of data, experience, insight, and intuition in a logical and thorough way. Thus incorporating both objective and subjective considerations in the decision-making process. By means of a pair-wise *relative* comparison process, ratio scale weights or priorities are mathematically derived from a sequenced set of judgements. This is so because while it is difficult to justify weights that are arbitrarily assigned, it is relatively easy to justify judgements and their bases (i.e. objective data, knowledge, experience) as well as our natural proficiency at making relative rather than absolute judgements. The comparative judgement process can be performed using either words, numbers, or graphical means, and typically incorporates redundancy, which results in a measurement error reduction as well as a measure of consistency (Forman & Selly 2001: pp. 43-45).

Pair-wise comparisons are fundamental to AHP and according to Forman & Selly (2001: p. 63) a number of these taken together form a sort of 'average' calculated through a complex mathematical process using eigenvalues and eigenvectors, the results of which are very accurate. It was thought prudent to make use of an existent on-line calculator rather than derive a custom version of one's own in that nothing is to be gained by such a course of action. Such a mechanism was found in that contained on the Canadian Conservation Institute's (CCI) website (Canadian Conservation Institute 2005) with its 'Analytical Hierarchy Process (AHP) Program' tool. The instructions provided therein are exceptionally clear and serve to highlight the principal reasons for the adoption of AHP as a strategy for decision criteria in the present work:

> *This Web application is designed to assist in assessing and prioritizing the relative importance of various criteria when determining the best option for a defined problem. It is based on a mathematical model that was developed in the early 1970s by Thomas Saaty (Saaty, T.L., The Analytic Hierarchy Process, Pittsburgh: RWS Publication, 1990). Saaty's analytical hierarchy process includes and measures all tangible and intangible, quantitatively measurable and qualitative criteria, and calibrates each into a numerical scale. The process works by providing a sequence of pairs of criteria (i.e. each criterion is evaluated by comparing it with all the others in a sequence of pairings). For each pairing, participants are asked to rank, on a scale from -9 to +9, how important that criterion is compared with the other one. Continuing with this procedure, the mathematical model eventually gives a relative weight for each criterion, and the summation is normalized to 100%. The model also provides a*

consistency ratio, which shows how consistent the evaluation of the criteria was during the comparison. A low value (ideally below 0.10) is proof of good consistency.

Table 5.2 describes in tabular form the pair-wise comparisons performed in evaluating the relative importance of each of the criteria in the context of the proposed GEA. The numeric values of which correspond to the verbal descriptions given. In so doing, the tool calculated a consistency ratio of 0.266 and relative weights for the criteria as per Figure 5.4.

Table 5.2: Pair-wise relative comparison of identified design criteria importance evaluation.

Pair-Wise Comparison	***Value***	***Description***
Power ⇔ Degree	6.0	power moderately more important
Power ⇔ Connectivity	−5.0	power moderately less important
Degree ⇔ Connectivity	−6.0	degree moderately less important

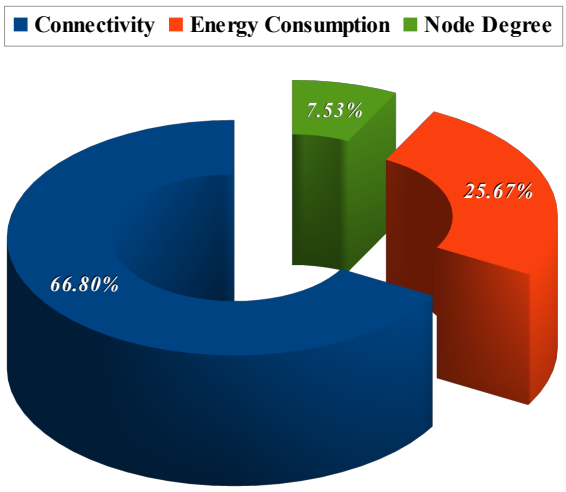

Figure 5.4: Multi-criteria relative preference weighting vector values derived via Analytic Hierarchy Process (AHP) method.

Thus equipped, it is possible to apply the weighting vector to each of the criteria forming an aggregate score for each individual comprising the pareto-optimal set of solution vectors. Subsequently, a further ranking can be performed and hopefully a unique solution is

obtained thereby. This does in no way imply that a weighting vector is to be utilised in the genetic computation itself as AHP is a post-process operation upon the resulting population.

The relatively high consistency ratio obtained from the pair-wise comparison can be explained as Forman & Selly (2001: p. 46) observe, 'The theory of AHP does *not* demand perfect consistency.' In addition to outlining four possible causes of inconsistency:

1. clerical error;
2. lack of information;
3. lack of concentration; and
4. the 'real world' is not always consistent.

it is elucidated how important it is that a low inconsistency *not* become the goal of the decision-making process. A low consistency ratio is necessary but not sufficient for a good decision, meaning, it is possible to be perfectly consistent but consistently wrong. It is more important to be accurate than consistent (Forman & Selly 2001: pp. 47-49).

5.7.2 Minimal Range Deviation (MRD)

By means of a process of code verification and consultation of the relevant associated Matlab documentation, it was able to be established with certainty that the particular sequence of calls within the implementation engenders an application of the typical textbook definition for standard deviation s of a data vector X (viz. the in-built function `std`) as per that in Equation (5.20):

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \quad \text{where} \quad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{5.20}$$

and n is the number of elements in the sample. The result s is the square root of an unbiased estimator of the variance of the population from which X is drawn, as long as X consists of independent, identically distributed samples. In this case, X is the element-wise difference between the previous set of range assignments for the network, currently in effect, and that of each instance of the local Pareto-optimal set of solutions calculated for the objective functions defined in the fitness function of the proposed GEA. As The MathWorks, Inc. (2009a) reminds us, 'All solutions in a Pareto set are equally optimal; it is up to the designer

to select a solution in the Pareto set depending on the application.' That being the case, the subsequent action then is to sort the resultant calculation (viz. the in-built function `sort` in its default 'ascending' order behaviour) thereby selecting our candidate solution as the one first in the order so derived which will best exhibit the property of *minimal range deviation* (MRD).

Utilisation of this heuristic principle as a decision criterion is desirable for a number of reasons. First, it helps to promote a measure of stability in the output values of the algorithm. Second, and perhaps more importantly, it coaxes the establishment of inter-nodal connections over successive iterations of the algorithm in borderline cases where the transmitting range required of the candidate solution of certain nodes is close to the maximum capability.

5.8 Special Considerations

In what follows, the presented lemmas, theorems, and algorithm all form a condensation of the key findings of Italiano et al. (2010) relevant to the present work. Subsequent to which, we will discuss their implications.

In essence, the current section endeavours to answer the question: when presented with a strongly connected graph (i.e. directed), how are we to determine the number of strong articulation points in an efficient manner? Let us consider this in conjunction with the example digraph provided (see Figure 5.5), to which we will make reference once the necessary context has been established in order to solve the problem.

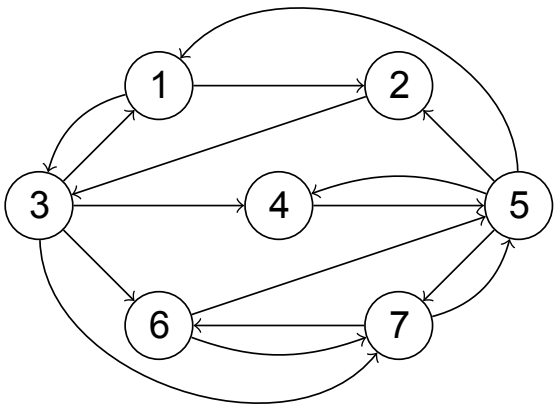

Figure 5.5: An example strongly connected graph.

Lemma 1. *Let $G = (V,E)$ be a strongly connected graph, and let $v \in V$ be a vertex of G. Then v is a strong articulation point in G if and only if there exist vertices x and y in G, $x \neq v$, $y \neq v$, such that all the paths from x to y in G contain vertex v.*

Lemma 2. *Let $G = (V,E)$ be a strongly connected graph, and let s be any vertex in G. Let $G(s) = (V,E,s)$ be the flowgraph with start vertex s. If u is a non-trivial dominator of a vertex v in $G(s)$, then u is a strong articulation point in G.*

Lemma 3. *Let $G = (V,E)$ be a strongly connected graph. If u is a strong articulation point in G, then there must be a vertex $s \in V$ such that u is a non-trivial dominator of a vertex v in the flowgraph $G(s) = (V,E,s)$.*

Lemma 4. *Let $G = (V,E)$ be a strongly connected graph and $G^R = (V,E^R)$ be its reversal graph. Vertex v is a strong articulation point in G if and only if v is a strong articulation point in G^R.*

Theorem 1. *Let $G = (V,E)$ be a strongly connected graph, and let $s \in V$ be any vertex in G. Then vertex $v \neq s$ is a strong articulation point in G if and only if $v \in D(s) \cup D^R(s)$.*

Algorithm StrongArticulationPoints(G)

Input : A strongly connected graph $G = (V,E)$, with n vertices and m edges.
Output : The strong articulation points of G.

1. Choose arbitrarily a vertex $s \in V$ in G, and test whether s is a strong articulation point in G. If s is an articulation point, output s.
2. Compute $D(s)$, the set of non-trivial dominators in the flowgraph $G(s) = (V,E,s)$.
3. Compute the reversal graph $G^R = (V,E^R)$.
4. Compute $D^R(s)$, the set of non-trivial dominators in the flowgraph $G^R(s) = (V,E^R,s)$.
5. Output $D(s) \cup D^R(s)$.

Theorem 2. *Algorithm* `StrongArticulationPoints` *computes the strong articulation points of a strongly connected graph G in time $O(m+n)$.*

Theorem 3. *The strong articulation points of a directed graph G can be computed in time $O(m+n)$.*

It is clear, that a careful exploitation of the relationship between strong articulation points and dominators yields a linear-time algorithm for computing the strong articulation points

of a directed graph. Lemmas 2 & 3 are necessary but not sufficient to arrive at such a formulation. Indeed, to compute all the strong articulation points of a strongly connected graph G, we would need to compute all the non-trivial dominators in the flowgraphs $G(s)$, for each vertex s in V. Since the dominators of a flowgraph can be computed in $O(m+n)$ time (Buchsbaum et al. 1998, as cited in Italiano et al. 2010) and there are exactly n flowgraphs to be considered, the running time of this algorithm is $O(n(m+n))$ (Italiano et al. 2010: p. 162).

This observation is indicative of the central misunderstanding of the author as it relates to an implementation of the findings in a practical context. The cited version of the research carried out by Italiano et al. is subsequent to an earlier preliminary one which was only made available after completing the practical component of the present work. As such, even though we have presented the findings in full, it must be remembered that that which informed the implementation was substantially different to what was the intention of the authors of the published version in which it is explicated in no uncertain terms. This is an attempt at explanation of the reasons as to why the actual implementation is not as efficient in its execution as it could be by virtue of being the result, in effect, of an apprehension of the findings up to and including Lemmas 2 & 3. Therefore, exhibiting a run-time proportional to $n(m+n)$. It is obvious, then, that the implication for further work consequent to that which has already been carried out is to utilise a more efficient version of the algorithm. It is likely that such would render some of the special processing requirements (see §5.6) virtually unnecessary or at the very least transformation of the problem into better tractability for the algorithm as it now stands may not prove to be very scalable. It is obvious, too, to the author, that such a modification would be relatively minor as well as being easily accommodated.

Nevertheless, we will give an explication as to how the algorithm for the calculation of the number of strong articulation points in a graph achieves its objective in an applied context as that implemented within the present work. Again, with reference to Lemmas 2 & 3, let us assume we are given the strongly connected graph $G = (V,E)$ in its form as an adjacency matrix, A_G (see §5.2):

Phase 1 Let $D(G) = \{\emptyset\}$ be an initialisation of the set of non-trivial dominators for the graph G.

Phase 2 Choose arbitrarily a vertex $s \in V$ in G.

Phase 3 The resultant non-zero structure of A_G as input to the MatlabBGL library function `lengauer_tarjan_dominator_tree`, returns the predecessor array, $p(i,j)$, for the

dominator tree, $DT(s)$, rooted at some vertex $s \in V$. By definition, a predecessor array does not contain elements as leaf nodes.

Phase 4 The Matlab in-built function `setdiff` performs a set-theoretic difference on the output (viz. $p(i,j) \setminus \{s,0\}$) to determine the set of non-trivial dominators, $D(s)$, for the flowgraph, $G(s) = (V,E,s)$, contained in $p(i,j)$.

Phase 5 Further, too, the Matlab in-built function `union` applies a set-theoretic union of sets $D(s)$ and $D(G)$. Formally, $D(s) \cup D(G)$.

Phase 6 Repeat preceding *Phases 2-through-5* until each vertex $s \in V$ in G has been traversed.

Phase 7 Application of the Matlab in-built function `numel`, returns the number of elements in the array thus formed — representing the cardinality of members of the set of non-trivial dominators for the graph G, $D(G)$. Formally, $|D(G)| = x$, for some value x. Thus, by Lemmas 2 & 3, also the number of strong articulation points in the graph G.

It is possible, now, to return to our example strongly connected graph (see Figure 5.5). If we select each node, in turn, as a starting vertex and thereby calculate their associated dominator tree of the flowgraph relative to the graph we arrive at the following set as shown in Figure 5.6. By examination, excluding the starting vertex and leaf nodes in each dominator tree so formed, as per the definition of a non-trivial dominator, we obtain the set of nodes $\{3,5\}$ which have been marked in bold within the figure. Therefore, we are able to ascertain the number of strong articulation points as being $|\{3,5\}| = 2$.

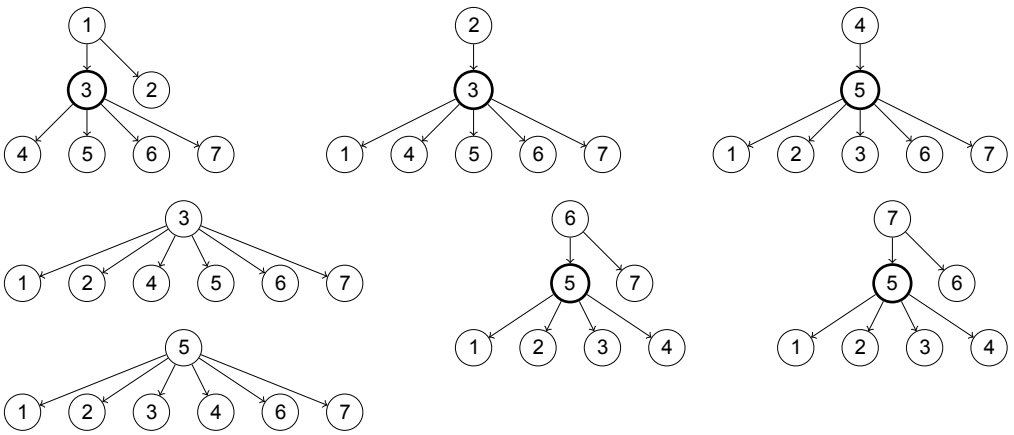

Figure 5.6: Dominator trees of the flowgraphs relative to the example strongly connected graph.

From this result, we can determine also that this particular example graph, although strongly connected, is not, at the same time, 2-connected (viz. biconnected or $|BC| = 1$). This is the case because for a graph to be 2-connected, it must contain no strong articulation points as, by definition, there exists at least two node-disjoint paths between any pair of distinct nodes in its graph. In other words, their existence and subsequent removal would render a graph's number of connected components to increase and if previously connected (viz. $|CC| = 1$) would now be disconnected (viz. $|CC| = 2$) as a result.

6 Simulation Scenario Design and Development

6.1 Mobility Model

From the early inception of the practical component of the present work, it was assumed as given that the domain of application dictates that it be as that in a dynamic context allowing for the freedom of movement of all nodes constituting the network. If we were to assume a static environment, on the other hand, we would be unduly overstating the relevance and credibility of any performance metrics so obtained. In the presence of mobility, questions as to how well the algorithm adapts to the changing circumstance from moment-to-moment arises. In short, is it robust enough to handle the demands placed upon it in a mobile environment? These are the questions which the simulation studies contained herein primarily attempts to reveal an answer to. Before all else, though, one must address how to approach modelling of the factor of mobility within the simulation itself.

That said, it was decided upon to utilise the Random Waypoint Mobility (RWM) model as the means to achieve the stated objective — the mobility model of choice within the mobile ad hoc network (MANET) research community of which many published results make exclusive use (Navidi, Camp & Bauer 2003: p. 1). First, it is necessary to expound upon what it constitutes in its standard form before we elaborate as to the particular variety utilised within the present work:

> *In this model, each node is assigned an initial location* (x_0, y_0)*, a destination* (x_1, y_1)*, and a speed S. The points* (x_0, y_0) *and* (x_1, y_1) *are chosen independently and uniformly on the region in which the nodes move. The speed is chosen uniformly on an interval* (v_0, v_1)*, independently of both the initial location and destination. After reaching the destination, a new destination is chosen from the uniform distribution, and a new speed is chosen uniformly on* (v_0, v_1)*, independently of all previous destinations and speeds. Nodes may pause upon reaching each destination, or they may immediately begin traveling to the next destination without pausing. If they pause, the pause times are chosen independently of speed and location.*
>
> *(Navidi & Camp 2004: p. 99)*

Navidi & Camp make the observation that there are some often unaddressed issues pertaining to the RWM model in its standard usage. They make the case for the need to sample the initial speed and location (and pause time, if applicable) from the stationary distribution rather than the uniform distribution. In addition to deriving the stationary distributions for speed, location, and pause time for a node moving in a rectangular area under the RWM model, Navidi & Camp provide an easily implemented procedure for

their sampling. The effect of such a procedure is in more efficient and reliable simulations in that convergence to stationarity is immediate and no data need be discarded — being the primary method of dealing with the initialisation problem (i.e. discarding an initial sequence of observations in the hope that the values observed past this initial sequence will have been sampled approximately from the stationary distribution) when sampling from the uniform distribution (Navidi & Camp 2004: p. 99). Otherwise, in most cases, the probability distributions of the initial locations and speeds of the nodes differs from the corresponding distributions at later points in the simulation. In fact, it is generally true that the probability distributions of both location and speed vary continuously over time, and converge to a 'steady-state' distribution, known in the probability literature as the *stationary* distribution. At any given point in the simulation, the distribution of location and speed is a weighted average of the initial distribution and the stationary distribution, with the weight shifting from the initial distribution to the stationary distribution as the simulation progresses (Navidi et al. 2003: p. 1). This is of particular relevance and importance when it comes to evaluating network performance metrics over the course of the simulation in that given the distributions of location and speed vary over time, as will, too, the associated performance thereof. In particular, network performance early in a simulation may differ substantially from its performance later in the same simulation (Navidi & Camp 2004: p. 99).

However, all these concerns are obviated if we were to sample from the stationary distribution at the very outset which will therefore be our strategy when it comes to a formulation of the means to mobility model implementation within the simulation scenario framework pertinent to our study.

6.2 Node Deployment

Further to the research findings of Navidi & Camp and Navidi et al., as that presented in the previous section, the Toilers Research Group at the Colorado School of Mines have freely made available to the research community an implementation of the steady-state random waypoint mobility model, `mobgen-ss`, for use within simulation studies. Navidi et al. (2003: p. 9) stress the importance of using `mobgen-ss`, especially when the minimum speed is small or when a non-zero pause time is used. This is true of how we have set its input parameters for the purposes of our simulation study, as we will introduce shortly, and is therefore a cautionary note of particular significance.

We have reproduced the `README` file packaged with the `mobgen-ss` software distribution

in Listing E.1 for reference purposes in that it succinctly describes the core functionality and methods of invoking the program that obviates the need to consult the cited references mentioned in its opening passage which nonetheless have been treated of in the previous section for the sake of completeness. In addition to which, we have reproduced the NS-2 compatible format output file generated by `mobgen-ss` in Listing E.2 utilising the input parameters as per that contained in the comment header which have been replicated in tabular form with additional comments as appropriate (see Table 6.1).

Table 6.1: `mobgen-ss` input parameters productive of an output mobility file in NS-2 compatible format subsequently pre-processed for use within Simulink as the basis for node deployment within the simulation area.

Parameter	***Value***
No. of Nodes	20
Simulation Area	$500\,\mathrm{m} \times 500\,\mathrm{m}$
Simulation Time	$150\,\mathrm{s}$
Speed Mean[a]	$10\,\mathrm{m/s}$
Speed Delta[b]	9.999
Pause Mean	$30\,\mathrm{s}$
Pause Delta[c]	30
Output Format	NS-2

[a] A mean speed of $10\,\mathrm{m/s}$ is equivalent to a fast sprint over $100\,\mathrm{m}$.
[b] Such a speed mean and delta produces an interval in the range $(0.001, 19.999)$.
[c] Such a pause mean and delta produces an interval in the range $(0, 60)$.

There are particular consequences for our use of the steady-state random waypoint mobility model within our simulations. One of which is that the simulation time need not be of relative long duration (i.e. $150\,\mathrm{s}$ in our case). This is due to the fact that the stationary distribution is already reflective of the *long-term* state of the mobility pattern of the network. A longer duration is simply not necessary and besides which would be an inefficient use of one's resources. Further to which, by selection of a speed/pause mean and delta productive of a diverse range of values, we are introducing a variable factor more in keeping with a natural pattern of movement. That is to say, at any one time, a proportion of nodes are stationary and the rest are in motion. In addition, for any two nodes in the same state of either being paused or moving, they are likely to be doing so for different lengths of time or speeds, respectively. The network will, therefore, not be marked by artificial states where

all nodes are in motion or in a state of pause, except by chance according to the stationary distribution. That being the case, a singular scenario has the potential to exhibit the whole gamut of behaviours, thereby obviating the need to manipulate parameters unduly. As a result, simplifying the process of scenario generation to a great extent.

This leaves node density as the primary factor determining network performance metrics in correlation with radio coverage as a function of the simulation area dimensions and the effective value of r_{max} governing the transmitting range of the nodes constitution. By keeping the simulation area dimensions constant, as we will do in the context of our formulation of the design of experiments (DOE) in the next chapter (see Chapter 7), it becomes essential that the number of nodes parameter be set to a value in congruence with that to be expectant within the domain of application. In that respect, we know from experience that WSNs, in general, are not constituent of a large number of nodes, but that of a reasonable amount only — enough to be sufficient for sensor coverage of the region of interest. For this reason, an input parameter setting of 20 for the number of nodes suggested itself to be of a value justifiable under the circumstances. That said, a greater network dimension would only serve to improve the performance metrics and, as such, the present value is more reflective of that which we could hope for as a baseline.

Additionally, we have given consideration to ensuring that the minimum speed, v_0, be set to a non-zero value — as can be verified by reference to the footnote within the table, where the effective speed interval for our scenario is in the range $(0.001, 19.999)$. For Navidi & Camp (2004: p. 100) note that, 'if ... taken to be zero, the mean node speed approaches zero.' Such a result would, therefore, not be in keeping with the design goal of evaluating performance in the presence of mobility. Hence, the need for the mean node speed to be maintained at non-zero levels throughout the duration by setting the parameter accordingly.

A further point is related to the available options in output format for the operation of `mobgen-ss` (i.e. NS-2, QualNet, and `gnuplot`). It is, of course, a necessary condition, whichever selection is ultimately made, for its transformation into one compatible for use within the Matlab/Simulink environment, of which is that relevant to the practical component of the present work. In our case, we select an output in conformance with the NS-2 mobility file format, possessing a knowledge of its implicit syntax and semantics, we then make use of Matlab in the manner of a text pre-processor, due to its extensive string function capability, in node localisation time series generation suitable for the purpose

of determining the mobility pattern scenario within the simulation. This is an efficient process for we need only specify the waypoints in the mobility file as co-ordinates and its associated time value, with which Simulink is able to perform an interpolation of intermediate points in-between as the simulation progresses at the granularity required of the specific solver utilised.

6.3 Simulink Model Parameters

It was thought advantageous to encapsulate all essential run-time parameters with which the simulation is dependent. Thus affording the possibility for dynamic adaptation to circumstances as required, such as we have done in the operation of the proposed GEA where a process of multi-stage computation performs an optimisation for a different objective function to that of previous at each stage, which would have proved impossible given a more static implementation (see §5.6 and Appendix D).

Of course, by design, this was primarily intended for that in a programmatic context, but we have also facilitated ease of use within an interactive session. This feature is demonstrable by double-clicking the optimiser subsystem block (see Figure F.7 for its associated component block diagram) contained within the Simulink root node of the Model Browser for the topology control optimisation simulation model (see Figure F.1), where we then obtain the block mask dialog box as that in Figure 6.1, which allows the user to dynamically set all essential run-time parameters interactively. As can be seen, there is a one-to-one mapping from the objective functions covered in §5.4 to the available options within the dialog box in the case of drop-down box algorithm selection of GA. In addition to which, one has the ability to select alternate algorithms of EMST as well as that of None (i.e. no optimisation — an option of particular relevance when it comes to the design of experiments covered in Chapter 7, which would otherwise be of no particular utility in normal usage).

On a side note, as a consequence of the discussion in §5.4.3 and §5.4.4, it is now understood that the mutual exclusivity of options for GEC & LEC optimisation, in its present form as an implementation, are not necessarily to be considered as being at opposing ends of the scale. Hence, does not preclude the possibility of simultaneous optimisation of both objectives, which could be of benefit by not only minimising the global energy consumption, but that too of the maximum singular instance of energy consumption from the local (per-node) perspective. As such, would be productive of a more energy-efficient usage as well as concurrent important effects as a load balancing agent. In so doing,

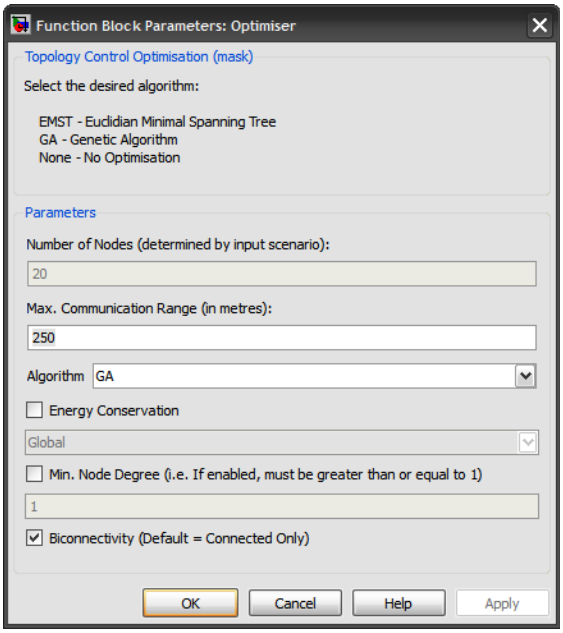

Figure 6.1: Optimiser subsystem component block mask dialog box allowing the user to dynamically set all essential simulation model run-time parameters.

extending the potential lifetime of the network considerably. This is a matter, though, for possible avenues in a furtherance of the present work in a future iteration.

6.4 Communication Graph

Once the parameters and options have been set according to the user's wishes, it is possible to visually track the progress of the simulation as it unfolds in real-time as that in Figures 6.2 & 6.3, which are furnished as example instances. All else being equal, for the same combination of parameters, excepting that of GA or EMST as chosen optimisation algorithm, respectively. Given both figures and associated subfigures are at the same time instant (i.e. $t = 150\,\text{s}$, indicating the end of the simulation as per Table 6.1) in addition to their utilisation of the exact same effective value of r_{max} (i.e. Communication Range = 240 in each individual figure), it is possible to make a comparison of the state of the network according to the corresponding algorithm used in its derivation.

First, it is important to note the essential differences between the two subfigures, contained within each figure similarly, in both Figures 6.2 & 6.3. On the one hand, we have the LHS subfigure representing the state of the network at the time of the last scheduled topology maintenance (TM) update (i.e. set to occur every 10 s of simulation time duration), which

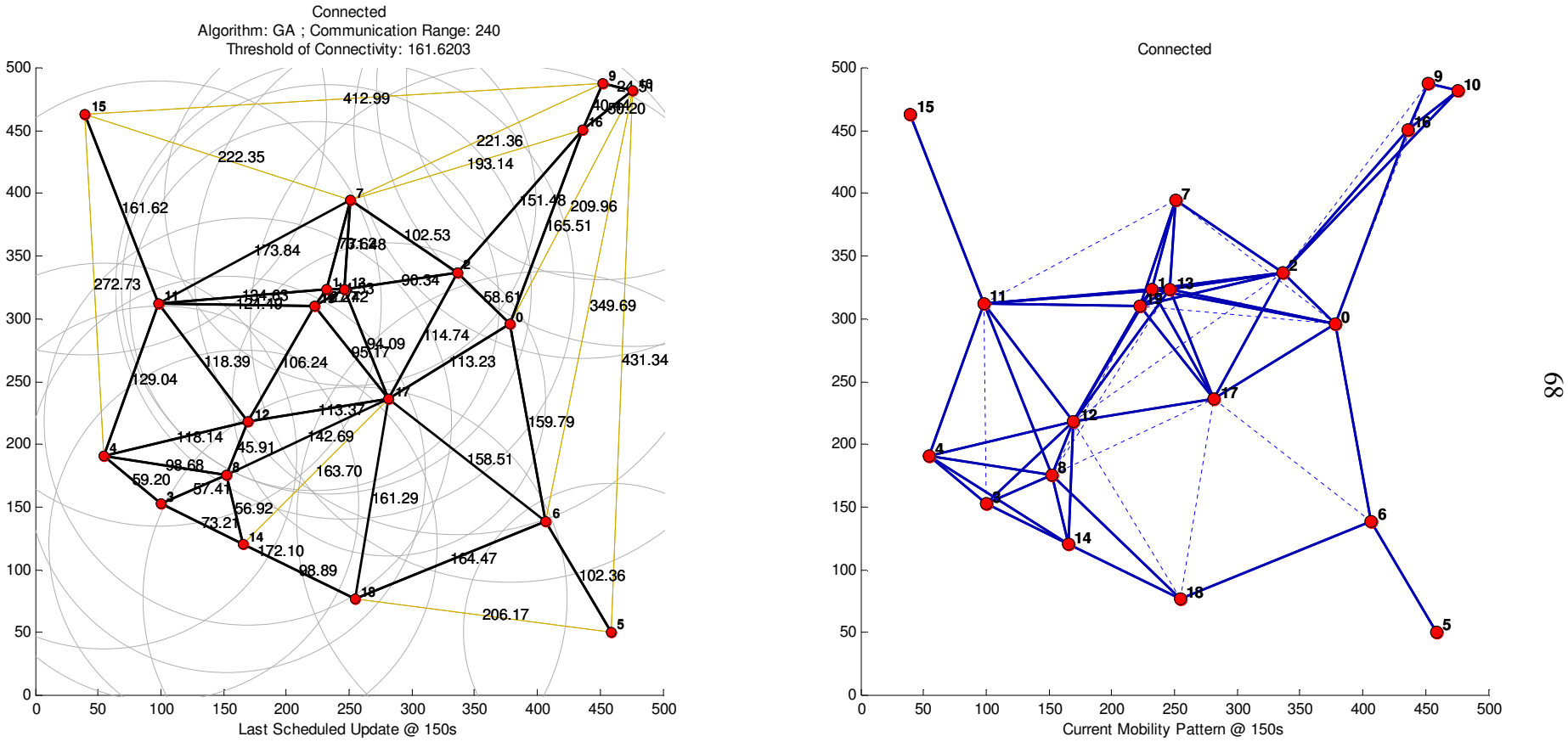

Figure 6.2: State of the network visualisation for an in-progress simulation with a Simulink model algorithm parameter set to GA.

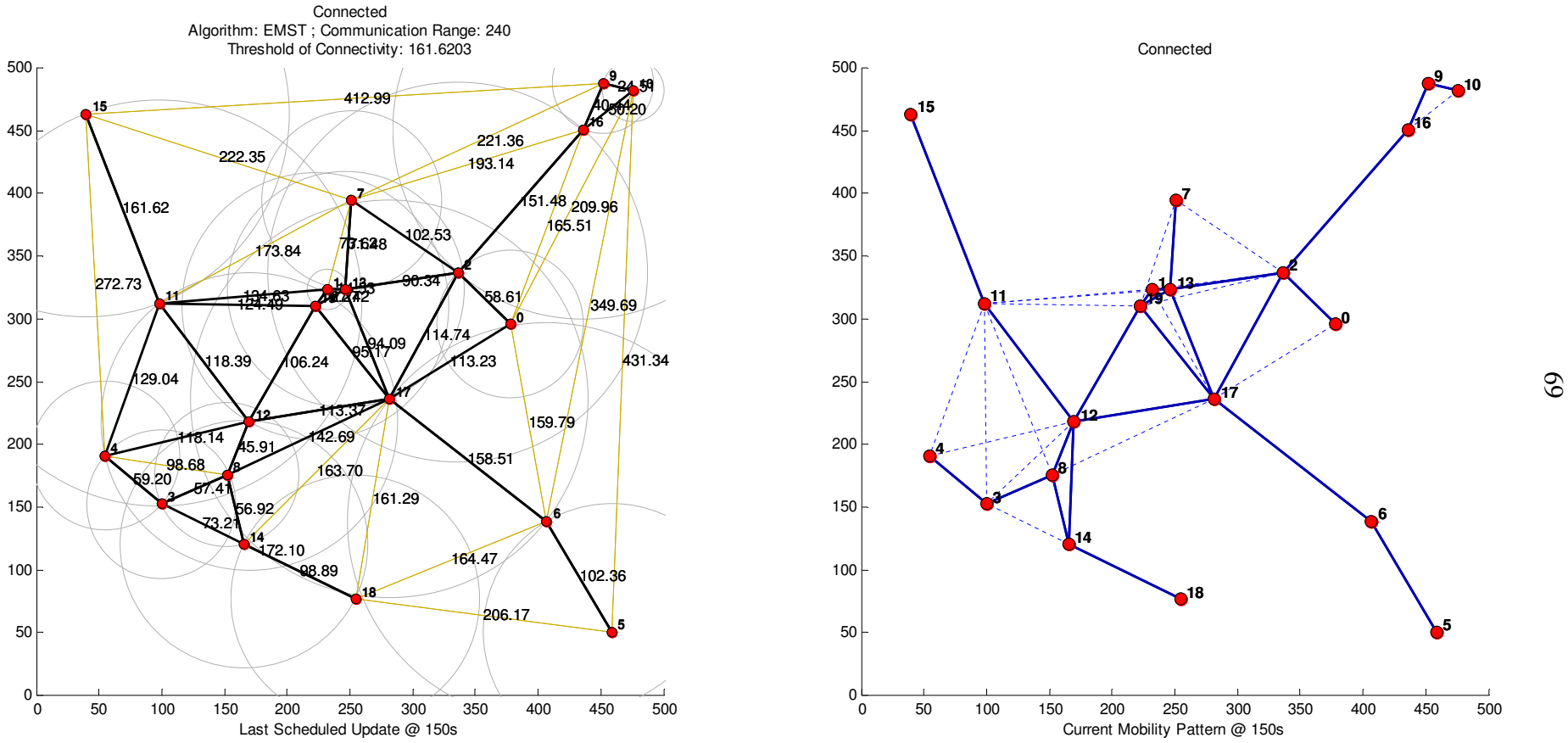

Figure 6.3: State of the network visualisation for an in-progress simulation with a Simulink model algorithm parameter set to EMST.

can be likened to a snapshot that remains fixed until the appropriate interval has elapsed, signalling the next scheduled TM update. On the other hand, the RHS subfigure is the current mobility pattern exhibited by the network with each node conforming to its respective transmitting range assignment as that determined by the most recent TM update. Its graphical representation is continuously refreshed and is therefore dynamic as opposed to the static nature of the adjacent subfigure. Since the time instants for both subfigures coincide in each figure, certain information contained therein may appear to be redundant. Though, over a period of time removed from that of the last optimisation, the presented information may also tend to diverge given the presence of mobility and other determining factors within that same interval. By providing alternate views on the same network, each subfigure is designed with the aim in mind of rendering different but complementary sets of information. Thus, utilising the limited available screen real estate in an efficient manner.

Let us turn our attention now to an examination of the wealth of information encapsulated within each subfigure that, once understood, can be quickly deduced from mere casual inspection. That is, with the exception of the threshold of connectivity (ToC) property, which will be left as a discussion for §7.2.1, specifically, and Chapter 7, in general.

A feature common to both subfigures is the connected status property for the represented graph. Its value is derived by a call to the MatlabBGL library function, `dfs`, using the non-zero structure of the adjacency matrix representation, A_G, for the graph, G, as input. In turn, each vertex constituting the graph is arbitrarily chosen as a starting vertex, whereupon a simple depth-first search (DFS) is performed. If any vertex, other than the starting vertex, should prove unreachable, further traversal of the graph is precluded and its connected status automatically set to 'Disconnected'. Otherwise, if all starting vertices have been successful in reaching all other vertices within the graph, only in that instance will it be understood as being 'Connected'. That being the case, in addition to our exclusive use of the *directed* form of a graph within the present work, the connected status property is therefore a measure of whether or not the graph so represented is *strongly connected*.

Exclusive to the LHS subfigure, is that of a set of circles, in *medium grey*, centred at each respective node of radius $RA(u)$ where $u \in N$ and $G = (N, E)$. Thus, representing the radio coverage unique to that node in two-dimensional space, like that in Figure 4.1, and thereby encapsulating, in visual form, the function of the optimisation algorithm as productive of a set of transmitting range assignments (see §4.2.2).

Technically speaking, the LHS subfigure is not a communication graph in the strictest sense, rather, it is a Delaunay triangulation (DT) of the localised network at the time moment of the most recent TM update (see Appendix A.3 for their formal definition as graph-theoretical constructs, namely, that of Communication Graph & Delaunay Triangulation). The graph so formed is comprised of the same set of vertices as would that of the communication graph, but with a set of edges entirely composed of those of its associated DT for such a localisation, irrespective of the actual per-node transmitting range assignment. As such, necessarily, partitioned as those links inside and outside range, relative to the node, and that designated within the subfigure as lines of *black* and *deep gold*, respectively. The edge weights of which are the inter-nodal Euclidean distance overlaid as text at the edge's center point. Given that the nearest neighbour graph (NNG) is a subgraph of the DT, such a graphical device has the effect of providing the user a visual indicator of (relative to each node considered individually): the distance to its nearest neighbours, in addition to, those of which are within its currently effective transmitting range (i.e. actual communication links).

Therefore, in combination with visualisation of per-node radio coverage, as that already outlined, it is possible to quickly gauge the relative distance between nodes as well as their absolute values. This is of particular utility in situations where one needs to evaluate problematic communication links that prevent connected status of the network. In such a case, on the basis of visual indicators, one can adjust r_{max} to a sufficient degree appropriate for the purpose of rendering the link active, which value can be established by immediate feedback as to the effect of such a change in parameter setting.

The RHS subfigure, on the other hand, is a true communication graph and thus reflects the current state of the network at each time instant, including intermediate stages in-between consecutive optimisations. As will be evident, by inspection, the set of edges has been partitioned into two separate subsets. The first of which is composed of asymmetrical or unidirectional links, indicated within the subfigure by dashed, medium-weighted lines of *light blue* in colour. The remainder comprise the second subset, the communication graph's symmetrical or bidirectional links, which are indicated by those solid, bold-weighted lines shown in *dark blue*. In other words, we have distinguished the type of edge via visual cues of line style, thickness, and colour. It should be noted that the direction of the edge itself, as would be usual in a digraph representation instance, is not made explicit. Although, that being said, the direction can be established, if so desired, with some small ingenuity in a comparison of the communication link of interest's endpoints with that of their associated

per-node radio coverage in the adjacent subfigure. By design, this feature was primarily due to working within the limitations of the graphical drawing routines available to the user in a Matlab environment. Besides which, indicating the direction in each and every instance would be of doubtful practicality in situations where the number of nodes is of such a degree as to make distinguishing specific detail exceptionally difficult within what would be a morass. As such, the informational device, in its present form, serves its purpose without proving to be too much of a distraction.

We are now in a position to utilise our knowledge of the visual semantics contained within each individual subfigure in that of a comparison of the two algorithms utilised in the derivation of Figures 6.2 & 6.3, in purely graphical terms. The primary difference between corresponding subfigures is in the total number of constructed edges, both symmetric and asymmetric, in the case of BC optimisation for the proposed GEA over that of EMST. This concurs with our theoretical understanding of the EMST as being, by definition, a spanning tree of the graph, G, doing so at *minimum cost* in terms of edges weighted with their inter-nodal Euclidean distance. In order to establish a measure of robustness, *biconnectivity* requires that additional links be established over and above that of the minimal subset of possible edges required for mere *strongly connected* status. This observation will be a key point of understanding relative algorithmic performance, when it comes to an analysis of the simulation results to be presented in Chapter 8.

7 DESIGN OF EXPERIMENTS

7.1 Design Statement

The DOE, as presented herein, endeavours to establish, by means of simulation for a representative set of alternate topology control optimisation algorithms, an impression of the behaviour over time of the interaction between: the observed measure of a network's connectivity, namely, that of the *connectivity-time ratio*, being simply the proportion of time the network exhibited the property of strong-connectedness during the simulation's duration as a ratio percentage; the effective *maximum transmitting range*, r_{max}, as that characteristic property of the radio transceiver equipping nodes constituting the network assuming homogeneous specification thereof; and the *threshold of connectivity*, as that taken to be the mobility scenario under observation's minimum transmitting range requirement for sustainability of a network's strongly connected status throughout the simulation time's duration. In short, we attempt to answer the question:

> *Is an r_{max} value equal to or in excess of the threshold of connectivity, in addition to being necessary, a sufficient condition to establish connectivity throughout the duration?*

7.2 Data Representation

Fundamental to our realisation for the DOE is the concept of the threshold of connectivity (ToC), already encountered in the previous chapter (see §6.4), but only alluded to as it was left for the present discussion to make a formal definition of its terms. This, in combination with the connectivity-time ratio (CtR), also, too, yet to be defined formally, establishes the quantifiable bases in which the relative algorithmic performance will be predicated upon for the simulation results to be presented in Chapter 8.

N.B. For the sake of convenience, when referring to specific values of ToC or CtR we will denote each instance with the mathematical shorthand convention of c_t and r_c, respectively.

7.2.1 Threshold of Connectivity (ToC)

During the course of the practical component of the present work it was found, rather indirectly, that the EMST could be utilised as a device to derive a measure of the minimum requirements for establishing connectivity throughout the duration of the particular simulation scenario in question — a metric for which we have coined the term, threshold of connectivity (ToC). By utilising this property as the pivot point about which manipulation of the r_{max} value is based, we are able to obtain a picture as to the sensitivity with which

connectivity is tied to the particular combination of values for each. As a quantity, it is specific to the mobility pattern under consideration and requires that it be calculated anew in each instance thereof, which explains the careful attention paid to ensuring the scenario itself as used herein, having as its essential parameters those of Table 6.1, is representative of that in the domain of application (see §6.2).

Its utility comes as a consequence of the observation already made as to how construction of an EMST for a graph in no way guarantees connectivity if the currently effective value for the maximum transmitting range, r_{max}, is less than that of the longest edge so formed. If we were to then compute the EMST, regardless of the actual topology control optimisation algorithm under observation for the particular simulation, in conjunction with whatever processing is required, we would thereby determine the theoretical lower bound on the value of r_{max} necessary to establish connectivity throughout the duration. That is to say, the ToC is a measure of the longest edge in the EMST encountered thus far and its final value at the end of the simulation is our characteristic connectivity-based property for the mobility scenario in question. Given its efficient implementation as an algorithm (see §5.5) and a mechanism was already in place for visualisation of the state of the network (see §6.4), it was a somewhat trivial matter to incorporate an additional computation without significant effect in overall performance.

With that in mind, using the example as that presented in Figures 6.2 & 6.3 for two separate algorithms — GA and EMST, respectively — with all other parameters being equal, including that of the exact same mobility scenario, we can see that $c_t = 161.6203$ at simulation time, $t = 150\,\text{s}$. Further too, an r_{max} value of 240 for each algorithm was sufficient to produce a strongly connected graph, in each instance. Although, this gives no indication as to how the connected status property varies over time by virtue of being but a momentary observation of its value. Therefore, necessitating an additional means of quantifying the same, which will be the subject of the next section.

7.2.2 Connectivity-Time Ratio (CtR)

As such, it is therefore mandatory to segue into an explication of our terms of reference as it pertains to quantifying connectivity over time as a measure. For that purpose, we have coined yet another new term to encapsulate the idea represented — connectivity-time ratio (CtR). What is meant by its usage is simply the proportion of time the graph exhibited the property of *strong-connectedness* during the simulation's duration as a ratio percentage.

Similarly, as in the case for ToC, it was a somewhat trivial matter to incorporate its additional computation. Since we already have in place a mechanism that computes the connected status of the network from moment to moment (see §6.4), we simply sample that as a Boolean value at the granularity level of the fixed-step discrete solver utilised within the encompassing Simulink model. Subsequent to the conclusion of the simulation, we then perform a summation of the logically 'true' values over that of total number of samples as the denominator, thereby deriving a CtR value for that simulation run's particular instance.

7.3 Data Collation

A natural corollary of the design statement's (see §7.1) presupposition — that each topology control optimisation algorithm under observation effects a change in a network's connectivity over time — is for the GEA's operation upon specific objective functions selected in accordance with their direct congruency as measures of connectivity, namely, those optimising for CC and BC (see §5.4.1 & §5.4.2, respectively). Further too, in answer to the concerns raised in §5.3, the specific variety of the proposed GEA to be utilised for the purpose of algorithmic comparison will be as that in its SOO-capacity. That is to say, we will make exclusive use of its ability to optimise for a single objective per se as opposed to a potential multi-objective optimisation. In so doing, affording the basis in which to make the comparison of its performance in each case against that of an EMST optimisation — considered to be as that typifying a standard topology control technique in the literature. In addition to which, we will make the further concession in our comparative algorithmic suite to that of inclusion of the simulation results for no optimisation performed whatsoever during the course to be sure of the influence, if any, of the random occurrence in connectedness. To summarise, therefore, the comparative suite of algorithms, or their derivative objective function thereof, is inclusive of:

1. GA optimising for BC;
2. GA optimising for CC;
3. EMST; and
4. None.

Prior to execution of the simulation suite, we performed a trial run in order to establish the effective value of c_t for the mobility pattern under consideration, as per the conditions and parameters of the steady-state mobility file detailed in §6.2, with which the DOE conforms.

As such, the value for c_t at the end of the simulation was found to be 165.1915 and hence taken to be the minimum transmitting range requirement for a sustained connectivity status in this particular instance (see §7.2.1). Using this as a baseline from which to determine a set of r_{max} assignments effective within a category of simulations, allows the means of ascertaining the sensitivity of the CtR to its manipulation thereof. Our first scenario, that is to say, category of simulation, will therefore be that of *minimal* range and its associated r_{max} value is that by rounding up the c_t value to the nearest multiple of five, 170, thereby allowing for some leeway in the respective algorithm's operation. In order to substantiate whether scaling factors are in evidence, we will take into account auxiliary scenarios at *medial* and *maximal* range to be situated at equally spaced intervals of an additional 20% from that of the baseline. Again, rounding up to the nearest multiple of five, we obtain auxiliary r_{max} values of 205 and 240 for medial and maximal range, respectively. Consequently, we arrive at three scenarios, of which each of the four algorithms is to be assessed. As a result, there are twelve combinations of scenario-algorithm pairings that are simulated 50 times apiece for a total suite of 600 simulation runs.

We do not project the effective values of r_{max} in a simulation any further than that already given in the interests of keeping within the dictates of the rationale for the utilisation of topology control techniques as that presented in §3.2, to wit, the benefit of multi-hop communications in the form of a savings in energy consumption and an increased network capacity. It is possible to demonstrate that the r_{max} values for each scenario, as given, are sufficient to furnish the conditions necessary for multi-hop communications. If we take the simulation area dimensions as being constant (see Table 6.1), the maximal inter-nodal distance between nodes is in the range of the longest side (i.e. sides are equal in this instance by virtue of being a square area) and that of the diagonal formed by conjoining the opposite corners of the same. The maximal number-of-hops, as an interval, will therefore be a function of the maximal inter-nodal distance's range so derived and the effective value of r_{max} within a particular scenario. The postulated values of which we obtain by calculation: $(2.94, 4.16)$, $(2.44, 3.45)$, and $(2.08, 2.95)$; for the scenarios of minimal, medial, and maximal range respectively. In all cases, endpoints in the interval are multiple (i.e. greater than one) and thereby conducive of the desired result.

Moreover, we will provide a brief account of the initial conditions determining the suite of simulations to be performed. In order that they all begin on a par with each other, an initial range assignment for the nodes constituting the network will be calculated, subsequently utilised, as the solitary starting point per run, subject to each and every entry therein not

exceed the effective value of r_{max} for the respective simulation. The values of which will be sampled from the uniform distribution via calls to the Matlab in-built function `rand` — a pseudo-random number generator. Given that topology maintenance updates are scheduled to occur every 10 s of simulation time duration, we will exclude the time series of samples for connectivity up to that of the first invocation of the optimisation algorithm, for that period's values will be equivalent in all instances. Besides which, being necessarily outside of the scope of operations of the respective algorithm. In the case of an optimisation algorithm type classification of None (i.e. no optimisation is performed whatsoever), the initial range assignments will remain in force throughout the duration of the simulation, but in every other case, subsequential range assignments will indubitably be of a different nature.

7.4 Data Visualisation

In probability and statistics, *density estimation* is the construction of an estimate of the density function, based on observed data assumed to be a sampling from an unknown underlying probability density function (Silverman 1998: p. 1). Kernel density estimation (KDE) is, therefore, a specific method of density estimation that, as Silverman (1998: p. 17) observes, is perhaps, apart from the histogram, the most commonly used estimator and certainly the most studied mathematically.

That being the case, in order to facilitate interpretation of the simulation results in the final analysis, we will derive the kernel density estimators for the connectivity-time ratio exhibited by each respective algorithm as an aggregate of the 50 runs performed at the designated value of r_{max}, namely, at either minimal, medial, or maximal range. For this purpose, we will make use of an implementation for univariate data as developed by Zdravko Botev (Researcher and Statistician at The University of Queensland in Brisbane, Australia) made freely available on the MathWorks Matlab Central File Exchange.

8 Discussion and Results

Further to the discussion in §6.2 and §7.2.1, we posit that c_t is dependent upon node density, as a function of the number of nodes constituting the network in addition to the physical dimensions of the simulation area, and the mobility pattern in question. In turn, $r_c \to 1$ for some value of r_{max}, equal to the product of a threshold-scaling coefficient (i.e. the scale factor for the threshold of connectivity), ω^*, and c_t, such that $\omega^* \geq 1$.

Consequently, by establishing an estimate of ω^*, we not only quantify the conditions necessary for sustainable connectivity, but also constitute the basis in which to make an informed decision as to a sufficient node density to bring this about. In other words, decision criteria for a connectivity-based node deployment. As such, the implication being that there is an optimal node density according to our terms of reference. Thus, when it comes to an analysis of the simulation results, our stated objective will be in the provision of just such an estimation as to ω^*. Further to which, should it prove to be the case that r_{max} is of such a degree to be expectant of a connectivity-time ratio approaching that of one, and the optimisation algorithm under observation is not productive of such a result, it calls into question its adequacy for the task at hand. Such would be the method of reasoning when it comes to a comparison of the relative merit of a particular algorithm over that of another.

To illustrate the utility of such decision criteria by way of example: should the effective value of r_{max} prove to be insufficient for the task of maintaining, in the ideal case, a connectivity-time ratio of one according to our condition that it be equal to $\omega^* c_t$, the only recourse is to manipulate the value of c_t to a sufficient degree, such that r_{max} not be exceeded. For, given the domain of application in that of a WSN, the region of interest (i.e. simulation area) will tend to be a fixed quantity, as that of r_{max}. Unless, of course, the wireless transceivers equipping each of the nodes were to be exchanged for another differently specced unit, which would be impractical to say the least. Assuming, also, that the mobility pattern has already been established, excepting that of any further nodes joining the network, it is left to node density as the variable and means by which we can bring about the desired effect of a change in the threshold of connectivity value. In particular, this would require an additional number of nodes to be deployed within the region of interest, thereby increasing the node density with a commensurate reduction in the threshold of connectivity. Note that the aforementioned necessitates that c_t must be recalculated anew in each instance to gauge the relative effect, if any, on its value by an

augmentation of the network. The same procedure applies, except in reverse, in situations where r_{max} is in excess of $\omega^* c_t$, which would be reflective of a node density over and above that required, in principle, for maintenance of a connected status of the network.

8.1 Relative Algorithmic Performance

As described in the previous chapter's treatment of the DOE (see §7.3), there are three categorical groupings or scenarios, on the basis of range, with which we have organised the simulation results to be presented in this section and further analysed numerically in §8.2. As we are endeavouring to establish the sensitivity of the connectivity-time ratio, r_c, to the effective maximum transmitting range, r_{max}, within the network (see §7.1), it follows that the relative performance of each algorithm be likewise considered according to the particular scenario at each respective minimal, medial, or maximal transmitting range. For this purpose, Figures 8.1–8.3 will herein be made reference to in the given order of presentation. All else being equal, excepting communication range, similarity in composition of all three graphs will facilitate making a comparison as we progress along in their respective interpretations. In each case, we will also make reference to the relevant component in Figure 8.4, as appropriate, for it affords an encapsulation or alternate view of the same results within Figures 8.1–8.3 (see §7.4).

To this end, it is worth bearing in mind the guiding principle for our investigation as put forward in the DOE's design statement and that epitomised in the question: 'is an r_{max} value equal to or in excess of the threshold of connectivity, in addition to being necessary, a sufficient condition to establish connectivity throughout the duration?' For instance, given $c_t = 165.1915$ in Figure 8.1, as that in Figures 8.2 & 8.3, an answer in the affirmative, by examination of the minimal range scenario contained therein, would obviate the need to consider any scenarios over and above that of its context.

N.B. As an unavoidable precursor to §8.2, for the scenario of minimal range in that of the next section, $r_{max} = \omega c_t = 170$. By substitution, therefore, the threshold-scaling coefficient, ω, is equal to 1.03. In the discussion that follows, we will make reference to the respective value for ω without qualification.

8.1.1 Minimal Range

In reference to Figure 8.1, where $\omega = 1.03$, we notice, in the case of an optimisation algorithm type classification of None (i.e. no optimisation was performed whatsoever throughout the duration of the simulation run in question — see §7.3) with a mean and

standard deviation of zero, no significant change or result in observation values is to be evidenced. Furthermore, the same applies in all subsequent scenarios. As such, it can be concluded that, the strongly connected status of the network is not determined by its random occurrence in any instance whatsoever. Consequently, negating its import as a factor in our discussion hereupon.

Similar to which, we can reason with respect to EMST optimisation, albeit exhibiting a non-zero mean, yet still productive, again, of no significant variation by virtue of its standard deviation value of zero in not only the scenario of minimal range, but also that of the remainder. Although this is suggestive of a stability in r_c values in all cases, it is of such small magnitude to render it as an ineffectual means of achieving a connected status of the network in the presence of mobility. In this context, it is made plain that alternative techniques are unequivocally required in order to achieve the stated objective of sustained as opposed to a momentary connected status of the network, which is as the case proves to be when observant of the associated communication graph (see Figure 6.3) in any particular instance of an in-progress simulation run as it proceeds towards conclusion. The observable behaviour of which is that subsequent to the algorithm's invocation (at the TM update interval setting in force for that simulation, which specifies the time period between consecutive optimisations), even should the result happen to be a connected status of the network, immediately thereafter, it just as soon disconnects should any singular node's position deviate from its initial position as at the point in time when the algorithm was first invoked in a particular instance. That being the case, the connectivity-time ratio for EMST is an aggregate sum of discontinuous time moments of connected status over the total duration of the simulation. In other words, although, in itself, an efficient technique for producing a connected network, by doing so at minimal cost, it does not constitute the necessary conditions for a sustained status thereof beyond the conclusion of its operation. Thence, a result categorically contrary to that which is desired. It too, will therefore be precluded from further consideration as it pertains to our discussion of the relative performance characteristics of the algorithms under observation.

As a result, what remains is for the operative result in each instance of the proposed GEA performing in its capacity to optimise for either BC or CC (i.e. in isolation and therefore signifying a SOOP formulation). Although improving upon the result for None and EMST, both variations in GA are as yet far removed from a connectivity-time ratio approaching one (viz. connected status of the network throughout the duration of the simulation). In particular, BC optimisation achieved a greater mean, but with greater deviation in values,

relative to CC optimisation, as borne out by the associated kernel density estimators for the distribution in Figure 8.4: indicating the range of output values extend as far as 0.5–0.6. It is suggestive of the presence of a certain link(s) problematic for the algorithm whose transmitting range requirement, in order to render the network connected, is at or close to the boundary delimited by the threshold of connectivity and resultingly unable due to the same being at or near the effective value of r_{max}, such that both measures are virtually equivalent. Hence, the marked pendulum effect between consecutive simulation runs in Figure 8.1: in one instance the problematic link is established on the whole, whereas, in another, it is not, producing great variation in connectivity-time ratio values. The same can be said for CC optimisation, but, in this case, the pendulum effect is much reduced over that of BC.

As a consequence, we can state, without reservation, that an r_{max} value equal to c_t, or its approximation, is not sufficient to produce the conditions necessary for a sustained connected status of the network. That is to say, nonoptimal due to the outcomes being far removed from a value for $r_c \to 1$. It remains to be seen as to what degree the threshold-scaling coefficient needs to be increased before such becomes observable in the outcomes.

8.1.2 Medial Range

In reference to Figure 8.2, where $\omega = 1.24$, we observe a marked improvement in output values for both varieties of GA. Where they diverge, apart from sheer magnitude in the mean, is seen in terms of standard deviation. Inasmuch as, while the shift from minimal-to-medial range resulted in significant reduction for BC, it actually increased in the case of CC optimisation over that of the corresponding values in the minimal range scenario, albeit not of a substantial differential. We can deduce from this observation that, while BC optimisation tends towards stabilisation in instantiated output values as we increase the effective r_{max} value past that of the threshold of connectivity, c_t, the contrasting trend in CC optimisation is for the overall spread of values to be relatively constant, irrespective of its actual shift about a new higher mean level, which the increase in maximum transmitting range necessarily entails. By inspection, the same can be verified in the corresponding kernel density estimators for each algorithm at medial transmitting range in Figure 8.4.

Returning now to our guiding principle in an interpretation of the experimental results, we can see that, in the case of BC optimisation, we are approaching a result conforming to our expectation that we achieve sustainability in the connected status of the network. We do well, though, to ask whether it is possible to do even better.

8.1.3 Maximal Range

In reference to Figure 8.3, where $\omega = 1.45$, we again notice an improvement in output values in the mean for both varieties of GA, although not quite as significant as the shift from minimal-to-medial range. Similar, though, in respect to a reduction in standard deviation, which the shift from medial-to-maximal range entailed for BC optimisation, albeit of reduced magnitude in this instance. The relative difference in mean values between the two algorithms for this scenario remains about the same as before, as does that of their standard deviation. We can see that BC optimisation now approaches fairly stable output values in addition to being close to the limit achievable in magnitude for the mean value. In the case of CC, the previous trend of an increased standard deviation has now reversed, but still within a few points short of the value exhibited in the minimal range scenario. Such would indicate that no significant improvement in the spread of values would be achieved by a further incremental change in the effective maximum transmitting range. For, the trend has been fairly consistent thus far throughout the whole gamut of values for its instantiation, which is made all the more clear by an examination of its associated series of kernel density estimators from minimal through maximal ranges in Figure 8.4. Likewise, the trend continues, as before, for BC optimisation in that its overall spread of values diminishes with each incremental change in r_{max}.

Thus, in the case of BC optimisation, we can be fairly confident in stating we are approaching optimality purely in terms of raw values in the observed variable, r_c, for a value in ω of 1.45. It is left for the next section to establish an estimation as to the optimal threshold-scaling coefficient, ω^*, and by implication the best to be expected of each algorithm in regards to their sustainability in the connected status of the network.

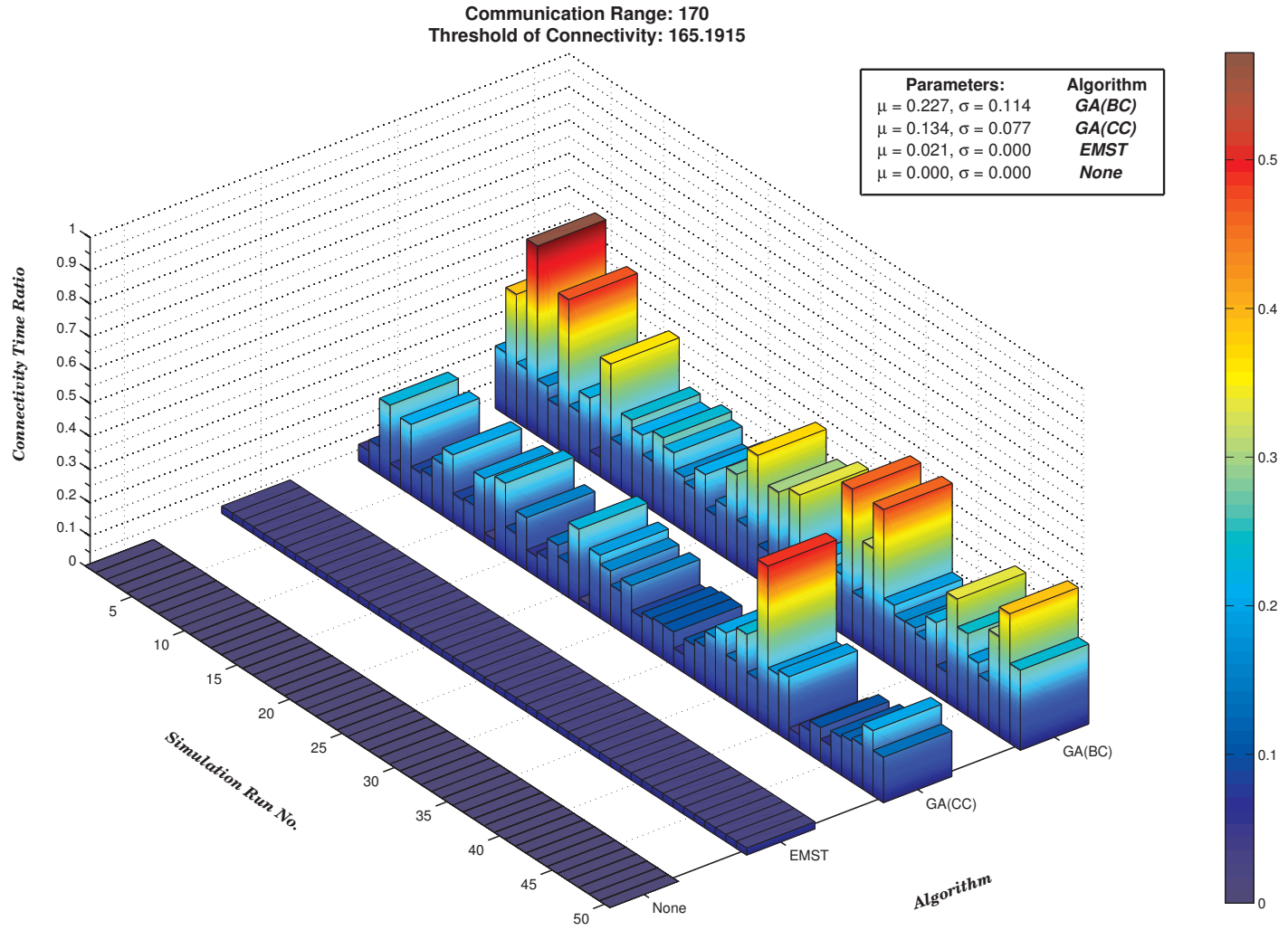

Figure 8.1: 3-D bar graph for the connectivity-time ratio per simulation run for each algorithm at minimal transmitting range.

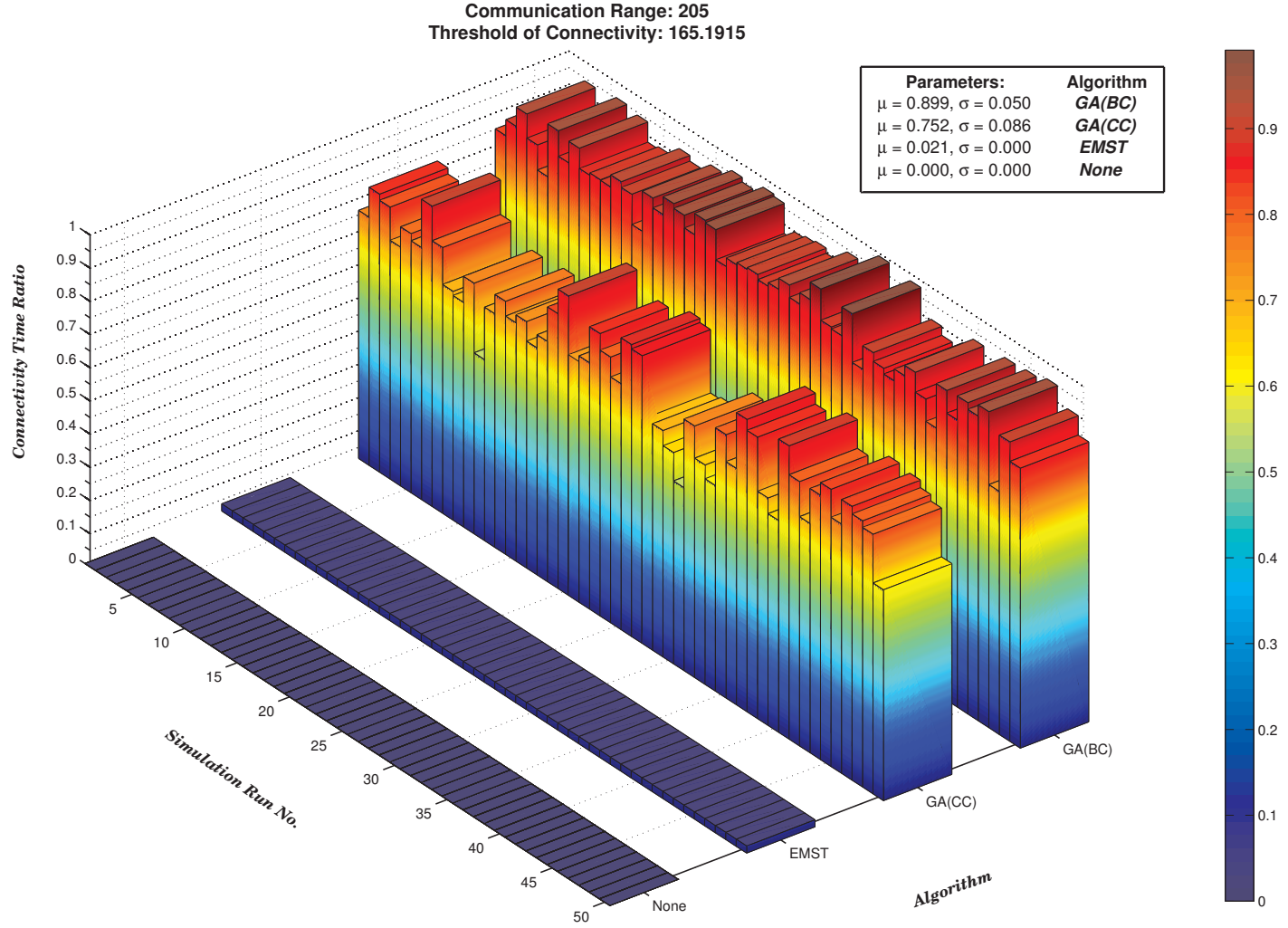

Figure 8.2: 3-D bar graph for the connectivity-time ratio per simulation run for each algorithm at medial transmitting range.

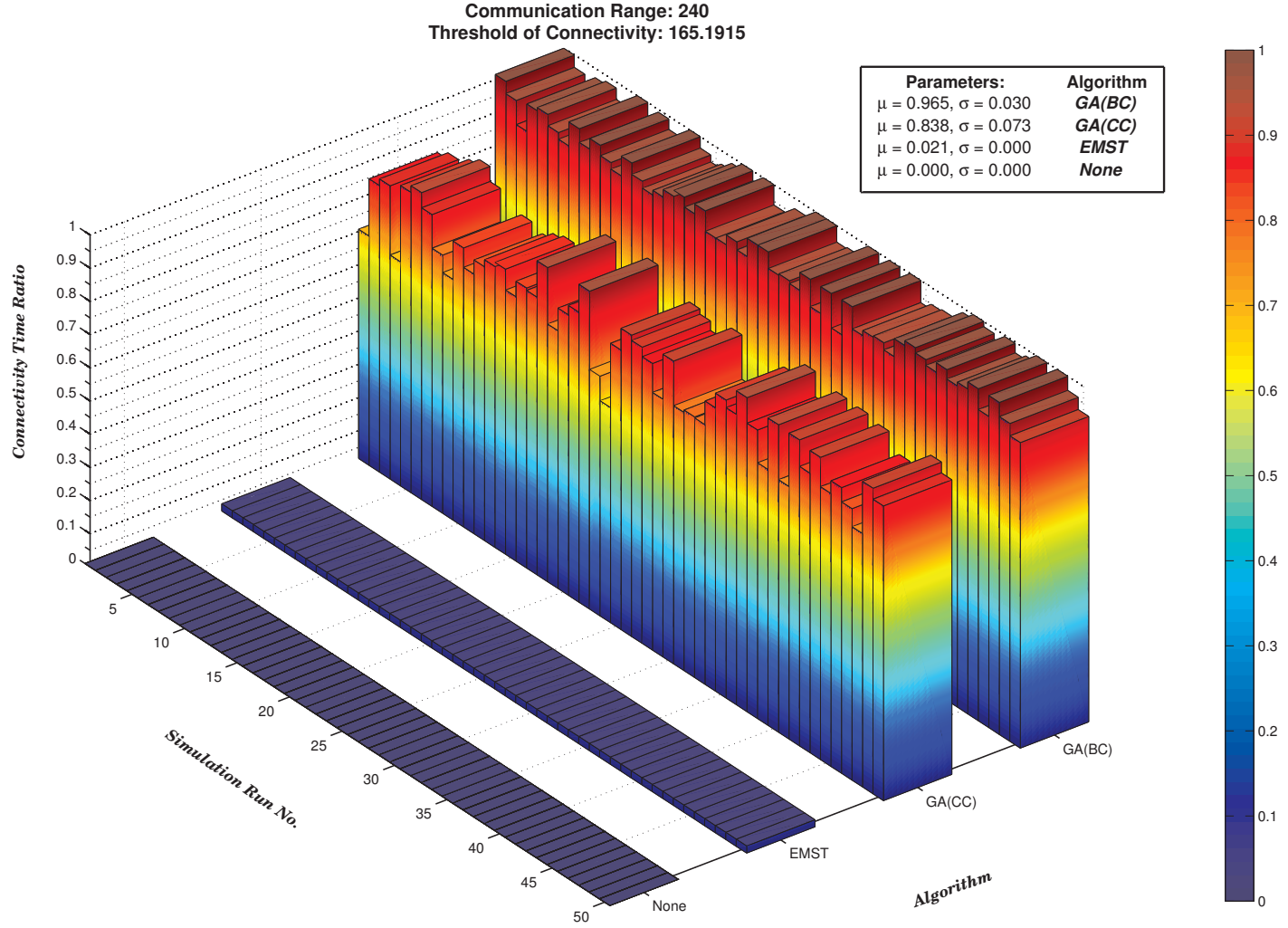

Figure 8.3: 3-D bar graph for the connectivity-time ratio per simulation run for each algorithm at maximal transmitting range.

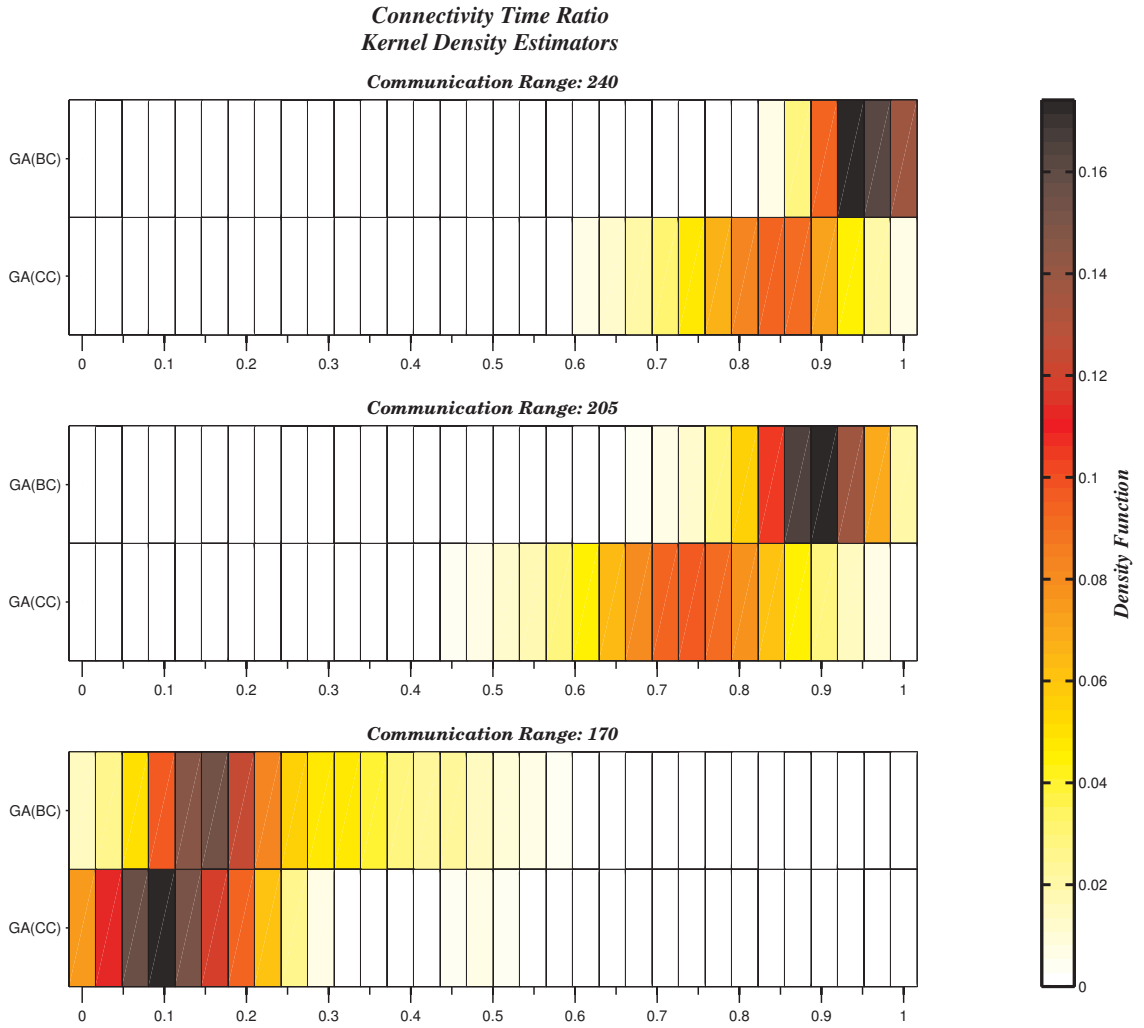

Figure 8.4: Connectivity-time ratio kernel density estimators for GA optimisation at minimal, medial, and maximal transmitting ranges.

8.2 Threshold-Scaling Coefficient

To summarise the results and to place them within the context of a formulation of an estimate as to the optimal value for the threshold-scaling coefficient, ω^*, as that made reference to previously and to be developed further as we proceed along, we present the following equations (i.e. we will exclude from consideration those results pertaining to EMST and None for the same reasons as stated in §8.1.1):

$$r_c\left(\left(\mu-\sigma\right),\mu,\left(\mu+\sigma\right)\right)=F_{BC}\left(\omega\right)$$
$$=\begin{cases}(0.935,0.965,0.995) & \text{if } \omega=1.45 \text{ and } r_{max}=\omega\, c_t=240,\\ (0.849,0.899,0.949) & \text{if } \omega=1.24 \text{ and } r_{max}=\omega\, c_t=205,\\ (0.113,0.227,0.341) & \text{if } \omega=1.03 \text{ and } r_{max}=\omega\, c_t=170.\end{cases} \tag{8.1}$$

and

$$r_c\left(\left(\mu-\sigma\right),\mu,\left(\mu+\sigma\right)\right)=F_{CC}\left(\omega\right)$$
$$=\begin{cases}(0.765,0.838,0.911) & \text{if } \omega=1.45 \text{ and } r_{max}=\omega\, c_t=240,\\ (0.666,0.752,0.838) & \text{if } \omega=1.24 \text{ and } r_{max}=\omega\, c_t=205,\\ (0.057,0.134,0.211) & \text{if } \omega=1.03 \text{ and } r_{max}=\omega\, c_t=170.\end{cases} \tag{8.2}$$

where $c_t=165.1915$ in each case.

The associated probabilities for the observed connectivity-time ratio values on the interval, $\left(\left(\mu-\sigma\right),\left(\mu+\sigma\right)\right)$, for each algorithm, obtained by integration of the corresponding kernel density estimators, are as that below:

$$P_{BC}\left(\left(\mu-\sigma\right)\leq r_c\leq\left(\mu+\sigma\right)\right)$$
$$=\begin{cases}0.5588 & \text{if } \omega=1.45 \text{ and } r_{max}=\omega\, c_t=240,\\ 0.6104 & \text{if } \omega=1.24 \text{ and } r_{max}=\omega\, c_t=205,\\ 0.6554 & \text{if } \omega=1.03 \text{ and } r_{max}=\omega\, c_t=170.\end{cases} \tag{8.3}$$

and

$$P_{CC}\left(\left(\mu-\sigma\right)\leq r_c\leq\left(\mu+\sigma\right)\right)$$
$$=\begin{cases}0.6782 & \text{if } \omega=1.45 \text{ and } r_{max}=\omega\, c_t=240,\\ 0.5681 & \text{if } \omega=1.24 \text{ and } r_{max}=\omega\, c_t=205,\\ 0.6976 & \text{if } \omega=1.03 \text{ and } r_{max}=\omega\, c_t=170.\end{cases} \tag{8.4}$$

where, again, $c_t = 165.1915$ in each case. Thus constituting an estimation of the likelihood that the connectivity-time ratio falls within one standard deviation either side of the mean in any particular simulation run. Therefore, by our reckoning, it does so more often than not in all cases.

By a process of an exploration of the data, namely, Equation (8.1) & Equation (8.2), in an interactive session within Matlab's Curve Fitting Tool environment for fitting curves to univariate data, we found that tool's parametric model library's standard type of `rational` to be productive of the best goodness-of-fit statistics. That is to say, unless otherwise indicated, resulting in values for SSE $\cong 0$ and $R^2 = 1$. To clarify, rational models are defined as ratios of polynomials with the leading coefficient of the denominator set to 1, thereby ensuring uniqueness of the numerator and denominator in cases where the polynomial degrees of each are equal. Model names are in the form '`ratij`', where i is the degree of the numerator and j that of the denominator. In our case, we specified the `rat11` option, indicating a linear polynomial in both numerator and denominator, of the form:

$$f(x) = \frac{p_1 x + p_2}{x + q_1}. \tag{8.5}$$

As such, we began by using the model so formed to establish a fit for the observed values in r_c, the connectivity-time ratio, at its mean, μ, as a function of ω, the threshold-scaling coefficient, for each algorithm and scenario under consideration. The results of which produce the following parametric model representations:

$$\bar{r}_c = F_{BC}(\omega) = \frac{1.045\omega - 1.065}{\omega - 0.983} \tag{8.6}$$

and

$$\bar{r}_c = F_{CC}(\omega) = \frac{0.952\omega - 0.970}{\omega - 0.961}. \tag{8.7}$$

The graphs of which resemble that of a logarithmic function, as do all subsequent parametric fitting procedures, albeit with an x-intercept within the neighbourhood of the minimum value for ω in addition to, necessarily, converging to a limit approximating a value about one in its maximal value in y, the dependent variable, r_c. Thus providing a visual cue as to the non-linear relationship between ω and r_c. Likewise, we additionally performed fits to data one standard deviation either side of the mean for each algorithm, productive in total, therefore, of six parametric curves in order to establish a range of values in any subsequent predictors.

Now that the proper tools are in place to substantiate our claims, let us return to the primary concern of the present section, that is, the establishment of an estimate of the optimal value for ω^*. The starting point for such a determination is a continuation and replication of the reasoning and procedure as set forth in §7.3 regarding the imposition of a limit on the effective value of r_{max} in the interest of keeping within the bounds of multi-hop communications due to its inherent benefit to the domain of application in that of a WSN.

By extension, we derive the corresponding limit for ω^*. Hence, it is assumed as given that the lower bound on the maximal number-of-hops is a constant ratio of the simulation area's shortest physical dimension to that of r_{max}, namely, $2:1$. So too, will the limit on ω^* be similarly proportioned, except, in this case, the ratio is $2c_t:1$. In our instance, for a shortest side length of 500 m and $c_t = 165.1915$, we obtain by calculation: $\omega^* = 1.51$. The resulting maximal number-of-hops interval will therefore be in the range $(2.00, 2.83)$ for such a value so derived. Exceeding the limit so imposed, the situation will inevitably start to verge toward direct communications between nodes compromising the desirability of it being otherwise.

Taking this then to be our estimation of the optimal threshold-scaling coefficient, ω^*, it is worthwhile to examine the behaviour of the predicted data points in r_c as ω approaches and/or exceeds the value for ω^*. We do so in the following for each algorithm in equally spaced intervals of the input parameter, ω, where the value to the far right within parentheses indicates the incremental change in estimate over that of directly previous in order to highlight the trend for those parametric curves as that about the mean:

$$\begin{aligned}
\hat{F}_{BC}(1.6) &= 0.984167 \quad (+0.011839) \\
\hat{F}_{BC}(1.5) &= 0.972328 \quad (+0.017515) \\
\hat{F}_{BC}(1.4) &= 0.954813 \quad (+0.028565) \\
\hat{F}_{BC}(1.3) &= 0.926248 \quad (+0.054888) \\
\hat{F}_{BC}(1.2) &= 0.871360 \quad (+0.000000)
\end{aligned} \tag{8.8}$$

and

$$\begin{aligned}
\hat{F}_{CC}(1.6) &= 0.864189 \quad (+0.016243) \\
\hat{F}_{CC}(1.5) &= 0.847946 \quad (+0.023637) \\
\hat{F}_{CC}(1.4) &= 0.824309 \quad (+0.037566) \\
\hat{F}_{CC}(1.3) &= 0.786743 \quad (+0.068949) \\
\hat{F}_{CC}(1.2) &= 0.717794 \quad (+0.000000).
\end{aligned} \tag{8.9}$$

It is clear that the trend in both is similar, where we obtain diminishing returns at each stepped increment of the input parameter. Although, in the case of optimisation for BC, it is of smaller proportion in each instance, which can be understood in terms of that we are approaching the bound on values in r_c of one. As a consequence, for all intents and purposes, essentially obviating the requirement for a value in excess of ω^* purely on the basis of its effect upon the dependent variable. On the other hand, optimising for CC, there may be some benefit in doing so, but, as previously stated, we have already reached the point of diminishing returns and if we were to factor in the additional cost of communications such an action would entail, likewise too, in this case, the value we posit for ω^* should be sufficient under the circumstances.

If utilised as the effective value for ω in a particular instance of simulation, we can provide an estimate as to a range of observed values to be instantiated in the connectivity-time ratio for each algorithm. By incorporating the accompanying parametric curves one standard deviation either side of the mean into our analysis, we obtain:

$$\hat{r}_c\left((\mu-\sigma),\mu,(\mu+\sigma)\right)=\hat{F}_{BC}\left(\omega^*\right)=(0.95,0.97,1.00) \tag{8.10}$$

and

$$\hat{r}_c\left((\mu-\sigma),\mu,(\mu+\sigma)\right)=\hat{F}_{CC}\left(\omega^*\right)=(0.78,0.85,0.92), \tag{8.11}$$

where $\omega^*=1.51$ in each case.

A result affording further means of comparison in the relative performance of the two algorithms: not only does the optimisation for BC approach the upper bound in r_c, it does so with a fair degree of consistency compared to CC optimisation, which exhibits a standard deviation more than double that for BC; besides, being of lower overall magnitude in output values.

In closing, we will now furnish a concrete example demonstrative of the derived threshold-scaling coefficient's use as a network diagnostic tool. For instance, let us suppose the nodes constituting a WSN are specified as having an r_{max} value of 150 m. Assuming the same conditions herein, it is evident from the outset, before any other considerations, that such a value is inadequate even in terms of the minimum requirement for sustained network connectivity status. A maximum transmitting range of such low level would require that the threshold of connectivity be reduced from its present level of 165.1915 to 99.3378, or

as close as is practicable. Thus, satisfying the equation:

$$c_t = \frac{r_{max}}{\omega^*} = \frac{150}{1.51} = 99.3378 \tag{8.12}$$

All else being equal, the only means for achieving such a result would be to increase the number of nodes deployed within the region of interest, thereby increasing the effective node density and from which we should expect to find a concomitant reduction in c_t from its previous value. Needless to say, it remains to be seen whether and to what extent such can be verified by further studies.

9 CONCLUSIONS

It has been shown that the specific variety of algorithm that achieves the stated objective of an uninterrupted connected status of the network in the presence of mobility is that of GA optimising for BC. That is to say, a minimisation in the number of strong articulation points in the associated graph for the network. In which, a result of zero represents a maximally biconnected subgraph comprising the whole graph, or simply a biconnected graph. Such incorporates a measure of planned redundancy in the formation of communication links, such that there exists at least two node-disjoint paths between any pair of distinct nodes in the network, thereby attaining a degree of robustness against potential link(s) and/or node(s) failures, whatever its cause.

Besides which, it is a categorical imperative that the effective maximum transmitting range, as that characteristic property of the radio transceiver equipping nodes constituting the network assuming homogeneous specification thereof, is of sufficient magnitude to satisfy being the product of a certain optimality in a threshold-scaling coefficient and that of the threshold of connectivity, as that taken to be the mobility scenario under observation's minimum transmitting range requirement for sustainability of a network's strongly connected status throughout the duration. Otherwise, accomplishment of the stated objective is precluded. By deriving such determining characteristics of a network's sustainable connectivity, we established the quantifiable bases in which to predict the outcome in the observed value of the connectivity-time ratio, being simply the proportion of time the network exhibited the property of strong connectedness as a ratio percentage.

Recognising the inherent limitations within the domain of application in that of a WSN, the effective maximum transmitting range may not always be of the proper constitution to effect the desired result. Unless, of course, the wireless transceivers equipping each of the nodes were to be exchanged for another differently specced unit, which would be impractical to say the least, or even impossible for all intents and purposes depending on the environment in which the nodes are deployed. As such, all else being equal, we identified the only available means to bring about the necessary conditions. That being, an informed decision as to a sufficient node density in the deployed region. In other words, decision criteria for a connectivity-based node deployment. At present, since the aforesaid is but only in the manner of a heuristic principle, it remains for further potential studies in the topic to establish it as a principle in closed form and to substantiate our claims.

BIBLIOGRAPHY

Canadian Conservation Institute (2005). Analytical Hierarchy Process (AHP) Program [online]. [cited May 7, 2011]. Available: https://www.cci-icc.gc.ca/tools/ahp/index_e.asp.

Chao, X. (2008). *Wireless Network Study and Analysis Using NS-2 Simulator*. Master's thesis, University of Vaasa, Vaasa, Finland.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*. Chichester, England: John Wiley & Sons, Ltd.

Forman, E. H. & M. A. Selly (2001). *Decision by Objectives: How to Convince Others that You Are Right*. Singapore, Republic of Singapore: World Scientific Publishing Co.

Gleich, D. (2007). Matlab BGL v2.1 (Documentation) [PDF]. [cited May 3, 2011]. Available: http://www.stanford.edu/~dgleich/programs/matlab_bgl/matlab_bgl_v2.1.pdf.

Gross, J. L. & J. Yellen, eds. (2003). *Handbook of Graph Theory*, vol. 25 of *Discrete Mathematics and Its Applications*. Boca Raton, FL: CRC Press LLC.

Italiano, G., L. Laura & F. Santaroni (2010). Finding strong bridges and strong articulation points in linear time. In W. Wu & O. Daescu, eds., *Combinatorial Optimization and Applications*, vol. 6508 of *Lecture Notes in Computer Science*, Berlin / Heidelberg, Germany: Springer-Verlag GmbH, pp. 157–169.

Jungnickel, D. (2008). *Graphs, Networks and Algorithms*, vol. 5 of *Algorithms and Computation in Mathematics*. 3rd Ed. Berlin / Heidelberg, Germany: Springer-Verlag GmbH.

Karl, H. & A. Willig (2005). *Protocols and Architectures for Wireless Sensor Networks*. Chichester, England: John Wiley & Sons, Ltd.

Labrador, M. A. & P. M. Wightman (2009). *Topology Control in Wireless Sensor Networks*. Dordrecht, Netherlands: Springer Science+Business Media B.V.

Navidi, W. & T. Camp (2004). Stationary distributions for the random waypoint mobility model. *IEEE Transactions on Mobile Computing* 3(1), 99–108.

Navidi, W., T. Camp & N. Bauer (2003). Improving the accuracy of random waypoint simulations through steady-state initialization. Tech. Rep. MCS-03-08, The Colorado School of Mines, Golden, CO.

Nethi, S., M. Pohjola, L. Eriksson & R. Jäntti (2007a). Platform for emulating networked control systems in laboratory environments. In *Proceedings of the 2007 IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM 2007), Helsinki, Finland, June 18–21, 2007*. Los Alamitos, CA: IEEE Computer Society, pp. 1–8.

Nethi, S., M. Pohjola, L. Eriksson & R. Jäntti (2007b). Simulation case studies of wireless networked control systems. In *Proceedings of the 2nd ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks (PM2HW2N 2007), Chania, Crete Island, Greece, October 22–26, 2007*. New York, NY: ACM, pp. 100–104.

Santi, P. (2005). *Topology Control in Wireless Ad Hoc and Sensor Networks*. Chichester, England: John Wiley & Sons, Ltd.

Sedgewick, R. (2002). *Algorithms in C, Part 5: Graph Algorithms*. 3rd Ed. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.

Sensinode Ltd. (2007). *NanoSensor N711 (Datasheet)*. Sensinode Ltd., Oulu, Finland.

Siek, J. G., L.-Q. Lee & A. Lumsdaine (2002). *The Boost Graph Library: User Guide and Reference Manual*. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.

Silverman, B. W. (1998). *Density Estimation for Statistics and Data Analysis*. Boca Raton, FL: CRC Press LLC.

Sohraby, K., D. Minoli & T. Znati (2007). *Wireless Sensor Networks: Technology, Protocols, and Applications*. Hoboken, NJ: John Wiley & Sons, Inc.

The MathWorks, Inc. (2009a). *Genetic Algorithm and Direct Search Toolbox (User's Guide)*. The MathWorks, Inc., Natick, MA.

The MathWorks, Inc. (2009b). *Optimization Toolbox (User's Guide)*. The MathWorks, Inc., Natick, MA.

Toivonen, E. & L. M. Eriksson (2008). Wireless sensor networks in real-time control systems: A case study. In *Proceedings of the 2008 Workshop on Wireless Communication and Applications (WoWCA 2008), Vaasa, Finland, April 2, 2008*. Vaasa, Finland: University of Vaasa, Department of Computer Science.

Winston, W. L. (1991). *Operations Research: Applications and Algorithms*. 2nd Ed. Boston, MA: PWS-KENT Publishing Company.

Zhonglei, L. (2008). *Mobility Control in Wireless Sensor Networks: A Simulation Case Study*. Master's thesis, University of Vaasa, Vaasa, Finland.

Appendices

A ELEMENTS OF GRAPH THEORY

The following introduces a selection of pertinent background material related to the vast field of graph theory which underpins the very theoretical foundation of the work carried out herein and thus its incessant use as terminology. It adopts a similar format and content (with modifications) as that contained in the appendix to Santi (2005: pp. 225*ff.*) unless otherwise indicated.

A.1 Basic Definitions

A.1.1 Graph

A graph G is an ordered pair of disjoint sets (N,E), where $E \subseteq N \times N$. Set N is called the vertex, or node, set, while set E is the edge set of graph G. Typically, it is assumed that self-loops (i.e. edges of the form (u,u), for some $u \in N$) are not contained in a graph.

A.1.2 Subgraph

Given a graph $G = (N,E)$, a subgraph of G is any graph $G' = (N',E')$ such that $N' \subseteq N$ and $E' \subseteq E$. Given any subset N' of the nodes in G, the subgraph of G induced by N' is defined as $G_{N'} = (N',E(N'))$, where $E(N') = \{(u,v) \in E : u,v \in N'\}$, that is, $G_{N'}$ contains all the edges of G such that both endpoints of the edge are in N'.

A.1.3 Order of a graph

The order of graph $G = (N,E)$ is the number of nodes in G, that is, the cardinality of set N.

A.1.4 Complete graph

The complete graph $K_n = (N,E)$ of order n is such that $|N| = n$, and $(u,v) \in E$ for any two distinct nodes $u,v \in N$.

A.1.5 Planar graph

A graph $G = (N,E)$ is planar if it can be drawn in the plane in such a way that no two edges in E intersect.

Note that a graph G can be drawn in several different ways; a graph is planar if there exists at least one way of drawing it in the plane in such a way that no two edges cross each other.

A.1.6 Directed and undirected graph

A graph $G = (N,E)$ is directed if the edge set is composed of ordered node pairs. A graph is undirected if the edge set is composed of unordered node pairs.

A.1.7 Weighted graph

A weighted graph is a graph in which edges, or nodes, or both, are labelled with a weight.

A.1.8 Neighbour nodes

Given a graph $G = (N,E)$, two nodes $u,v \in N$ are said to be neighbours, or adjacent nodes, if $(u,v) \in E$. If G is directed, we distinguish between incoming neighbours of u (those nodes $v \in N$ such that $(v,u) \in E$) and outgoing neighbours of u (those nodes $v \in N$ such that $(u,v) \in E$).

A.1.9 Node degree

Given a graph $G = (N,E)$, the degree of a node $u \in N$ is the number of its neighbours in the graph. Formally,

$$deg(u) = |\{v \in N : (u,v) \in E\}|.$$

If G is directed, we distinguish between in-degree (number of incoming neighbours) and out-degree (number of outgoing neighbours) of a node.

A.1.10 Path

Given a graph $G = (N,E)$, and given any two nodes $u,v \in N$, a path connecting u and v in G is a sequence of nodes $\{u = u_0, u_1, \ldots, u_{k-1}, u_k = v\}$ such that for any $i = 0, \ldots, k-1, (u_i, u_{i+1}) \in E$. The length of the path is the number of edges in the path.

A.1.11 Cycle

A cycle is a path $C = \{u_0, \ldots, u_k\}$ such that $k \geq 3$, $u_0 = u_k$, and the other nodes in C are distinct from each other and from u_0.

A.1.12 Connected and strongly connected graph

A graph $G = (N,E)$ is connected if for any two nodes $u,v \in E$ there exists a path from u to v in G. If G is directed, we say that G is strongly connected if for any two nodes $u,v \in E$ there exists a path from u to v, and a path from v to u in G.

A.1.13 k-connected graph

A graph $G = (N,E)$ is k-(node-)connected, for some $k \geq 2$, if removing any $k-1$ nodes from the graph does not disconnect it. It can be easily proven that a graph is k-connected

if and only if there exist at least k node-disjoint paths between any pair of distinct nodes in G.

A.1.14 Graph connectivity

The (node) connectivity of a graph $G = (N,E)$, denoted as $\kappa(G)$, is the maximum value of k such that G is k-connected.

A.1.15 Tree

A tree $T = (N,E)$ is a connected graph with n nodes and $n-1$ edges, that is, a tree is a minimally connected graph.

A.1.16 Rooted tree

A rooted tree $T = (N,E)$ is a tree in which one of the nodes is selected as the tree root.

A.1.17 Spanning tree

Given a connected graph $G = (N,E)$, a spanning tree of G is a tree $T = (N,E_T)$ that contains all the nodes in G and is such that $E_T \subseteq E$.

A.1.18 Cost of a spanning tree

Given an edge-weighted graph $G = (N,E)$, the cost of a spanning tree T of G is the sum of the weights on its edges.

A.1.19 Minimum spanning tree

Given an edge-weighted graph $G = (N,E)$, a Minimum Spanning Tree (MST) for G is a spanning tree of G of minimum cost.

A.1.20 Euclidean MST

Given a set N of nodes placed in the d-dimensional space (with $d = 1,2,3$), and a set of edges E between these nodes, a Euclidean MST (EMST) is a MST of the edge-weighted graph $G = (N,E)$, where each edge has a weight equal to the Euclidean distance between its endpoints.

A.2 Advanced Concepts

The current subsection is a subset of the prefatory material presented in Italiano et al. (2010) and is essential for an understanding of the discussion as to core algorithm design and development that forms the main body of the present work.

A.2.1 Reversal graph

Given a directed graph $G = (V, E)$, define its reversal graph $G^R = (V, E^R)$ *by reversing all edges of G: namely, G^R has the same vertex set as G and for each edge (u, v) in G there is an edge (v, u) in G^R. We say that the edge (v, u) in G^R is the* reversal *of edge (u, v) in G.*

A.2.2 Flowgraph

A flowgraph $G(s) = (V, E, s)$ is a directed graph with a start vertex $s \in V$ *such that every vertex in V is reachable from s.*

A.2.3 Dominance relation

The dominance relation *in $G(s)$ is defined as follows: a vertex u is a* dominator *of vertex v if every path from vertex s to vertex v contains vertex u.*

A.2.4 Trivial dominator

Let $dom(v)$ be the set of dominators of v. Clearly, $dom(s) = \{s\}$ and for any $v \neq s$ we have that $\{s, v\} \subseteq dom(v)$: we say that s and v are the trivial dominators *of v in the flowgraph $G(s)$.*

A.2.5 Dominator tree

The dominance relation is transitive and its transitive reduction is referred to as the dominator tree *$DT(s)$. Note that the dominator tree $DT(s)$ is rooted at vertex s. Furthermore, vertex u dominates vertex v if and only if u is an ancestor of v in $DT(s)$. If u is a dominator of v, and every other dominator of u also dominates v, we say that u is an* immediate dominator *of v. It is known that if a vertex v has any dominators, then v has a unique immediate dominator: the immediate dominator of v is the parent of v in the dominator tree $DT(s)$.*

A.2.6 Strong articulation point

Let $G = (V, E)$ be a strongly connected graph, and let s be any vertex in G. Let $G(s) = (V, E, s)$ be the flowgraph with start vertex s. If u is a non-trivial dominator of a vertex v in $G(s)$, then u is a strong articulation point *in G.*

A.2.7 Strongly connected components

Let $G = (V, E)$ be a directed graph, with m edges and n vertices. A directed path *in G is a sequence of vertices $v_1, v_2, \ldots, v_k$, such that edge $(v_i, v_{i+1}) \in E$ for $i = 1, 2, \ldots, k-1$. A directed graph G is* strongly connected *if there is a directed path from each vertex in the graph to every other vertex. The* strongly connected components *of G are its maximal strongly connected subgraphs.*

A.3 Proximity Graphs

Proximity graphs are a class of graphs introduced in the theory of Computational Geometry that are based on proximity relationships between nodes.

A.3.1 Maximal planar subdivision

Given a set N of points in the plane, a maximal planar subdivision of N is a planar graph $G = (N, E)$ such that no edge connecting two nodes in N can be added to E without compromising graph planarity.

A.3.2 Triangulation

Given a set N of points in the plane, a triangulation of N is a maximal planar subdivision whose node set is N.

A.3.3 Delaunay triangulation

Given a set N of points in the plane, the Delaunay triangulation of N is the unique triangulation DT of N such that the circumcircle of every triangle contains no points of N in its interior.

A.3.4 Adjacency matrix

The adjacency matrix of a digraph $G = (V, E)$, denoted A_G, is given by:

$$A_G[u,v] = \begin{cases} \textit{the number of arcs from } u \textit{ to } v \textit{ if } u \neq v \\ \textit{the number of self-loops at } v \textit{ if } u = v \end{cases}$$

(Gross & Yellen 2003)

A.3.5 Communication graph

Given a set N of nodes (representing units of an ad hoc or sensor network), the communication graph is the directed graph $G = (N, E)$ such that edge $(u, v) \in E$ only if v is within u's transmitting range at the current transmit power level.

A.3.6 Maxpower graph

Given a set N of nodes (representing units of an ad hoc or sensor network), the maxpower graph is the communication graph $G = (N, E)$ such that $(u, v) \in E$ if and only if v is within u's transmitting range at maximum power, that is, the maxpower graph contains all possible wireless links between the nodes in the network.

B PiccSIM - A Wireless Networked Control System Simulation Platform

PiccSIM is an acronym that provides a complete description of its purpose and is defined by its co-authors as a 'Platform for Integrated Communications and Control Design, Simulation, Implementation and Modelling'. Its use allows various networks to be emulated using the real-time extension of the widely known NS-2 network simulator and integration of the control design tools available in Matlab/Simulink. In so doing, yielding the possibility for evaluating the performance of communication protocols in real-time control systems and similarly test their robustness and other characteristic properties in wireless networked environments (Nethi, Pohjola, Eriksson & Jäntti 2007b).

B.1 Motivation

The use of wireless communications in real-time control systems for the provision of measurement data or the transmission of control signals is generally considered an unreliable process. Traditional control theories assume a perfect, synchronous sampling paradigm, thereby do not take into account the crucial factors due to the inherent nature of its shared medium and the resultant random retransmission times after collisions, namely packet loss, asynchronous sampling or varying time-delay. Thus there is a definite need for the development of new theories incorporating an integrated approach to wireless communications and control. Likewise, we are simultaneously obliged to develop suitable simulation platforms for theory testing and verification prior to implementation in real industrial systems. It could well be considered that PiccSIM, as envisaged, fulfils the criteria and is therefore a valuable tool in the evaluation and research of different control algorithms in networked systems (Toivonen & Eriksson 2008; Nethi, Pohjola, Eriksson & Jäntti 2007a; Nethi et al. 2007b).

B.2 Key Features

Nethi et al. (2007b) detail five key features of the platform:

1. support for powerful control design and implementation tools provided by Matlab, Simulink and xPC Target enabling automatic code generation from Simulink models for real-time execution;

2. real-time control of a true or simulated process over a user-specified network;

3. capability to emulate any wired/wireless network readily available within NS-2;

4. easy-to-use network configuration tool; and

5. the platform is accessible over the Internet, i.e. it supports remote experimentation.

It should be evident that, constituted as a whole, it provides the possibility to integrate and extend the functionality of the available tools into a cohesive framework in which to test an implementation in a realistic networked control system.

B.3 Architecture

As Nethi et al. (2007a) go on to elaborate, the hardware consists of at least three computers connected through a network router:

Webserver, Database, xPC Host The server is responsible for maintaining connections between users and processes, running a reservation system for controlling processes.

xPC Target RTOS The computer controls the real process or simulates a process in real-time according to the user-specified control algorithms and is equipped with an I/O controller board.

NS-2 Network Simulator A customised version of NSE (Network Simulator Emulator) captures live packets transmitted by the xPC Target machine (as UDP control signals) over the network.

B.4 Network Simulator Integration

Essentially, PiccSIM is an extension to the functionality of the pre-existing MoCoNet (Monitoring and Controlling Laboratory Process over Internet) system at the Helsinki University of Technology designed to incorporate the capabilities of NS-2 to model wired/wireless networks. This section attempts to provide a broad overview as to how this integration is effected by paraphrasing the relevant sections in (Nethi et al. 2007a).

NS-2 is a discrete event-based simulator widely regarded as the de facto standard for simulating both wired and wireless networks. With an active research community dedicated to its ongoing evolution, facilities for the incorporation of new technologies are continually being made available. The Network Simulator Emulator (NSE) is one such extension to the core framework that provides basic utilities for reading/writing live packets from/to the network (see Figure B.1 for a representation of its emulation model). Thus forming the very basis on which an integration of distributed systems can proceed as called for by

the circumstances of desiring a better vehicle in which to conduct research in the field of wireless networked control systems (WiNCS).

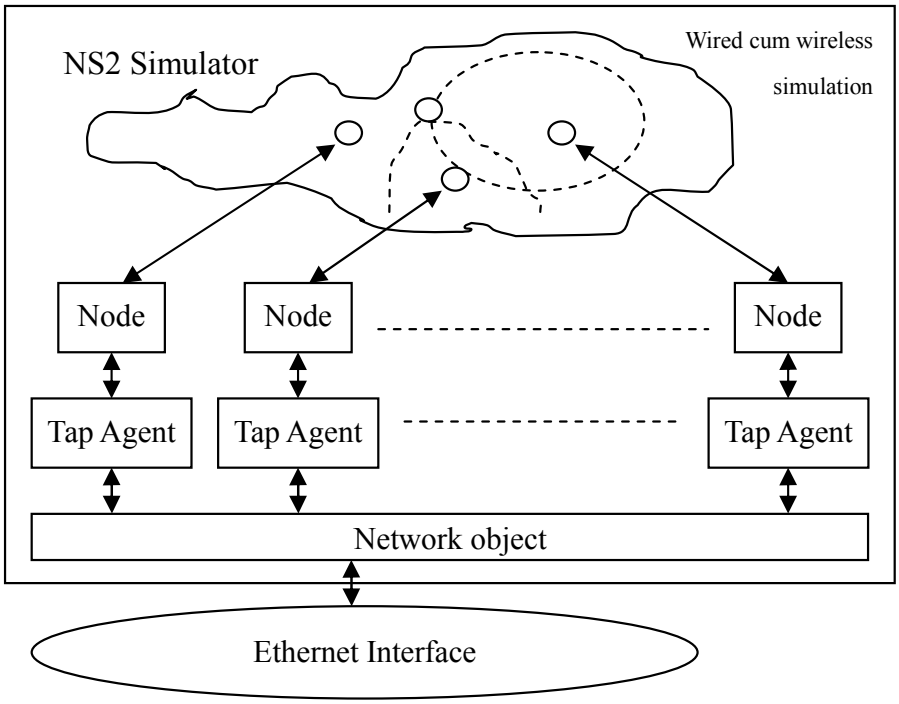

Figure B.1: Network Simulator Emulator (NSE): Flow Diagram Internals.

The components of which are listed below:

Real-time Scheduler The function of the soft real-time scheduler is to track, as closely as possible, the virtual event execution time within the simulator to real-time.

Tap Agent The tap agent injects live packets into the simulated network by generating simulator packets containing arbitrarily-assigned values within the common header. It uses the packet type field setting of `PT_LIVE` to indicate packets so generated. It is also responsible for the reverse operation of writing packets onto the network interface.

Network Objects Three types of objects are currently supported: `pcap/bpf`, `raw IP` and `UDP/IP`. The `pcap/bpf` code has been re-factored to support MoCoNet system requirements. Based on `libpcap`, `pcap/bpf` enables live packets to be captured from the Ethernet interface at the link layer (promiscuous mode access enabled). The BPF (Berkeley Packet Filter) provides sufficient matching capability to distinguish

packets from various processes and assign the proper mapping to the intended destination.

From another perspective, the process is measured and controlled by the xPC Target machine equipped with an I/O controller board. The measurement data is funnelled through the NS-2 simulator emulating the user-specified network. One of the nodes in the network is the process controller that calculates the control signal for the process. This is then transmitted over the network to the actuator in the process. The sensor and actuator nodes in the real process and those of the simulated network are associated by their UDP port numbers (see Figure B.2). In this way, the real process has an associated mapping with that of the corresponding simulated network.

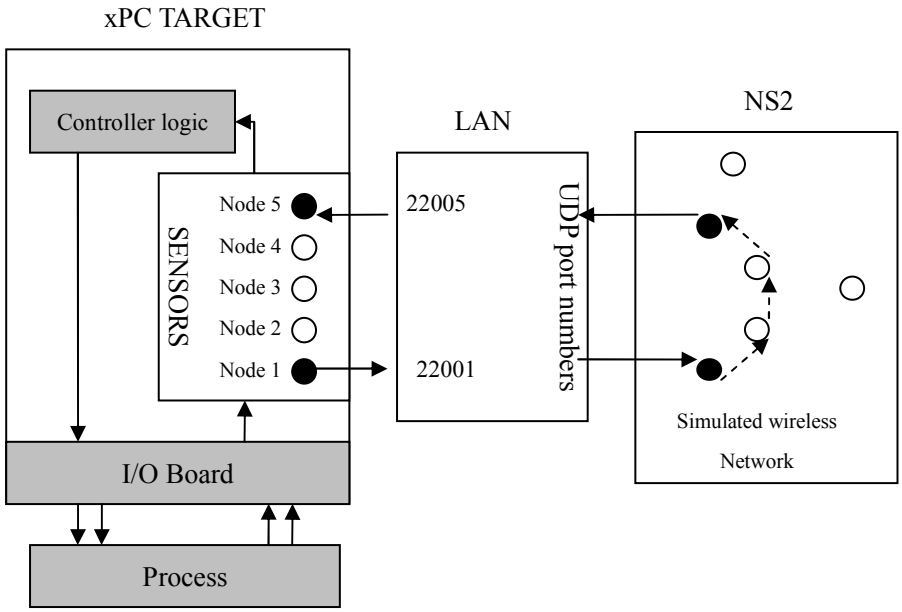

Figure B.2: UDP port mapping of xPC Target and NS-2 machine nodes.

B.5 A (Modified) Systematic Approach

The practical work of authors Chao (2008) & Zhonglei (2008), as part of their master's dissertations, was based upon a modified version of the unified framework afforded by the PiccSIM platform. The essential difference being the forgoing of the part played by the entity serving in the combined capacity of webserver, database and xPC Host. In effect, reducing the hardware requirement to two machines instead of three. This

reflects its intended use as a dedicated stand-alone platform rather than it facilitate remote experimentation by students and researchers alike off-campus as well as being a matter for practical expediency. A further qualification and a logical consequence of this decision is that since the machine is now both development host and cross-compilation target we are now no longer in the realm of real-time computing. Even so, the rapid prototyping tools built-in to the MathWorks Real-Time Workshop product allow for highly optimised code to be generated for non-real-time execution on the host computer. In the context of standalone simulations this is more than sufficient as the Rapid Simulation target generates an executable that runs fast. These provisions aside, the basic premise still holds that much is to be gained by factoring in the effects of uncertainty in wireless transmission. In particular, by the employment of co-simulation techniques in the manner (or similar) proposed by the PiccSIM platform.

These considerations are of import to the present work as it too is an implementation of control algorithms utilising the same platform albeit with certain qualifications of its own. In this case, however, it is due to a reconsideration of the possibility of remaining within the bounds of a reasonable time-frame and scope befitting that of a master's thesis. This was a realisation come to late in the course of proceeding with the practical work. As such, what this entailed was to limit even further the original intention of following in like manner as my two former classmates in the institution of a two-entity structure by transforming it into a purely standalone implementation. That said, the provision is there to expand upon the work at a later date into what was its goal in the beginning. Owing to its plug-in-like architecture, it simply needs to be 'plugged-in' so to speak. Of course, what this means in terms of the final analysis of the results is that a certain idealised scenario is engendered in that crucial factors inherent in wireless communications are not considered a la the conception of traditional control theory. In my defence, all that can be said is the situation calls for careful deliberation.

C PARETO OPTIMALITY

You might need to formulate problems with more than one objective, since a single objective with several constraints may not adequately represent the problem being faced. If so, there is a vector of objectives,

$$F(x) = [F_1(x), F_2(x), \ldots, F_m(x)],$$

that must be traded off in some way. The relative importance of these objectives is not generally known until the system's best capabilities are determined and trade-offs between the objectives fully understood. As the number of objectives increases, trade-offs are likely to become complex and less easily quantified. The designer must rely on his or her intuition and ability to express preferences throughout the optimisation cycle. Thus, requirements for a multi-objective design strategy must enable a natural problem formulation to be expressed, and be able to solve the problem and enter preferences into a numerically tractable and realistic design problem.

Multi-objective optimisation is concerned with the minimisation of a vector of objectives $F(x)$ that can be the subject of a number of constraints or bounds:

$$\begin{aligned} &\underset{x \in \mathbb{R}^n}{\text{minimise}}\ F(x) \\ &\qquad \text{s.t.} \quad G_i(x) = 0,\ i = 1, \ldots, k_e \\ &\qquad\qquad G_i(x) \leqslant 0,\ i = k_e + 1, \ldots, k \\ &\qquad\qquad lb \leqslant x \leqslant ub \end{aligned}$$

Note that because $F(x)$ is a vector, if any of the components of $F(x)$ are competing, there is no unique solution to this problem. Instead, the concept of non-inferiority (also called Pareto optimality) must be used to characterise the objectives. A non-inferior solution is one in which an improvement in one objective requires a degradation of another. To define this concept more precisely, consider a feasible region, Ω, in the parameter space. x is an element of the n-dimensional real numbers $x \in \mathbb{R}^n$ that satisfies all of the constraints, i.e.,

$$\begin{aligned} &\Omega = \{x \in \mathbb{R}^n\} \\ &\qquad \text{s.t.} \quad G_i(x) = 0,\ i = 1, \ldots, k_e \\ &\qquad\qquad G_i(x) \leqslant 0,\ i = k_e + 1, \ldots, k \\ &\qquad\qquad lb \leqslant x \leqslant ub \end{aligned}$$

This allows definition of the corresponding feasible region for the objective function space which we will term as Λ:

$$\Lambda = \{y \in \mathbb{R}^m : F(x),\, x \in \Omega\}\,.$$

Point $x^* \in \Omega$ is a non-inferior solution if for some neighbourhood of x^* there does not exist a Δx such that $(x^* + \Delta x) \in \Omega$ where:

$$\begin{aligned} F_i\,(x^* + \Delta x) &\leqslant F_i\,(x^*)\,,\ i = 1, \ldots, m, \quad \text{and} \\ F_j\,(x^* + \Delta x) &< F_j\,(x^*) \quad \text{for at least one } j. \end{aligned}$$

(The MathWorks, Inc. 2009a: cited with only minor modifications)

In other words, if any of the above conditions are violated, the solution x^* does not dominate the solution $(x^* + \Delta x)$ for some Δx. To be non-inferior, it must be *no worse* than the alternatives in all objectives and *strictly better* in at least one objective (Deb 2001: p. 28).

Let us consider a two-objective (both as minimisation functions) optimisation problem with three different solutions shown in the objective function space, as illustrated in Figure C.1. The shaded area of the graph representing the feasible region.

It is clear by examination and application of our definition of a non-inferior solution that not all solutions within the feasible region are optimal. By direct comparison, p_1, as a solution, is better than p_3 in the second objective and p_1 is no worse than p_3 in the first objective (in fact, they are equal). By this observation, p_1 satisfies the conditions for non-inferiority whereas p_3 is evidently an inferior solution on account of failing both criteria (Deb 2001: p. 29). Since any point in objective function space that is an inferior solution represents a point in which improvement can be attained in all the objectives, it is clear that such a point is of no value (The MathWorks, Inc. 2009a).

By the same token, comparing p_1 with p_2 we find that each is better in one objective than the other while at the same time being worse in some other objective. Reflecting the situation where an improvement in one objective comes at the cost of degradation in that of

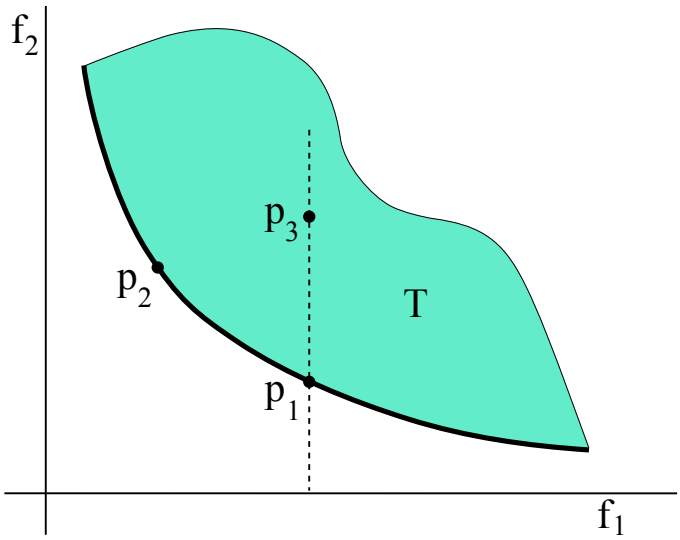

Figure C.1: Two Pareto-optimal solutions and one non-optimal solution for a two-objective optimisation problem.

the other objective. Thus justifying their classification as non-inferior solution points (The MathWorks, Inc. 2009a). There exist many such solutions in the search space. For clarity, these solutions are joined with a continuous curve in the figure. All non-inferior solutions lying on this curve are special in the context of multi-objective optimisation in that they form the *Pareto-optimal set*. The curve formed by joining these solutions is known as a *Pareto-optimal front*. It is interesting to note that this front lies in the bottom-left corner of the search space for problems where all objectives are to be minimised (Deb 2001: p. 20).

In summary, since the concept of domination allows a way to effectively compare solutions with multiple objectives, most multi-objective optimisation methods either explicitly or implicitly use it as a means to search for non-dominated solutions (Deb 2001: p. 29). Moreover, because of the presence of conflicting multiple objectives, a multi-objective optimisation problem results in a number of optimal solutions, known as Pareto-optimal solutions. Faced with multiple objectives, ideally a user is interested in finding multiple Pareto-optimal solutions. Thus, there are two tasks of an ideal multi-objective optimisation algorithm, namely (i) to find multiple Pareto-optimal solutions and (ii) to seek for Pareto-optimal solutions with a good diversity in objective and/or decision variable values (Deb 2001: p. 46). In essence, then, multi-objective optimisation is, therefore, concerned with the generation and selection of non-inferior solution points (The MathWorks, Inc. 2009a).

D GENETIC ALGORITHM PARAMETERS

As the present appendix serves as an adjunct to the preparatory material contained in that of §5.6, a modicum of an elucidation as to its content and manner of presentation is a requisite condition in order for the connections to become clear.

The following descriptions of parameters and options are as that current for the Genetic Algorithm and Direct Search (GADS) toolbox version at the time of release of Matlab R2009a. Although considered a separate product in and of itself, it is in every way compatible with the Optimization toolbox where its functions appear as selections available to the user within the GUI for the same — Optimization Tool (`optimtool`). As such, it is therefore possible to enact the same functionality, with restrictions depending on the mode of invocation, either programmatically via the command-line environment or an interactive session by use of the Optimization Tool. It is the user's choice as to which suits their purposes best in each instance. In what follows, therefore, each option and parameter's presentation makes the distinction between modes by indicating in **bold** when referring to the label as it would appear in the Optimization Tool and in `monospaced` font for the corresponding field of the `options` structure as that applicable in a programmatic usage (e.g. **Population type** and `PopulationType`).

Essentially, the content, structure and manner of presentation is as that in The MathWorks, Inc. (2009a), with the exception that it is an abridged version of those options and parameters that were of direct consequence in the present implementation of the proposed GEA. Direct quotes are not used for the cited material as such would be cumbersome; besides which, somewhat of a distraction and should be assumed as given. It was thought useful to take such an approach so that the individual relevance and meaning of each option and parameter is made the more clear by making immediate reference to the associated entry in the user's guide rather than simply leaving it to chance that the purpose of such is transparent. Where appropriate, we provide additional comments to further clarify the specific context due to our problem formulation. A final note as to convention is that of an indication as to which pass of the proposed GEA the respective parameter and option applies: (1) is indicative of the first-pass and (2) that of the second. In the event they are common to both, we use the notation $(1)\&(2)$ to designate such.

D.1 Population Options

Population options enable you to specify the parameters of the population that the genetic algorithm uses.

Population size (`PopulationSize`) *specifies how many individuals there are in each generation. With a large population size, the genetic algorithm searches the solution space more thoroughly, thereby reducing the chance that the algorithm will return a local minimum that is not a global minimum. However, a large population size also causes the algorithm to run more slowly.*

If you set `PopulationSize` *to a vector, the genetic algorithm creates multiple subpopulations, the number of which is the length of the vector. The size of each subpopulation is the corresponding entry of the vector.*

$$\begin{aligned} (1) \qquad & \text{no. of populations} = 25, \\ & \text{size of subpopulation} = 10, \\ & \text{total population} = 250. \\ (2) \qquad & \text{no. of populations} = 3, \\ & \text{size of subpopulation} = 30, \\ & \text{total population} = 90. \end{aligned}$$

Comments. Manipulation of the respective 'no. of populations' and 'size of subpopulation' parameters yielded the greatest impact on performance times; an observation with which the user guide readily concurs as stated. (1) showed far greater tolerance in values than (2) and therefore a larger overall 'total population' could be utilised in this scenario. The use of subpopulations over that of a single population as a consolidated unit was of considerable benefit in both cases as we are then able to leverage the technique of migration between populations (see Appendix D.5) in the propagation of the more promising combinations of decision variables as individuals in an exploration of the search space. As such, it could well be considered as perhaps the single most important parameter in general for the execution of the algorithm.

Creation function (`CreationFcn`) *specifies the function that creates the initial population for* `ga`.

> **Feasible population** (`@gacreationlinearfeasible`) creates a random initial population that satisfies all bounds and linear constraints. It is biased to create individuals that are on the boundaries of the constraints, and to create well-dispersed populations. This is the *default* if there are linear constraints.

(1)&(2) `@gacreationlinearfeasible`

Comments. For our purposes, the specification of a lower and upper bound on the decision variables (i.e. pursuant to our definition of the range assignment function) is sufficient to initiate the default action of `@gacreationlinearfeasible` as the means of creating an initial population without explicitly stating such. The proviso that the creation function is biased toward the boundary conditions on the constraints is of particular relevance when it comes to a consideration of the determination of settings for the `PopInitRange` parameter to appear later in the present section.

Initial population (`InitialPopulation`) *specifies an initial population for the genetic algorithm. The default value is* `[]`*, in which case* `ga` *uses the default* `CreationFcn` *to create an initial population. If you enter a nonempty array in the* `InitialPopulation` *field, the array must have no more than* `PopulationSize` *rows, and exactly* 'Number of variables' *columns. In this case, the genetic algorithm calls a* `CreationFcn` *to generate the remaining individuals, if required.*

(1) `[]`

(2) 1^{st}-pass optimal set of solutions

Comments. A detailed explanation of the usage of an initial population in (2) is given within §5.6.

Initial range (`PopInitRange`) *specifies the range of the vectors in the initial population that is generated by a creation function. You can set* `PopInitRange` *to be a matrix with two rows and* 'Number of variables' *columns, each column of which has the form* `[lb;ub]`*, where* `lb` *is the lower bound and* `ub` *is the upper bound for the entries in that coordinate. If you specify* `PopInitRange` *to be a 2-by-1 vector, each entry is expanded to a constant row of length* 'Number of variables'.

(1) $\left[\quad 0.25 \times r_{max} \quad ; \quad r_{max} \quad \right]$

(2) $\left[\quad \min\{1.05 \times \min\{\texttt{InitialPopulation}\}, r_{max}\} \quad ; \right.$
$\left. \quad \min\{1.05 \times \max\{\texttt{InitialPopulation}\}, r_{max}\} \quad \right]$

Comments. Given the default creation function, `@gacreationlinearfeasible`, is biased toward the boundary conditions on the constraints, as stated earlier, the parameter settings as they appear here are for the purpose of steering initialisation toward non-zero values in the decision variables, as long as they are not in excess of the effective r_{max} value. The reason being that it is unlikely that connectivity-based objectives are to be achieved at the lower end of the scale (i.e. allowable transmitting range) and hence it is more productive to begin the search at that portion more conducive to a convergence toward a solution. Further to which, the specific values of the constant factors in (2) were derived in an examination

as to their effect in borderline scenarios where connectivity would be established if but one communication link would conjoin, yet is unable due to insufficient transmitting range at the upper end of the scale. As such, they can be construed as performing an assisting role within the optimisation.

D.2 Selection Options

Selection options specify how the genetic algorithm chooses parents for the next generation.

Selection function (`SelectionFcn`) *specifies the function the algorithm uses for the process of selection.*

> **Tournament** (`@selectiontournament`) selection chooses each parent by choosing *'Tournament size'* players at random and then choosing the best individual out of that set to be a parent. *'Tournament size'* must be at least 2. The *default* value of *'Tournament size'* is 4.
>
> To change the default value of *'Tournament size'* at the command line, use the syntax
>
> ```
> options = gaoptimset('SelectionFcn', {@selectiontournament, size})
> ```
>
> where `size` is the value of *'Tournament size'*.
>
> (1) {`@selectiontournament`, 4}
>
> (2) {`@selectiontournament`, 8}

Comments. As we wish for there to be a reasonable chance that one or both parents be of good fitness value while still allowing for novelty, tournament selection of sufficient size affords an acceptable compromise.

D.3 Mutation Options

Mutation options specify how the genetic algorithm makes small random changes in the individuals in the population to create mutation children. Mutation provides genetic diversity and enables the genetic algorithm to search a broader space.

Mutation function (`MutationFcn`) *specifies the function the algorithm uses for the process of mutation.*

> **Adaptive feasible** (`@mutationadaptfeasible`) randomly generates directions that are adaptive with respect to the last successful or unsuccessful generation. The feasible region is bounded by the constraints and inequality constraints. A step length is chosen along each direction so that linear constraints and bounds are satisfied.

(1)&(2) `@mutationadaptfeasible`

Comments. None

D.4 Crossover Options

Crossover options specify how the genetic algorithm combines two individuals, or parents, to form a crossover child for the next generation.

Crossover function (`CrossoverFcn`) *specifies the function that performs the crossover.*

> **Intermediate** (`@crossoverintermediate`) creates children by taking a weighted average of the parents. You can specify the weights by a single parameter, *'Ratio'*, which can be a scalar or a row vector of length *'Number of variables'*. The *default* is a vector of all 1's. The function creates the child from `parent1` and `parent2` using the following formula:
>
> ```
> child = parent1 + rand * Ratio * (parent2 - parent1)
> ```
>
> If all the entries of *'Ratio'* lie in the range $[0, 1]$, the children produced are within the hypercube defined by placing the parents at opposite vertices. If *'Ratio'* is not in that range, the children might lie outside the hypercube. If *'Ratio'* is a scalar, then all the children lie on the line between the parents.
>
> To change the default value of *'Ratio'* at the command line, use the syntax
>
> ```
> options = gaoptimset('CrossoverFcn', {@crossoverintermediate, ratio})
> ```
>
> where `ratio` is the value of *'Ratio'*.

(1)&(2) {`@crossoverintermediate`, 1}

Comments. By apportioning true novel solution creation to that of the mutation function (see Appendix D.3), the role of the crossover function is to explore the search space more fully within the region formed by those candidate solutions already derived via combinatorial ways and means.

D.5 Migration Options

Migration options specify how individuals move between subpopulations. Migration occurs if you set `PopulationSize` to be a vector of length greater than 1. When migration occurs, the best individuals from one subpopulation replace the worst individuals in another subpopulation. Individuals that migrate from one subpopulation to another are copied. They are not removed from the source subpopulation.

Direction (`MigrationDirection`) *can take place in one or both directions.*

> If set to `forward`, migration takes place toward the last subpopulation. That is, the n^{th} subpopulation migrates into the $(n+1)^{th}$ subpopulation.
>
> If set to `both`, the n^{th} subpopulation migrates into both the $(n-1)^{th}$ and $(n+1)^{th}$ subpopulations.

Migration wraps at the ends of the subpopulations. That is, the last subpopulation migrates into the first, and the first may migrate into the last.

(1) `both`

(2) `forward`

Comments. Since (2) has relatively few 'no. of populations' (see Appendix D.1), as indeed that of `Generations` (see Appendix D.7), it was found important to be mindful that the individual subpopulations be given a measure of isolation from each other in order that promising directions in an exploration of the search space not unduly be cut short by the infiltration of those less so inclined. Hence, only the subpopulations of (1) migrate in both directions.

Interval (`MigrationInterval`) *specifies how many generations pass between migrations. For example, if you set* `MigrationInterval` *to* 20*, migration takes place every* 20 *generations.*

(1)&(2) 1

Comments. A setting of the parameter to a value of one ensures migration occurs in each and every generation in addition to the rapid elimination of inferior solutions within the population. An especially important outcome in the case of (2) given its paucity in number of `Generations` (see Appendix D.7).

Fraction (`MigrationFraction`) *specifies how many individuals move between subpopulations.* `MigrationFraction` *specifies the fraction of the smaller of the two subpopulations that moves. For example, if individuals migrate from a subpopulation of* 50 *individuals into a subpopulation of* 100 *individuals and you set* `MigrationFraction` *to* 0.1*, the number of individuals that migrate is* `0.1 * 50 = 5`.

(1)&(2) 0.4

Comments. Retain the better part of a subpopulation in each generation but still recognitive of the stimulating effect produced by an influx in new compositions of individuals.

D.6 Multi-Objective Options

Multi-objective options define parameters characteristic of the multi-objective genetic algorithm — `gamultiobj`.

Pareto fraction (`ParetoFraction`) *sets the fraction of individuals to keep on the first Pareto front while the solver selects individuals from higher fronts. This option is a scalar between* 0 *and* 1.

(1) 0.50

(2) 0.25

Comments. By setting the Pareto fraction to a fairly low value in both cases we do not unnecessarily limit ourselves to a search in areas of local optima and therefore broaden it to potential regions closer to the global optimum than that already arrived at.

D.7 Stopping Criteria Options

Stopping criteria determine what causes the algorithm to terminate.

Generations (`Generations`) *specifies the maximum number of iterations for the genetic algorithm to perform. The default is* 100.

(1) 5

(2) 3

Comments. As to be expected, directly proportional to the elapsed run-time of the algorithm. Consequently, it is essential to set the parameter carefully according to the trade-offs between performance, convergence toward a solution, etc.

Stall generations (`StallGenLimit`) — *The algorithm stops if the weighted average change in the fitness function value over* `StallGenLimit` *generations is less than* `TolFun`.

(1) 5

(2) 3

Comments. By setting the number of stall generations to be equal to the respective `Generations` parameter in each case, it functions more as a confirmatory device in evaluation of a particular instance of an invocation of the algorithm than that of an absolute necessity as a stopping criterion.

Function tolerance (`TolFun`) — *The algorithm runs until the cumulative change in the fitness function value over* `StallGenLimit` *generations is less than or equal to* `TolFun`.

(1)&(2) `default`

Comments. None

D.8 Vectorize Option

Objective function is vectorized (`Vectorized`) *specifies whether the computation of the fitness function is vectorized.*

If set to `On`, indicates that the fitness function is vectorized.

If set to `Off`, the algorithm calls the fitness function on one individual at a time as it loops through the population.

(1)&(2) `On`

Comments. Along with `PopulationSize` (see Appendix D.1), one of the more important considerations when it comes to evaluating the correct settings for the parameters as it pertains to our problem formulation. In this case, however, merely enabling the option is not sufficient, for the fitness function has to be designed and implemented accordingly with that purpose in mind from start to finish. It is well worth the effort, though, in that the net result is a significant reduction in the total number of function calls required of the algorithm in a single pass being directly proportional to the size of the total population as well as in a commensurate boost in performance.

E A STEADY-STATE MOBILITY FILE GENERATOR

The present appendix comprises of supporting material for the discussion as that in §6.2 and as such does in no way constitute a self-contained piece in its own right.

Listing E.1: `README` file packaged with the `mobgen-ss` software distribution that succinctly describes the core functionality and methods of invoking the program that obviates the need to consult the cited references mentioned in its opening passage.

```
/***************************************************************************
*   Copyright (C) 2004  Toilers Research Group -- Colorado School of Mines
*
*   Please see COPYRIGHT.TXT and LICENSE.TXT for copyright and license
*   details.
***************************************************************************/


=========================================================================
IMPLEMENTATION OF THE STEADY-STATE RANDOM WAYPOINT MOBILITY MODEL
=========================================================================

Original mobgen program written by Jeff Boleng <jeff@boleng.com>
(Ph.D. 2002 from the Colorado School of Mines).

Program modified to create mobgen-ss by Nick Bauer (M.S. 2004
from the Colorado School of Mines).

The papers related to the code:
        * W. Navidi and T. Camp, Stationary Distributions for the Random
        Waypoint Mobility Model, IEEE Transactions on Mobile Computing,
        vol. 3, no. 1, pp. 99-108, January-March 2004.

        * W. Navidi, T. Camp, and N. Bauer, Improving the Accuracy of
        Random Waypoint Simulations Through Steady-State Initialization,
        Proceedings of the 15th International Conference on Modeling and
        Simulation (MS '04), pp. 319-326, March 2004.

We are happy to share our code with you. We only ask that any published
research from using our code include a reference to the appropriate
preceding paper.

If you have any questions on the code, send email to
Tracy Camp <tcamp@mines.edu>.

The research group's URL is:  http://toilers.mines.edu

=========================================================================

The mobgen-ss program is used to generate mobility files for NS2, gnuplot,
and QualNet.  The mobility files use the random waypoint model, where a node
picks a random point on the simulation area and a random speed and then
travels to that point at the chosen speed.  Once it arrives, the node pauses
for a randomly chosen pause time, and then repeats the process until the
simulation ends.  The mobgen-ss program picks initial node positions,
speeds, and pause times according to the steady-state distributions of the
random waypoint model, so that no time is required to let the distributions
of position and speed settle at the beginning of the simulation.

To compile the mobgen-ss program, type "make" and the makefile will compile
the program.
(The program has been tested under
gcc version 3.3.2 and gcc 2.96)

To execute the mobgen-ss program, type:
```

```
mobgen-ss <number of nodes>
          <max-x> <max-y> <end time>
          <speed mean> <speed delta>
          <pause time> <pause time delta>
          <'N' or 'G' or 'Q'>
          'N' implies NS2 mobility file
          'G' implies gnuplot path file
          'Q' implies QualNet mobility file

where <number of nodes> is the number of nodes in the simulation, <max-x>
and <max-y> are the lengths of the sides of the simulation rectangle, and
<end time> is the length of the simulation.  The nodes will pick speeds from
a uniform distribution centered at <speed mean> with a range of <speed
delta> on either side.  The last command line argument is a single letter
indicating the type of mobility file to generate, where 'N' implies NS2
mobility file, 'G' implies gnuplot path file, and 'Q' implies QualNet
mobility file.

The permitted values of input parameters are:
        <number of nodes> must be greater than 0
        <max-x> and <max-y> must be greater than 0
        <end time> must be greater than or equal to 0
        <speed mean> must be greater than 0
        <speed delta> must be greater than or equal to 0
                 and less than <speed mean>
        <pause mean> must be greater than or equal to 0
        <pause delta> must be greater than or equal to 0
                and less than or equal to <pause mean>

NOTES: The minimum speed must be positive, because the steady-state
distribution is degenerate when the minimum speed is 0. An end time
of zero only gives the initial configuration of the nodes without
any movement.

================================================================================
```

Listing E.2: NS-2 compatible format output file generated by `mobgen-ss` utilising the input parameters as per that contained in the comment header which has been reproduced in tabular form with additional comments as appropriate (see Table 6.1).

```
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#       Steady-state Random Waypoint Model
#       numNodes   =     20
#       maxX       =    500.00
#       maxY       =    500.00
#       endTime    =    150.00
#       speedMean  =   10.0000
#       speedDelta =    9.9990
#       pauseMean  =     30.00
#       pauseDelta =     30.00
#       output     =      N
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# output format is NS2
#       Initial positions:
$node_(0) set X_ 436.443997304836
$node_(0) set Y_ 426.678879492604
$node_(0) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(0) setdest 487.785307451983 491.377178342723 0.000000000000"
$node_(1) set X_ 194.245701777755
$node_(1) set Y_ 334.589142462149
$node_(1) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(1) setdest 8.885554507787 264.319049084708 0.000000000000"
$node_(2) set X_ 119.234108806493
$node_(2) set Y_ 373.453915540379
$node_(2) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(2) setdest 109.672209066186 378.273550597147 0.181928659220"
$node_(3) set X_ 101.591601306835
```

```
$node_(3) set Y_ 154.648546633315
$node_(3) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(3) setdest 38.291780761579 81.226361021971 0.018570305284"
$node_(4) set X_ 306.423069995826
$node_(4) set Y_ 150.125028973128
$node_(4) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(4) setdest 364.278237970675 86.141902993499 0.000000000000"
$node_(5) set X_ 461.137570033640
$node_(5) set Y_ 49.969843818498
$node_(5) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(5) setdest 320.699682841403 97.976031525981 0.015667872212"
$node_(6) set X_ 406.245334300116
$node_(6) set Y_ 139.606758761079
$node_(6) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(6) setdest 459.739302499098 73.811601416120 0.006386744819"
$node_(7) set X_ 247.461210066330
$node_(7) set Y_ 392.372505409044
$node_(7) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(7) setdest 357.387163609912 462.905354082075 0.028979243103"
$node_(8) set X_ 207.817300933998
$node_(8) set Y_ 118.174015283431
$node_(8) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(8) setdest 28.477370752244 304.277354061733 0.533984385190"
$node_(9) set X_ 372.223982748172
$node_(9) set Y_ 213.817904609728
$node_(9) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(9) setdest 213.757726929038 293.626880642784 0.000000000000"
$node_(10) set X_ 222.760134540543
$node_(10) set Y_ 303.441170731994
$node_(10) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(10) setdest 428.671942757756 159.552387501836 0.000000000000"
$node_(11) set X_ 101.362404762062
$node_(11) set Y_ 316.516902814941
$node_(11) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(11) setdest 50.877364841698 257.760546988696 0.035057547470"
$node_(12) set X_ 146.137730431189
$node_(12) set Y_ 281.979662171990
$node_(12) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(12) setdest 167.326778251364 144.819768911609 0.000000000000"
$node_(13) set X_ 246.516241698587
$node_(13) set Y_ 323.667519937716
$node_(13) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(13) setdest 385.487563854776 0.804102514314 0.002051057846"
$node_(14) set X_ 165.384008312118
$node_(14) set Y_ 120.280206209987
$node_(14) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(14) setdest 444.026772838098 422.324961946497 0.002242095503"
$node_(15) set X_ 491.670924273176
$node_(15) set Y_ 418.748778985511
$node_(15) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(15) setdest 498.766927047990 441.333637545506 0.682827373271"
$node_(16) set X_ 177.217493029494
$node_(16) set Y_ 355.349709727971
$node_(16) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(16) setdest 3.330857727365 421.417867961069 0.000000000000"
$node_(17) set X_ 281.640801728529
$node_(17) set Y_ 235.867335172243
$node_(17) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(17) setdest 279.646766735076 249.355707433706 0.001134224619"
$node_(18) set X_ 202.109213962509
$node_(18) set Y_ 375.409825752103
$node_(18) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(18) setdest 483.673361588117 185.457851824098 0.000000000000"
$node_(19) set X_ 211.098916711327
$node_(19) set Y_ 172.883609210691
$node_(19) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(19) setdest 193.628103562458 127.252868668760 0.000000000000"
```

```
#       Movements:
$ns_ at 2.414615616277 "$node_(16) setdest 462.754571327360 446.833546481483 11.420830087721"
$ns_ at 2.473598833371 "$node_(0) setdest 405.729586214633 295.283749371433 14.892405752876"
$ns_ at 2.893171792593 "$node_(18) setdest 388.138421293413 206.253781312263 18.379434723645"
$ns_ at 5.074191321574 "$node_(10) setdest 316.603432556895 238.276641461196 10.289262405340"
$ns_ at 6.272441806577 "$node_(1) setdest 50.921086711307 49.946268345204 13.301864216312"
$ns_ at 11.534406696045 "$node_(0) setdest 405.729586214633 295.283749371433 0.000000000000"
$ns_ at 16.177964988844 "$node_(10) setdest 316.603432556895 238.276641461196 0.000000000000"
$ns_ at 16.573523760016 "$node_(18) setdest 388.138421293413 206.253781312263 0.000000000000"
$ns_ at 18.195648085881 "$node_(4) setdest 303.413874610986 380.925305597915 6.617338080137"
$ns_ at 20.329298756836 "$node_(10) setdest 350.470843189615 147.261153276666 9.129931690899"
$ns_ at 26.114717346658 "$node_(19) setdest 167.623757462773 274.514021945425 16.874281330296"
$ns_ at 28.667915754622 "$node_(16) setdest 462.754571327360 446.833546481483 0.000000000000"
$ns_ at 30.230760784238 "$node_(1) setdest 50.921086711307 49.946268345204 0.000000000000"
$ns_ at 30.368042579309 "$node_(12) setdest 281.612711391231 73.292162769144 12.713821113940"
$ns_ at 30.966006175013 "$node_(10) setdest 350.470843189615 147.261153276666 0.000000000000"
$ns_ at 32.665444796528 "$node_(19) setdest 167.623757462773 274.514021945425 0.000000000000"
$ns_ at 33.271884796520 "$node_(9) setdest 186.336470388964 23.730006778487 17.823799094991"
$ns_ at 34.669646439017 "$node_(15) setdest 498.766927047990 441.333637545506 0.000000000000"
$ns_ at 48.188533666943 "$node_(9) setdest 186.336470388964 23.730006778487 0.000000000000"
$ns_ at 49.937692616787 "$node_(12) setdest 281.612711391231 73.292162769144 0.000000000000"
$ns_ at 53.076727166317 "$node_(4) setdest 303.413874610986 380.925305597915 0.000000000000"
$ns_ at 57.125520640888 "$node_(9) setdest 468.260429784777 494.908501624553 15.642568656274"
$ns_ at 58.857606044688 "$node_(2) setdest 109.672209066186 378.273550597147 0.000000000000"
$ns_ at 60.078995713314 "$node_(19) setdest 474.875644303335 476.593859017172 0.729195089094"
$ns_ at 64.638832275933 "$node_(10) setdest 140.300638573384 280.007733628157 19.217600234167"
$ns_ at 64.762903789656 "$node_(1) setdest 368.546755457552 130.478576584942 14.455088747549"
$ns_ at 69.130463817499 "$node_(0) setdest 290.420975904176 298.102334047715 0.346945834002"
$ns_ at 72.044286243546 "$node_(18) setdest 231.507629031086 85.870192193366 11.326408655732"
$ns_ at 72.559164053996 "$node_(4) setdest 13.120340189487 159.162354962510 4.041229760359"
$ns_ at 77.535747215617 "$node_(12) setdest 241.566597829371 104.776893558342 15.264382709108"
$ns_ at 77.573970708472 "$node_(10) setdest 140.300638573384 280.007733628157 0.000000000000"
$ns_ at 80.872989350543 "$node_(12) setdest 241.566597829371 104.776893558342 0.000000000000"
$ns_ at 81.598314129851 "$node_(16) setdest 142.434091606380 495.280392931439 0.396890444083"
$ns_ at 83.284209351216 "$node_(15) setdest 33.512415612821 463.540822716216 6.890386668059"
$ns_ at 85.729074727292 "$node_(10) setdest 475.511826563399 481.747520613367 6.686747089722"
$ns_ at 87.431456043062 "$node_(1) setdest 368.546755457552 130.478576584942 0.000000000000"
$ns_ at 89.485693800458 "$node_(18) setdest 231.507629031086 85.870192193366 0.000000000000"
$ns_ at 92.227256759496 "$node_(9) setdest 468.260429784777 494.908501624553 0.000000000000"
$ns_ at 104.475780525408 "$node_(1) setdest 232.113418975898 323.598930064402 11.293337690347"
$ns_ at 111.897535358757 "$node_(2) setdest 362.591995560840 332.248440865543 6.035733721627"
$ns_ at 112.708581903486 "$node_(12) setdest 160.694329375724 115.411973379279 17.651411073891"
$ns_ at 117.329660183228 "$node_(12) setdest 160.694329375724 115.411973379279 0.000000000000"
$ns_ at 125.413076461208 "$node_(1) setdest 232.113418975898 323.598930064402 0.000000000000"
$ns_ at 138.067104513578 "$node_(12) setdest 185.932376275739 405.726876531600 8.657247056949"
$ns_ at 143.738376070126 "$node_(18) setdest 427.498974105110 10.503769857112 3.922629756745"
$ns_ at 144.238217420251 "$node_(10) setdest 475.511826563399 481.747520613367 0.000000000000"
$ns_ at 147.692525464288 "$node_(9) setdest 20.402021017113 308.443840736730 7.712051966204"
```

F SIMULINK MODEL ARCHITECTURE

F.1 Root Block Diagram

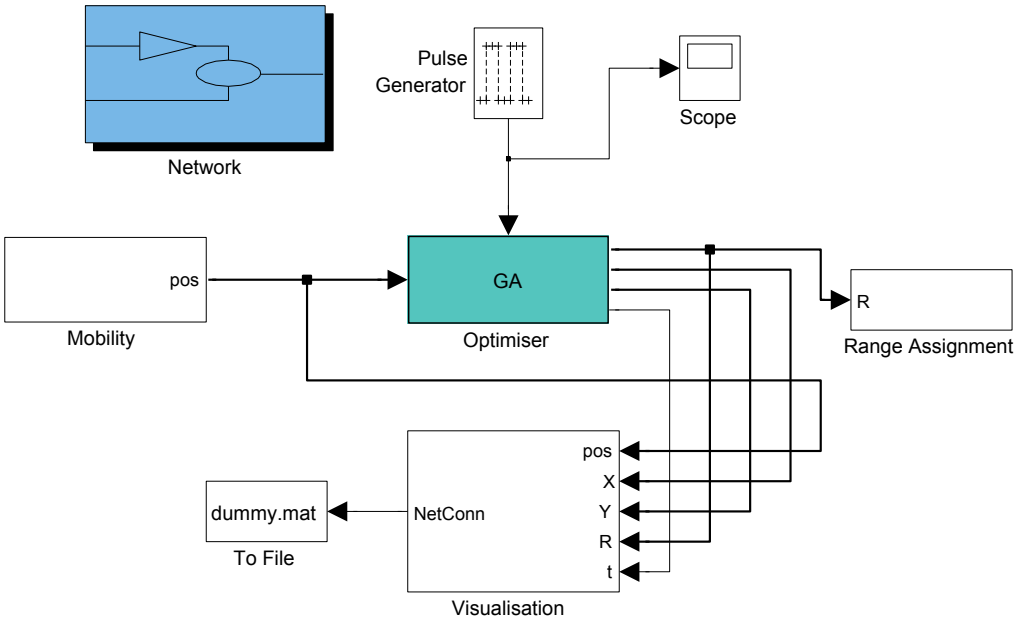

Figure F.1: Root block diagram for the topology control optimisation simulation model.

F.2 Subsystem Component Block Diagrams

F.2.1 Mobility Subsystem

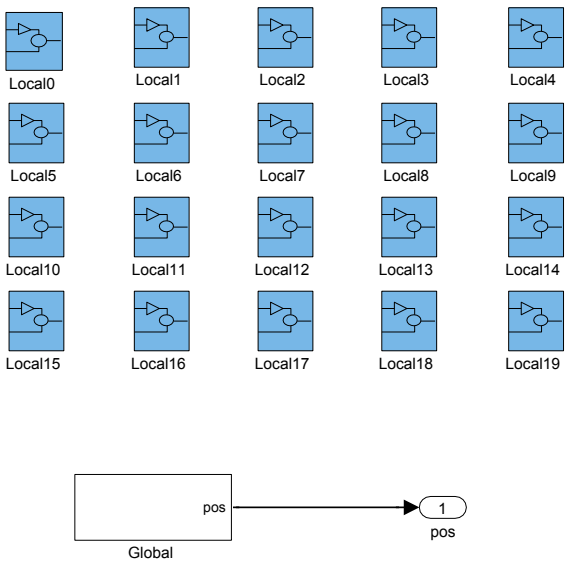

Figure F.2: Mobility subsystem component block diagram.

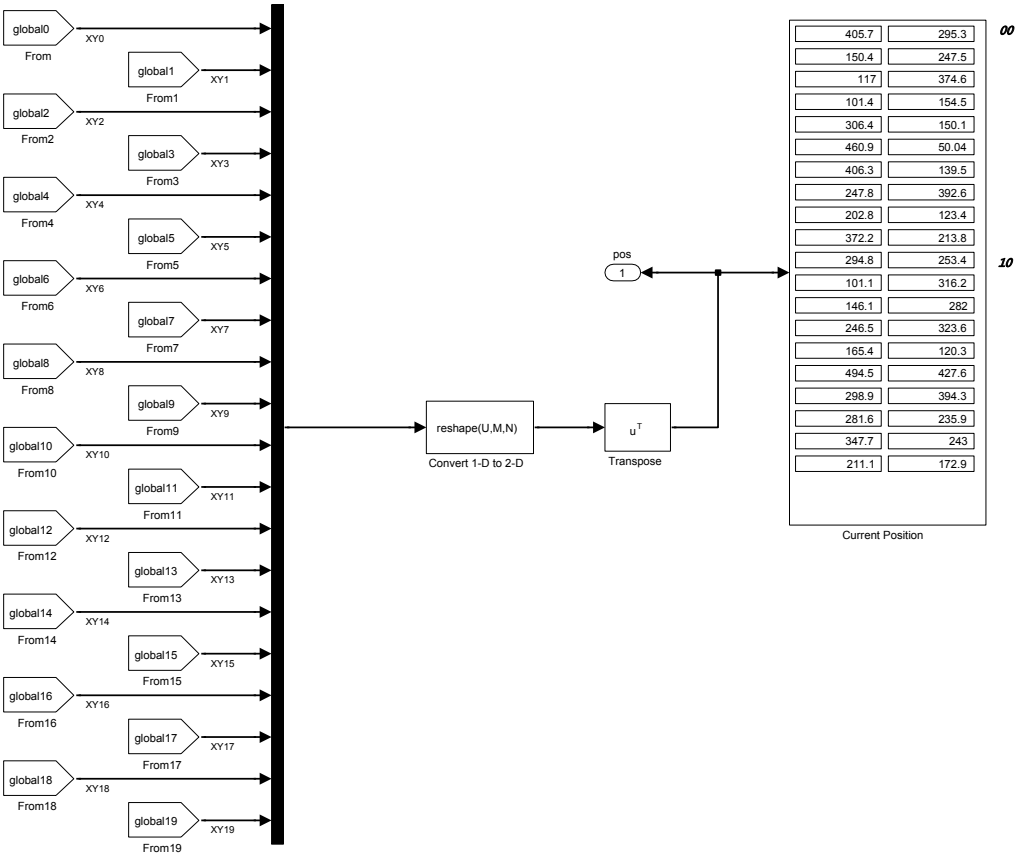

Figure F.3: Mobility subsystem global sub-component block diagram.

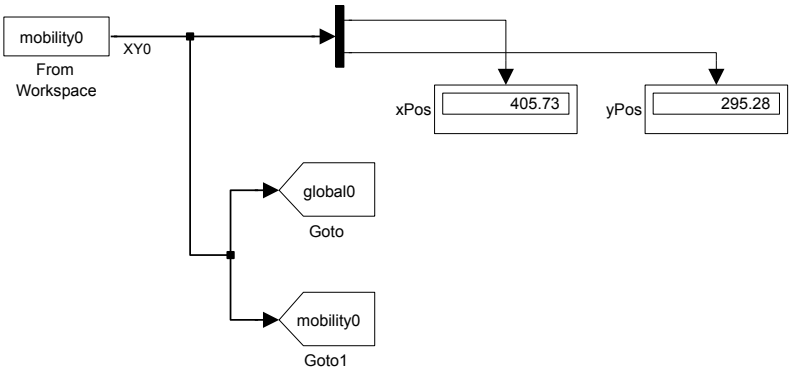

Figure F.4: Mobility subsystem local sub-component block diagram replicated as many times as there are nodes constituting the network.

F.2.2 Network Subsystem

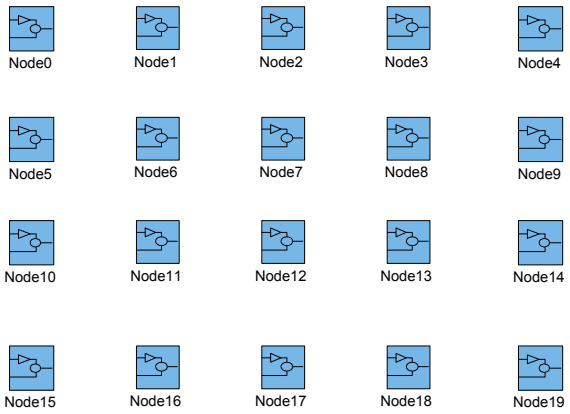

Figure F.5: Network subsystem component block diagram.

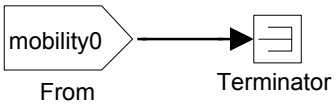

Figure F.6: Network subsystem node sub-component block diagram replicated as many times as there are nodes constituting the network.

F.2.3 Optimiser Subsystem

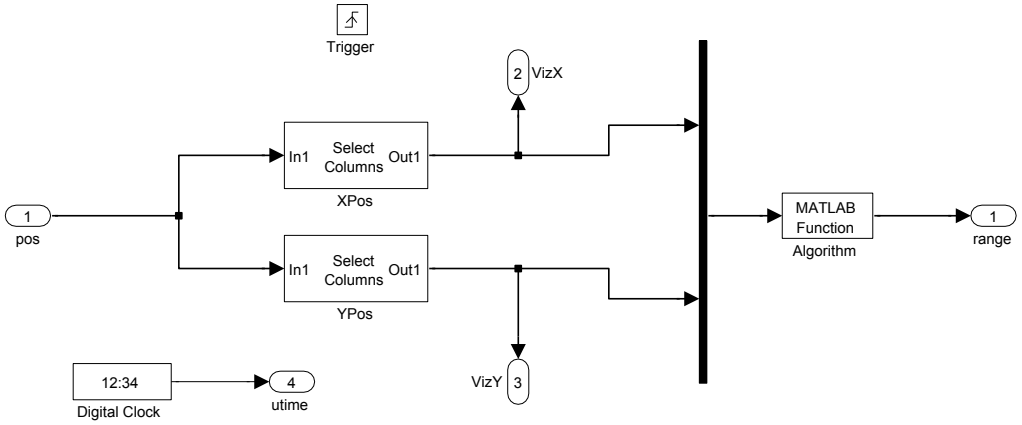

Figure F.7: Optimiser subsystem component block diagram.

F.2.4 Range Assignment Subsystem

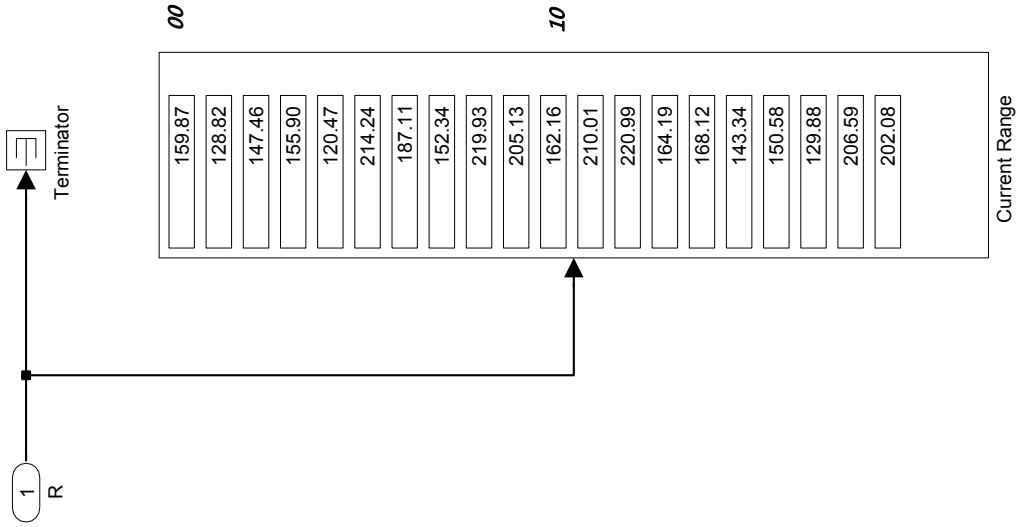

Figure F.8: Range assignment subsystem component block diagram.

F.2.5 Visualisation Subsystem

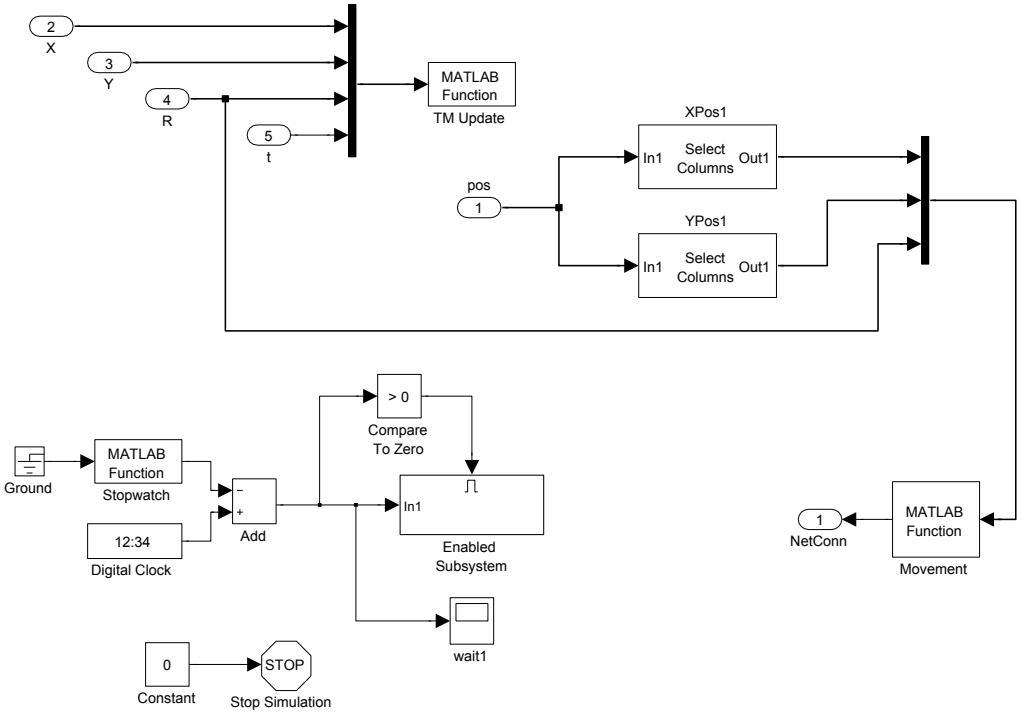

Figure F.9: Visualisation subsystem component block diagram.