Backbones for Military Multilayer Wireless Ad hoc Networks based on Network Science Concepts

A dissertation

submitted in fulfilment

of the requirements for the degree

of

Doctor of Philosophy in Electrical & Computer Engineering

at

The University of Thessaly

by

Dimitrios Papakostas

December 2018

We the undersigned committee
hereby approve the attached dissertation

Backbones for
Military Multilayer Wireless Ad hoc Networks
based on Network Science Concepts

by
Dimitrios Papakostas

Dimitrios Katsaros
Assistant Professor
Department of Electrical and Computer Engineering, University of Thessaly
Committee Chair

Christos Antonopoulos
Assistant Professor
Department of Electrical and Computer Engineering, University of Thessaly
Committee Member

Lefteris Tsoukalas
Professor
Department of Electrical and Computer Engineering, University of Thessaly
Committee Member

Athena Vakali
Professor
Department of Electrical and Computer Engineering, University of Thessaly
Outside Committee Member

Spyros Lalis
Associate Professor
Department of Electrical and Computer Engineering, University of Thessaly
Outside Committee Member

Dimitrios Bargiotas
Associate Professor
Department of Electrical and Computer Engineering, University of Thessaly
Outside Committee Member

Apostolos Papadopoulos
Associate Professor
Department of Informatics, Aristotle University of Thessaloniki
Outside Committee Member

Dedication

to my parents and
my loving wife Nektaria ...

Synopsis and Contributions

In the context of the present dissertation we focus on Military multilayer wireless networks and invest on the ad hoc network paradigm to improve their performance. We envision these Tactical wireless networks as having layers of Mobile Ad hoc Networks (MANETS) which will be all connected together as part of a seamless overarching network, allowing information, from multiple sources, to flow faster among its participants.

Towards the implementation of this vision first we build on the (minimum) Connected node Dominating Sets theory and propose novel distributed algorithms to efficiently connect multilayer ad hoc networks so as to reduce their latency, improve their scalability and increase their lifetime.

Then, we investigate the problem of distributed computation of a resilient network overlay for communication link monitoring in multilayer ad hoc networks. A hard to address problem, especially using distributed algorithms, because of coordination failures related to the multilayer environment which can lead to loss of communication between the network layers or over-dependence on a specific layer. In order to solve this problem we build on the (minimum) Edge Dominating Sets theory and propose novel distributed algorithms which calculate, for purposes of either network management/monitoring, an overlay with high numbers of inter-layer links, so as to the communication among different layers not to break easily (accidentally or due to malicious attacks).

The main contributions of the present dissertation are:

- It is introduced the problem of calculating (minimum) connected node dominating sets (MCDS) in multilayer networks which has not been considered in the literature so far. In this context, we prove that decomposition-based and aggregation-based approaches for dominating set calculation in multilayer networks will not work and highlight the significance of assessing and exploiting each node's intra- and inter-layer links in order to be considered as candidate members of DS. Thus, we propose a family of centrality measures able to rank a multilayer node, with respect to its "strategic" position in the multilayer network. We also develop distributed algorithms for calculating the connected dominating set which exhibit superior performance compared to other widely used algorithms.

- It is introduced the novel problem of finding (minimum) connected edge dominating sets (MCEDS) in multilayer networks. We prove that the problem of finding the MCEDS in a multilayer network is NP-hard and we propose a centrality measure-based technique that provides to multilayer network nodes awareness about the significance of the edges

that are adjacent to them. Moreover, we develop distributed algorithms that heuristically calculate the MCEDS.

- It is proposed a new trajectory prediction scheme for use in the Vehicle Ad hoc Network environment, which builds in a purely distributed fashion a rich summary of a vehicles' roaming history that subsequently is used to provide online accurate predictions. We compare the proposed method against several, model independent and highly accurate prediction algorithms and the results affirm its superior performance.

Σύνοψις και Συνεισφορές

Στο πλαίσιο της παρούσας Διδακτορικής διατριβής εστιάζουμε στα Στρατιωτικά πολυεπίπεδα ασύρματα δίκτυα και επενδύουμε στην τεχνολογία των ***ad hoc*** δικτύων προκειμένου να βελτιώσουμε την απόδοση τους. Οραματιζόμαστε τα Τακτικά ασύρματα δίκτυα να απαρτίζονται από στρώματα ***ad hoc*** δικτύων με κινούμενους κόμβους, τα οποία θα είναι αδιαφανώς συνδεδεμένα μεταξύ τους ως μέρος ενός απρόσκοπτου υπερδικτύου το οποίο θα επιτρέπει την άμεση ροή πληροφοριών μεταξύ των κόμβων του.

Για την υλοποίηση αυτού του οράματος αρχικά βασιζόμαστε στη θεωρία των ***Connected node Dominating Sets (CDS)*** και προτείνουμε καινοτόμους αλγόριθμους οι οποίοι επιτυγχάνουν κατανεμημένα αποτελεσματική σύνδεση των ανωτέρω πολυεπίπεδων ***ad hoc*** δικτύων ώστε να μειωθεί η καθυστέρηση τους ***(latency)*** , να βελτιωθεί η επεκτασιμότητά τους ***(scalability)*** και να αυξηθεί η διάρκεια της ζωής τους. Στη συνέχεια εξετάζουμε το πρόβλημα της αποτελεσματικής και ταυτόχρονα κατανεμημένης παρακολουθησης της διακίνησης των δεδομένων εντός του πολυεπίπεδου ***ad hoc*** δικτύου. Ένα δύσκολο πρόβλημα, ιδιαίτερα αν αυτό πρόκειται να επιλυθεί μέσω της χρήσης κατανεμημένων αλγορίθμων, λόγω των προβλημάτων συντονισμού που έχουν να κάνουν με το πολυεπίπεδο περιβάλλον καθώς δύνανται να οδηγήσουν είτε σε απώλεια επικοινωνίας μεταξύ των επιπέδων του δικτύου είτε σε υπερβολική εξάρτηση αυτού από ένα συγκεκριμένο επίπεδό του. Για την επίλυση του συγκεκριμένου προβλήματος βασιστήκαμε στη θεωρία των ***Connected edge Dominating Sets (CEDS)*** και προτείνουμε νέους κατανεμημένους αλγορίθμους οι οποίοι υπολογίζουν για σκοπούς διαχείρισης ή/και παρακολούθησης του πολυεπίπεδου δικτύου μία επικάλυψη αυτού η οποία εμπεριέχει μεγάλο αριθμό από διασυνδέσεις μεταξύ των επιπέδων του δικτύου έτσι ώστε η επικοινωνία μεταξύ αυτών να μη διακόπτεται εύκολα (είτε κατά λάθος είτε λόγω κακόβουλων επιθέσεων).

Οι κύριες συνεισφορές της παρούσας Διδακτορικής διατριβής είναι:

- Παρουσιάζεται το πρόβλημα του υπολογισμού του ***(minimum) Connected node Dominating Set (CDS)*** σε πολυεπίπεδα δίκτυα το οποίο δεν έχει εξεταστεί μέχρι στιγμής στη βιβλιογραφία. Στο πλαίσιο αυτό, αποδεικνύουμε ότι, οι προσεγγίσεις είτε για επίλυση του συγκεκριμένου προβλήματος με βάση την αποσύνθεση του πολυεπίπεδου δικτύου στα επιμέρους επιπέδα του είτε για συσσωμάτωση των επιπέδων του ώστε αυτά επί της ουσίας να μην υφίστανται δε θα φέρουν το επιθυμητό αποτέλεσμα και τονίζουμε την ανάγκη για αξιολόγηση αλλά και περαιτέρω εκμετάλλευση των διασυνδέσεων που διαθέτουν οι κόμβοι του δικτύου ώστε αυτοί να θεωρηθούν ως υποψήφια μέλη του ***DS***. Για το λόγο αυτό, προτείνουμε μία ομάδα από μέτρα για τη μέτρηση της σημαντικότητας του κάθε κόμβου εντός του πολυεπίπεδου δικτύου ώστε να προσδιοριστεί η στρατηγική θέση καθενός από αυτούς. Επίσης, αναπτύσουμε αλγορίθμους

οι οποίοι υπολογίζουν κατανεμημένα το επιζητούμενο ***DS*** και παρουσιάζουν ανώτερη απόδοση σε σύγκριση με άλλους ευρέως χρησιμοποιούμενους αλγόριθμους.

- Παρουσιάζεται επίσης, το πρόβλημα της εύρεσης ***(minimum) Connected edge Dominating Set (CEDS)*** σε πολυεπίπεδα δίκτυα. Αποδεικνύουμε ότι, το συγκεκριμένο πρόβλημα είναι ***NP-hard*** και προτείνουμε μία τεχνική η οποία προσδίδει στους κόμβους του πολυεπίπεδου δικτύου αντίληψη αναφορικά με τη σπουδαιότητα των ακμών που ακουμπούν σε πάνω σε αυτούς. Επιπρόσθετα, αναπτύσουμε νέους ευριστικούς αλγόριθμους οι υπολογίζουν κατανεμημένα το ***CEDS***

- Προτείνεται ένας καινοτόμος αλγόριθμος πρόβλεψης για χρήση στο περιβάλλον των ***Vehicle Ad hoc Networks (VANETs)*** , ο οποίος με καθαρά κατανεμημένο τρόπο και λαμβάνοντας υπόψη του την ιστορία κίνησης ενός οχήματος πραγματοποιεί ***online*** ακριβείς πρόβλεψεις αναφορικά με τη μελλοντική τροχιά του υπό εξέταση οχήματος. Συγκρίνουμε την προτεινόμενη μέθοδο με άλλους υψηλής ακρίβειας αλγόριθμους πρόβλεψης και τα αποτελέσματα επιβεβαιώνουν την ανωτερότητα του αλγορίθμου μας.

PUBLICATIONS

Articles in Journals

[J1] Dimitrios Papakostas, Soheil Eshghi, Dimitrios Katsaros, Leandros Tassiulas. " *Distributed Algorithms for Multilayer Connected Edge Dominating Sets*", **IEEE Control Systems Letters**, vol.3, pp.31-36, January, 2019.

[J2] Dimitrios Papakostas, Soheil Eshghi, Dimitrios Katsaros, Leandros Tassiulas. " *Energy-Aware Backbone Formation in Military Multilayer Ad Hoc Networks*", **Ad Hoc Networks (Elsevier)**, vol.81, pp.17-44, December, 2018.

[J3] Dimitrios Papakostas, Dimitrios Katsaros. " *A Simulation-based Performance Evaluation of a Randomized MIS-based Clustering Algorithm for Ad Hoc Networks*", **Simulation Modelling: Practice And Theory (Elsevier)**, vol.48, pp.1-23, November, 2014.

Articles in Conference Proceedings

[C1] Dimitrios Papakostas, Soheil Eshghi, Dimitrios Katsaros, Leandros Tassiulas. " *Distributed Algorithms for Multilayer Connected Edge Dominating Sets*", **Proceedings of the 57th IEEE Conference on Decision and Control (CDC)**, Miami Beach, FL, USA, December 17-19, 2018.

[C2] Dimitrios Papakostas, Dimitrios Katsaros. " *A Rich Dictionary Markov Predictor for Vehicular Trajectory Forecasting*", **Proceedings of the 30th International Conference on Tools with Artificial Intelligence (ICTAI)**, IEEE Press, Volos, Greece, November 5-7, 2018.

[C3] Dimitrios Papakostas, Pavlos Basaras, Dimitrios Katsaros, Leandros Tassiulas. " *Backbone Formation in Military Multilayer Ad Hoc Networks Using Complex Network Concepts*", **Proceedings of the 35th IEEE Military Communications Conference (MILCOM)**, Baltimore, Maryland, USA, November 1-3, 2016.

Submitted to Conference

[S1] Dimitrios Papakostas, Soheil Eshghi, Dimitrios Katsaros, Leandros Tassiulas. " *Energy-aware distributed edge domination of multilayer networks*", **Paper submitted for publication**.

ACKNOWLEDGEMENTS

Firstly, I would like to express my deep gratitude to Assistant Prof. Dimitrios Katsaros, my Doctoral supervisor, for his continuous support of my Ph.D study and related research, for his guidance, enthusiastic encouragement and immense knowledge. I acknowledge that his intuition regarding military operations objectives and needs and how all these constrains are interpreted in the military environment exceed common sense. All in all, his contribution in the present dissertation was invaluable.

I am particularly indebted to Prof. Leandros Tassiulas for giving me the opportunity to participate in various projects and broaden my knowledge in many aspects. Moreover, I would like to thank my co-authors; Pavlos Basaras and Soheil Eshghi for their precious contribution. I would also like to express my thanks to Leandros Maglaras for supporting me at the initial steps of my PhD. I am very gratefull to Assistant Prof. Antonopoulos Christos and Professor Tsoukalas Lefteris for their valuable comments and support for my dissertation. To Professor Athena Vakali, Associate Prof. Spyros Lalis, Associate Prof. Dimitrios Bargiotas and Associate Prof. Apostolos Papadopoulos I want to express my sincere thanks for accepting to serve in the examination committee of my dissertation.

I am grateful to my parents, Spyros and Freideriki for their wholehearted love and support over all these years.

As always, many thanks to Nektaria, my wife, who supports all my endeavours.

Finally, I would like to express my sincere apologies to my children Faye, Eleftheria, George and Myrto for spending so much time in my research during the last years and not with them.

AUTHOR'S DECLARATION

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: ..... DATE: .....

TABLE OF CONTENTS

List of Tables

List of Figures

ACRONYMS

μ-PCI	μ-Power Community Index
ALZ	Active LeZi
CDS	Connected node Dominating Set
CEDS	Connected Edge Dominating Set
clPCI	Cross-layer PCI
CR	Cognitive Radio
DFS	Dynamic Frequency Selection
DS	Dominating Set
DSRC	Dedicated Short-Range Communications
DTN	Delay Tolerant Networking
E2CLB	Energy-Efficient Cross Layer Backbone
E2MLB	Energy-Efficient MultiLayer Backbone
E2WDB	Energy-Efficient Weighted Degree Backbone
EclPCI	Energy aware clPCI
EDS	Edge Dominating Set
EUCLB	Energy-Unaware Cross Layer Backbone
FANETs	Flying Ad hoc NETworks
GPS	Global Positioning System
IEDS	Independent Edge Dominating Set
IoT	Internet of Things
IP	Internet Protocol
IS	Independent Set
ITS	Intelligent Transportation System
JAN-TE	Joint Airborne Network - Tactical Edge
JTRS	Joint Tactical Radio System
KTRD	Kendall Tau Rank Distance
laPCI	Layer-agnostic PCI
LZU	LeZi Update

MANETs	Mobile Ad hoc NETworks
MCDS	Minimum Connected node Dominating Set
MCEDS	Minimum Connected Edge Dominating Set
MCMCEDS	Multi-Colored MCEDS
MEDS	Minimum Edge Dominating Set
MEMCDS	Maximum Energy MCDS
MEMS	Micro Electro-Mechanical Systems
MIMO	Multiple-Input and Multiple-Output
MIS	Maximal Independent Set
ML-MEMCDS	Multilayer Maximum Energy MCDS
mlPCI	Minimal-layers PCI
MPPM	Modified Prediction by Partial Match
NCD	Network Centric Defence
NCW	Network Centric Warfare
NEC	Network Enabled Capability
PCI	Power Community Index
QoS	Quality of Service
RDM	Rich Dictionary Markov
SDN	Software-Defined Networking
SDR	Software Defined Radio
SIR	Susceptible-Infectious-Recovered
SIS	Susceptible-Infectious-Susceptible
SOP	Sensory Organ Precursor
SRW	Soldier Radio Waveform
TDM	Time Division Multiplexing
UAV	Unmanned Aerial Vehicle
UDG	Unit Disk Graph
VANETs	Vehicle Ad hoc NETworks
VBN	Virtual Backbone
WNW	Wideband Networking Waveform

CHAPTER 1

THE EVOLVING CONDUCT OF WAR

1.1 Introduction

While the concept of war remains constant in history; the conduct of war is constantly experiencing change in response to new concepts, technologies, and capabilities. How modern armies adapt to such changes determines their alertness to confront future operational challenges and threats. Applied immediately, technological innovations can provide battlefield advantage, particularly when they facilitate or complement new ways to conduct war.

According to General (ret) Eric K. Shinseki, the head of the US Armed Forces, perhaps the world's most experienced and combat ready army, all future military operations will be carried out by medium-sized forces, but with the flexibility of light forces and same efficacy and survival capacity as that of heavy forces. The implementation of this vision will be made possible through the use of an integrated information management system (see Figure 1.1), which will provide at all levels of governance a catalytic leadership in decision-making so that all kinds of operations be successfully executed. The information revolution [121], with the promise of accelerating breakthroughs for surveilling, understanding, and communicating is expected to create a base of knowledge for military planning and execution unprecedented in scope, volume, accuracy, and timeliness.

Network Centric Warfare (NCW), Network Centric Defence (NCD) and Network Enabled Capability (NEC) [131] describe current military initiatives to maximise the benefit of advanced technologies in information sharing. By connecting multiple communicating platforms to present information to users on a need to know basis, military capabilities can be increased dramatically. Fundamental to this is that everything in a military NEC environment will be connected together as part of a seamless network, allowing information from more sources to flow faster, be analysed

FIGURE 1.1. A schematic representation of an integrated information management system (taken from `https://www.thalesgroup.com/en/worldwide/news/smart-connectivity-airborne-operations`).

more effectively and be acted upon sooner. Fully realised, NEC offers the following benefits to military operations:

- Enhanced combat identification
- Assured communications interoperability
- Improved shared situational awareness
- Enhanced multi-national command, control and coordination
- Reduced risk of fratricide and collateral death and damage
- Improved mutual support
- Improved combat effectiveness

Towards the implementation of the NEC the ad hoc network technology plays a crucial role as this networking paradigm provides the necessary mobility [Mobile Ad hoc NETworks (MANETs), Vehicle Ad hoc NETworks (VANETs), Flying Ad hoc NETworks (FANETs)] required by the lower echelons of a military force. Likely, military wireless multilayer networks will be a heterogeneous network of networks, an amalgam of differing networks including Internet Protocol (IP), Time Division Multiplexing (TDM) and other legacy or emerging technologies. We acknowledge that there is a strong need for creating seamless links between the communicating platforms which will ultimately permit cross layer connectivity in the field and at the command level. Loss of seamless communications could jeopartize overall situational awareness and thus preclude fulfilment of the operational objectives. Moreover, we need the communicating platforms to be connected in an efficient way so as to e.g., reduce the number of redundant messages in the multilayer network, improve network scalability, reduce network latency, etc.

1.2 Motivation

The present dissertation focuses on the tactical wireless ad hoc networks which we envision working as a seamless multilayer ad hoc network. Specifically, it poses and answers the following main questions and challenges:

- How can we efficiently connect a multilayer ad hoc network ?
- How can we efficiently monitor the performance of that multilayer ad hoc network ?
- Is it possible to accomplish the aforementioned objectives based solely on local knowledge of the network topology and thus effectively deal with rapidly changing networks or incomplete knowledge of a network's connections ?

In the context of the present dissertation we investigate ways to constructing in a distributed fashion structures for improving connectivity and monitoring performance of a military multilayer network with the intent to support the NEC. Even though traditional graph-theoretic concepts [39, 143] can be used for these problems, network science concepts such as centralities can also provide essential tools for the management of military ad hoc networks [55, 79, 80], and especially in our case, they can help identify efficient cross-layer dominators.

1.3 Contribution

The present dissertation makes the following contributions:

In **Chapter 2** we set the conceptual framework of a military *multilayer* tactical network; i.e, having tiers of subnets built up with waveforms (a wireless multiple access radio frequency technology) and present some emerging technologies that can be used to homogenize these heterogenous subnets and thus enable cross layer connectivity. Then we argue on the importance of topology control in a multilayer network, we provide some graph theoretic approaches for the implementation of this task; the construction of a backbone of nodes of the multilayer network and discuss which approach fits better in the military environment. Finally, we present some well known centrality measures that are widely used to quantify the importance of a network node and thus be used in the backbone.

In **Chapter 3** we introduce the problem of calculating Minimum Connected node Dominating Set (MCDS) in multilayer networks which has not been considered in the literature so far. In this context, we prove that decomposition-based and aggregation-based approaches for Dominating Set (DS) calculation in multilayer networks will not work and highlight the significance of assessing and exploiting each node's intra- and inter-layer links in order to be considered as candidate members of DS. Finding the MCDS in the centralized setting (i.e., having knowledge

of the complete network topology) belongs to the class of NP-complete problems; it is understood that efficient (heuristic) distributed solutions to the same problem – which are those preferred for ad hoc networks – are much harder to devise.

In order to solve the problem we propose a family of centrality measures able to rank a multilayer node, with respect to its "strategic" position in the multilayer network. We also develop a distributed algorithm for calculating the connected dominating set and experimentally evaluate the proposed methods for constructing connected dominating set against a robust competitor.

In **Chapter 4** we investigate the issue of calculating energy-aware MCDS based backbones for multilayer networks. It generalizes the problem defined in the previous chapter. We highlight that assigning different weights on intra-layer versus inter-layer links can not help transform the problem into that of dealing with the calculation of a weighted dominating set of the multilayer network, because there is no algorithmic method yet for the determination of the relative weights so as to produce an efficient backbone for multilayer networks.

In order to solve the problem we propose a centrality measure for identifying nodes with high residual energy and central position within the multilayer network. Based on that centrality measure we develop an energy aware backbone construction algorithm; we analyze its performance both from a computational/communication complexity perspective and an experimentation-based perspective and finally we exhaustively compare it against relevant and baseline competitors, because there is no prior work on multilayer networks.

In **Chapter 5** we introduce the novel problem of finding a (minimum) connected edge dominating set (MCEDS) in multilayer networks. The problem of distributed computation of a resilient network overlay for communication link monitoring in single layer ad hoc networks has been studied in the past. However, in the context of multilayer ad hoc networks, this problem is significantly harder to address, especially using distributed algorithms, because of coordination failures which can lead to loss of communication or over-dependence on a specific layer. In this context, we formulated the problem with the additional constrain of including many inter-layer links into the EDS. Designing networks with high numbers of inter-layer links immunizes the network to (possibly correlated) failures in any particular layer, allowing the design of resilient network overlays for purposes of either network management/monitoring or data forwarding, in the sense that the communication among different layers can not break easily (accidentally or due to malicious attacks).

We prove that the problem of finding the MCEDS in a multilayer network is NP-hard. In order to solve it we propose a centrality measure-based technique that provides to networks nodes awareness about the significance of the edges that are adjacent to them. Then we develop three distributed algorithms that heuristically calculate the MCEDS and we evaluate their performance.

In **Chapter 6** we investigate the problem of constructing energy aware MCEDS for multilayer networks. Communication and monitoring are both energy-intensive activities. The resulting energy depletion may exhaust the batteries of network elements, disconnecting the overlay and/or resulting in the loss of monitoring capabilities. Thus, the distribution of energy among elements chosen in the overlay must be such that there are fewer low energy elements.

In that context, we propose an energy-aware centrality measure so as to provide to network nodes awareness about the energy status and the connectiveness of the edges that are adjacent to them. Then we develop three distributed algorithms to solve the problem and we evaluate their performance.

In **Chapter 7** we further investigate how to improve the performance of a military network. To elaborate, we focus on the vehicular environment, a fundamental ingredient of the modern battlefield, and we invest on the VANETs technology with the intent to improving robustness regarding the overarching multilayer network. In the context of a military multilayer network mobility is considered an inherent attribute of the participating hosts which makes backbone construction a difficult if not impossible task. In such demanding environment trajectory forecasting of mobile hosts is a viable option to improving routing protocols, traffic management, connectivity robustness, etc.

VANETs, are spontaneous, flexible wireless networks that are able to support the associated applications in dynamic, multihop topologies. However, the relatively high speed of the moving vehicles degrades link quality, causes fast fading, short connectivity and high frequency hand-offs. In order to improve the performance of a $VANET$ system we propose a new trajectory prediction scheme which builds in a purely distributed fashion a rich summary of a vehicle's roaming history that subsequently is used to provide accurate predictions. We compare the proposed method against several, model independent and highly accurate prediction algorithms and the results affirm its superior performance.

In **Chapter 8** we investigate the problem of scaling down the ad hoc networks. Providing a hierarchical organization to ad hoc networks has been identified as a viable and efficient option for this problem in the literature. However, despite the fact that a really large number of algorithms have been proposed during the last decade for dealing with this problem the proposal and study of protocols which exhibit low message complexity, fast convergence, incremental backbone maintenance, resilience to hub node failures, connectivity preservation, and backbone stability is still in quest.

In that context we select some baseline algorithms from different graph-theoretic classes, connected dominating set, independent sets, which exhibit a diversification in the degree of localization in this problem solving and evaluate their performance. We develop a rich comparison

framework for the ad hoc network clustering protocols by employing three families of performance measures, namely for protocol cost, for backbone description and for robustness. Further, we propose an improvement to an existing algorithm appropriate for cluster rebuilding processes in case of node failure.

Finally, in **Chapter 9** we conclude the work conducted in the context of this dissertation; i.e., integrate and synthesize the various issues raised in the preceding chapters, whilst reflecting the introductory dissertation statements and objectives. Moreover, we provide direction and areas for future research.

CHAPTER 2

INTRODUCTION TO TECHNOLOGY AND SCIENCE

2.1 Developments in Military Communications Systems

Tactical wireless networks have tiers of subnets [109] (islands of MANETs as in Figure 2.1). These subnets are constructed with waveforms (a waveform is a wireless multiple access radio frequency technology). Contrary to civilian wireless networks, tactical wireless networks nodes can be all mobile; there is no fixed infrastructure, and the end-users are part of the infrastructure.

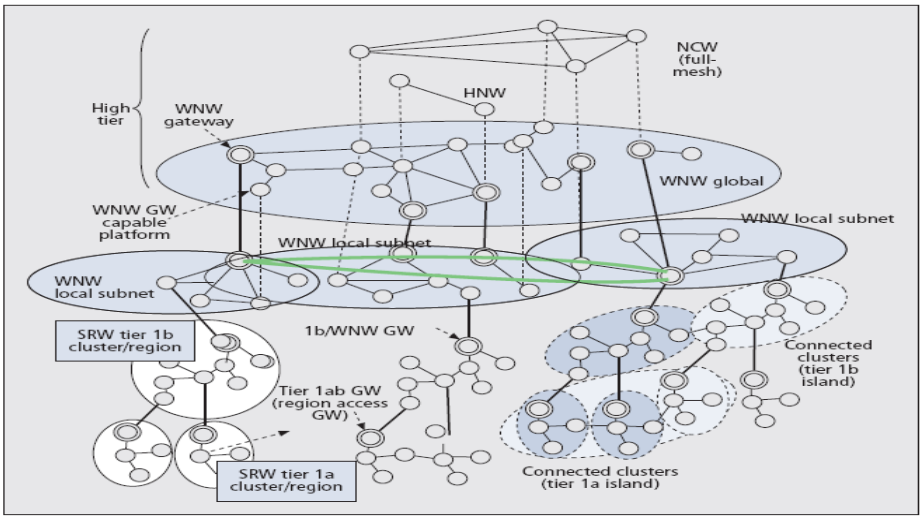

FIGURE 2.1. A detailed example of a military multilayer network structure (taken from [109]).

The task of interconnecting the subnets so as to create a homogeneous *multilayer* tactical network is one the most challenging and difficult problems to solve towards the implementation of the NEC. Thankfully, advances in technology are helping to break down the barriers to telecommunications interoperability that have long stymied military planners. Software Defined Radio (SDR) [117] technology is likely to revolutionize the integration of multiple waveforms and enable interoperability with legacy systems and communications with entities utilizing different standards. Further, SDR will enable multiband and cross-band operation. In the context of a military multilayer network an SDR-capable user could participate in more than one networks, permitting information exchange between heterogeneous networks. The ability for a communication device to operate in multiple bands is critical to supporting extant waveforms and providing seamless connectivity.

Maybe the most favorable attribute of military SDRs, which differentiates them from previous technologies, is the ability to change their functionality online by downloading another waveform application [97]. New waveforms may (on a fundamental level) incorporate the modulation scheme, communication protocols, cryptographic algorithms and possibly some network level applications including cross-banding and gateways. This expands their functionality and enables new communication capabilities to be present right away without, in many cases, making any changes to the existing communication hardware.

FIGURE 2.2. Improved connectivity through the use of SDR technology (taken from `https://asc.army.mil/web/news-alt-jfm17-reuse-refine-resolve`).

Further, the increase in the number of systems utilizing the wireless medium within a military multilayer network makes the problem of effective frequency management of significant importance. In this context, given the restricted capacity of the wireless networks [52], the need for detailed frequency management involving a large number of systems is envisaged. In addition, it would be profoundly favorable for the network to be able to perform frequency management

autonomously. Whilst this represents a major challenge, the solution of this particular problem, using Cognitive Radio (CR) techniques [4], would provide incredible advantage in the effective management of combat radio systems. CR is a new communication paradigm that senses, and is aware of, its operational environment and can dynamically and autonomously adjust its radio operating parameters accordingly. This approach includes wireless links from Unmanned Aerial Vehicle (UAV), satellites, and wired connections that can utilize a variety of anti-interference and stealth techniques to compensate for hostile environments. This is a promising ongoing area of research towards the implementation of autonomous, adaptive radio communications networks.

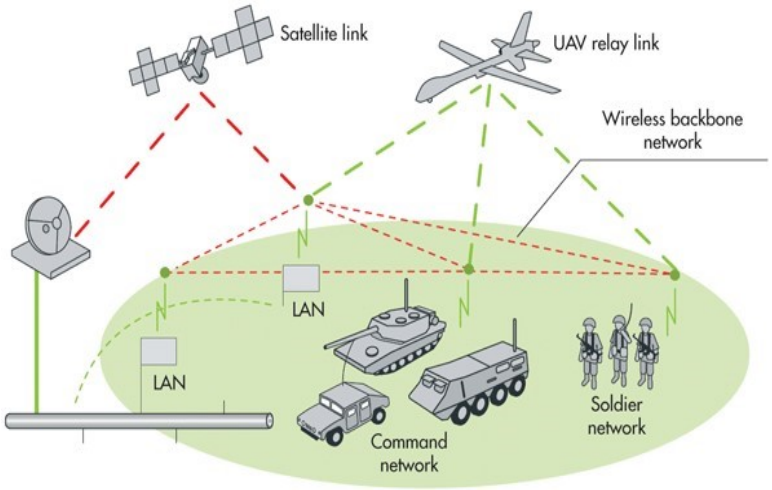

FIGURE 2.3. A cognitive-radio system could help gather a variety of different wireless links into a united and intelligent tactical wireless network (taken from `https://www.mwrf.com/systems/bringing-cognition-tactical-networks`).

Finally, establishing secure sharing of information and associated trust relationships across the tactical wireless network is also a major issue. In near future, existing legacy systems that make extensive use of cryptographic devices operating at the link layer will be upgraded so as to be able to work at the network layer, and thus support secure operation across IP-based networks [138]. Programmable architectures that will be able to support flexible operation using multiple cryptographic algorithms and interface types is also envisioned. All in all, the existing and the emerging advances in communication technology have set the ground for seamless information sharing among the actors of a multilayer tactical wireless network.

2.2 Topology Control in Tactical Networks

Once a multilayer network is deployed the communication links among its nodes comprise the topology of the network. Topology control is of particular interest in a multilayer network as it can extend its lifetime while simultaneously preserve other important characteristics such as

its connectivity and coverage. Topology control is the reorganization and management of node parameters and modes of operation from time to time to modify the topology of the network [29].

2.2.1 Virtual backbone construction

In the framework of the ad hoc networks a common practice for topology control is the construction of hierarchical topologies as it is beneficial to improving network scalability and efficiency. In order to build such topologies most broadcasting algorithms use the concept of the Virtual Backbone (VBN). A VBN is a subset of nodes that works as a communication layer, and only the nodes in the communication layer transmit data, significantly reducing the transmission of redundant information, simplifying the topology of the network, saving the energy for information gathering and filtering, routing and forwarding information required. Among the most-researched methods for VBN formation in wireless ad hoc networks are those based on clustering [142, 145] and those based on dominating sets [54, 104, 143].

2.2.2 Cluster-based hierarchical topologies

Under a cluster structure, network nodes are partitioned into a number of small groups called clusters. In each cluster nodes may be assigned a different status or role, such as cluster head (CH), cluster gateway (CG), or cluster member (CM) (see Figure 2.4).

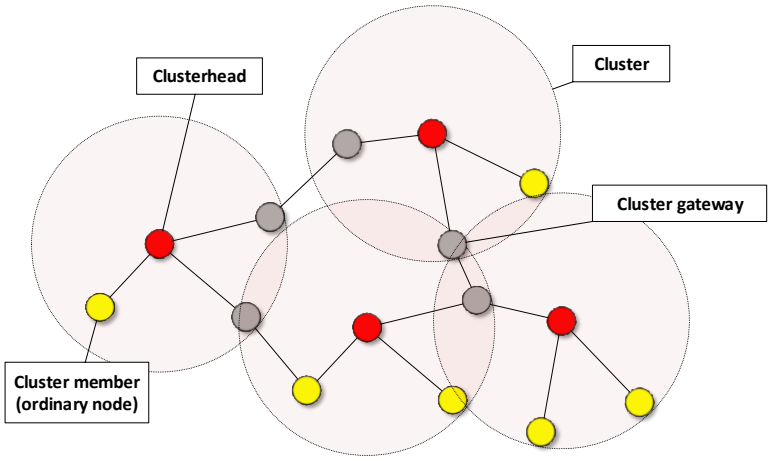

FIGURE 2.4. Cluster structure illustration.

A CH serves as a local coordinator for its cluster, performing intra-cluster transmission management and data forwarding. A CG is a node with inter-cluster links, so it can access neighboring clusters and forward information between clusters. In cases where overlapping clusters exist a CH may have dual duties and forward information to adjacent CHs. A CM is a non-CH node without any inter-cluster links. A clustered network structure results in a two-tier

hierarchy in which CHs form the higher tier while CMs form the lower tier. The benefits of a cluster-based approach to building a hierarchical topology in a multilayer network are:

- Facilitates the spatial reuse of resources to increase the system capacity.
- Improves network scalability and efficiency through the construction of a virtual backbone between the set of CHs and CGs.
- Improves network robustness as information processed and stored by each mobile node is greatly reduced as it concerns only the cluster it belongs.

Clustering reduces channel contention and packet collisions, resulting in better network throughput under high load. However, a cluster-based MANET has its side effects and drawbacks. To elaborate:

- Cluster formation in a dynamically changing scenario is an open issue as most clustering algorithms assume some degree of node stationarity when cluster formation is in progress.
- To maintain a cluster structure in a dynamically changing scenario often requires frequent information exchange between mobile node pairs which may consume considerable bandwidth and drain mobile nodes' energy quickly.
- Some clustering schemes may cause the cluster structure to be completely rebuilt over the whole network when some local events take place.

All in all, clustering can be problematic for modern battlefields because the participating units are highly mobile and thus frequent re-clustering might be necessary; this will jeopardize the communication ability of the entities and also put under question the robustness of the multilayer network. Even the approaches for clustering VANETs [83, 103] that take mobility into account will not work in a battlefield, because these approaches exploit the road network topology which is usually not present in a battlefield.

2.2.3 DS-based hierarchical topologies

The drawbacks of the cluster-based topology construction approach constitute clustering a precarious option in the dynamic environment of a multilayer military network. On the other hand, the option that is based on dominating sets for the construction of the VBN provides the required flexibility so as to be a viable approach. Before commenting on dominating sets we will first provide some basic definitions from graph theory.

A graph $G = (V, E)$ is used to represent a wireless network, where V represents the set of all nodes in the network and E represents the set of all links in the network. For a node $u \in V$, $id(u)$ denotes the unique id for u and $d(u)$ denotes the degree of u. $N(u)$ is the open neighbor set of u, that is, $N(u) = \{v|(v,u) \in E\}$ and $d(u) = |N(u)|$. $N[u] = N(u) \bigcup \{u\}$ is the close neighbor set of u. For a subset S of V, $G[S]$ is a subgraph of G induced by S. $N(S) = \bigcup_{u \in S} N(u)$, and $N[S] = \bigcup_{u \in S} N[u]$.

Definition 2.1. *The maximum node degree of the network is* Δ*, where* $\Delta = max\{d(u)|u \in V\}$.

Definition 2.2. *A graph* G *is said to be strongly connected if for any pair of nodes* $u, v \in V$*, there exists a (directed) path between them. Likewise, a subset* S *of* V *is called a strongly connected set if* $G[S]$ *is strongly connected.*

Definition 2.3. *An Independent Set (IS) is a subset* S *of* V *such that there is no edge in* $G[S]$ *(see Figure 2.5 LEFT).*

Definition 2.4. *A Maximal Independent Set (MIS) is an IS such that adding any node not in the set breaks the independence property of the set (see Figure 2.5 RIGHT). Thus, any node not in the MIS must be adjacent to some node in the set.*

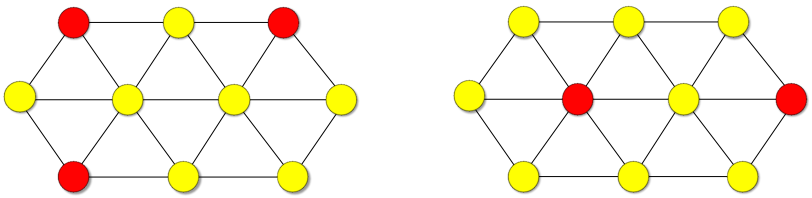

FIGURE 2.5. (LEFT) Nodes in red comprise an independent node dominating set. (RIGHT) Nodes in red comprise a maximal independent node dominating set.

Definition 2.5. *A DS of a network* (G,E) *is any subset of* G *with the property that any node* v *of* G *is either a member of the DS (then,* v *is called a dominator) or* v *is one hop away from a dominator (then,* v *is called a dominatee) (see Figure 2.6). Thus, every MIS is a DS. However, since nodes in a DS may be adjacent to each other, not every DS is an MIS. Finding a minimum DS belongs to the class of NP-complete problems [47].*

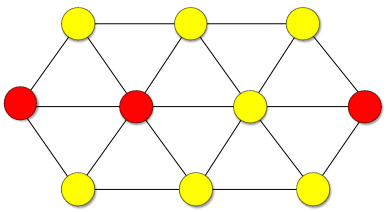

FIGURE 2.6. Nodes in red comprise a dominating set.

Definition 2.6. *A Connected node Dominating Set (CDS)* C *of a network* (G,E) *is a DS of* G*, where* $G[C]$ *is connected (see Figure 2.7 LEFT)).*

Definition 2.7. *A MCDS is a CDS with minimum cardinality (see Figure 2.7 RIGHT)). Finding the MCDS is also an NP-complete problem [47].*

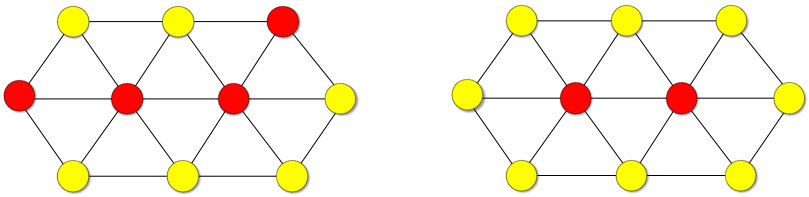

FIGURE 2.7. (LEFT) Nodes in red comprise a connected node dominating set. (RIGHT) Nodes in red comprise a minimum connected node dominating set.

We emphasize that there is no previous work on calculating connected dominating sets in multilayer networks. Nevertheless, the topic is relevant to a number of areas which we briefly discuss here. The wireless ad hoc networks community has extensively investigated the topic of distributed CDS algorithms for (single layer) ad hoc networks. Some of the proposed algorithms are centralized and yield a small CDS under the assumption that the complete network topology is known. However, such an assumption is unrealistic for the dynamic environment of a military multilayer network. The majority of the proposed CDS-related algorithms are decentralized; i.e., the decision process is distributed and requires only a constant number of communication rounds, making them an attractive option for constructing a VBN destined for the military multilayer environment. Nice surveys of recent results on this topic are presented in [104, 143]. The main focus of these works is to produce a small connected dominating set under various constraints, such as type (unidirectional/bidirectional) of links, mobility management, backbone diameter length, energy budget, different transmission ranges, interference, and so on.

Research on multilayer complex networks [16, 86] (generalization of multi-plex, or interdependent networks) is relatively new and spans directions such as formation mechanisms [111], centralities, communities [62], diffusion processes [106]. In this dissertation we are striving for an improved topology management of a military multilayer network by constructing robust $CDSs$ under the concept of influential spreaders. Influential spreaders in a complex network are those nodes which under a specific spreading model (e.g., Susceptible-Infectious-Recovered (SIR) [1], Susceptible-Infectious-Susceptible (SIS) [2]) are able to spread the "infection" in a large part of the network. After the seminal work of [71], measures such as *k-shell* [71], PCI [13] and others have been proposed to identify influential spreaders over single-layer complex networks. There is some work on positive influence dominating sets in single layer social networks [130]. Finally, our work [12] investigated the issue of detecting influential spreaders for multilayer complex networks using concepts similar to those presented in this dissertation.

[1] An SIR model is an epidemiological model that computes the theoretical number of people infected with a contagious illness in a closed population over time. The name of this class of models derives from the fact that they involve coupled equations relating the number of susceptible people S(t), number of people infected I(t), and number of people who have recovered R(t).

[2] SIS is very similar to SIR. SIS offers no immunization for the network nodes, i.e., the recovered state is excluded. SIS assumes that agents (the nodes) can only exist in the two remaining discrete states: susceptible or healthy (S) and infected (I).

Next, we will provide some basic information about centrality measures which describe the importance of a vertex in a networked system using some set of criteria, because they form the basis for defining influential nodes.

2.3 Centrality Measures

The importance or prominence of a node is synonymous with the strategic location of the node within the network. This prominence is described by numerous centrality measures [65]. The most noteworthy and substantively interesting centrality metrics are described in the sequel:

- *Degree centrality* is loosely defined as the number of one-hop neighbors of a node [132]. For a network with n nodes, the normalized degree centrality of a node a_i is:

$$DegC_{a_i} = \frac{degree(a_i)}{n-1}$$

  Intuitively, a node is prominent if it has many direct links to other nodes in the network; thus play a crucial role in the functioning of the network, e.g., robustness, when network nodes fail or malfunction, or for the efficient diffusion of information within the network. Degree centrality however can be deceiving as it is a local measure. To exemplify, in Figure 2.9 the red node has a degree = 3 but it is the most prominent node as it connects three different subgraphs.

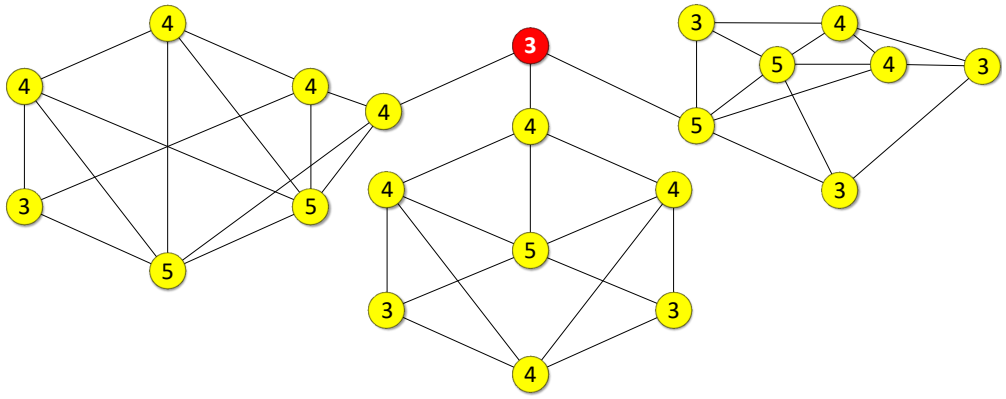

FIGURE 2.8. A degree centrality example. Degree centrality can be deceiving; in this particular example it fails to show how prominent is the red node.

- *Eigenvector centrality* quantifies the influence of a node in a network [132]. It is based on the spectral properties of the matrix that describes the relationships among the nodes. A high eigenvector value means that a node is connected to many nodes who themselves have high eigenvector values too (see Figure 2.9).

Let $A = (a_{i,j})$ be the adjacency matrix of a graph. The eigenvector centrality x_i of node i is:

$$x_i = \frac{1}{\lambda} \sum_k a_{k,i} x_k$$

where $\lambda \neq 0$ is a constant. In matrix form we have:

$$\lambda x = xA$$

This type of equation is well known and solved by the eigenvalues and eigenvectors of A. From the set of different eigenvectors only one seems to be an appropriate solution that can serve as a centrality measure. As A is the adjacency-matrix of an undirected (connected) graph, A is nonnegative and due to the theorem of Perron–Frobenius [95], there exists an eigenvector of the maximal eigenvalue with only nonnegative (positive) entries. Google's *PageRank* [91] and *Katz centrality* [67] are variants of the eigenvector centrality.

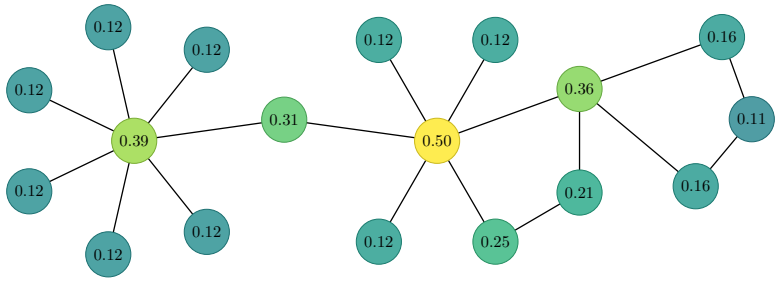

FIGURE 2.9. An eigenvector centrality example. Nodes are awarded high scores if they are connected to many nodes who themselves have high scores.

- *Betweenness centrality* measures the frequencies of nodes in the shortest paths between indirectly connected nodes and is formally defined as the fraction of the shortest paths between any pair of nodes that pass through a node [132] (see Figure 2.10).

FIGURE 2.10. A betweeness centrality example. Nodes towards the center of the graph are more prominent because the fraction of the shortest paths between any pair of nodes that passes through them is larger.

Mathematically, let $n^i_{s,t}$ be the number of geodesic paths from s to t that pass through i and let $n_{s,t}$ be the total number of geodesic paths from s to t. Recall that a geodesic path

is not necessarily unique and the geodesic paths between a pair of vertices need not be node-independent, meaning they may pass through some of the same vertices. Then the betweenness centrality of vertex i is:

$$b_i = \sum_{s,t} w^i_{s,t} = \sum_{s,t} \frac{n^i_{s,t}}{n_{s,t}}$$

It can be used, for instance, in message-carrying applications where we need to forward a packet to a node that is promising to deliver it with success and/or faster to its final destination.

- *Closeness centrality* quantifies the efficiency of information propagation from one node to all the others and gives an estimate of how long it will take information to spread from a given node to the rest of the network nodes [132]. Closeness centrality measures the mean distance from a node to other nodes. The distance can be measured in number of hops, delays, and so on. Recall that a geodesic path is a shortest path through a network between two nodes.

  Suppose $d_{i,j}$ is the length of a geodesic path from i to j, meaning the number of edges along the path. Then, for a network with n nodes, the normalized geodesic distance for node i is:

$$l_i = \left[\frac{\sum^n_{j \neq i} d_{i,j}}{n-1} \right]^{-1}$$

  This quantity takes large values for nodes that are separated from others by only a short geodesic distance on average (see Figure 2.11). Evidently, this measure could be used in applications where, for instance, we need to elect a single leader node to propagate alert messages.

FIGURE 2.11. A closeness centrality example. Nodes towards the center of the graph are more prominent because, on the average, their geodesic distance from the other nodes in the graph is shorter.

- *K-Core centrality* is a measure that can help identify tightly interlinked groups within a network. A k-core is a maximal group of entities, all of which are connected to at least k other entities in the group (see Figure 2.12). It has been widely used for the identification of influential spreaders in complex networks.

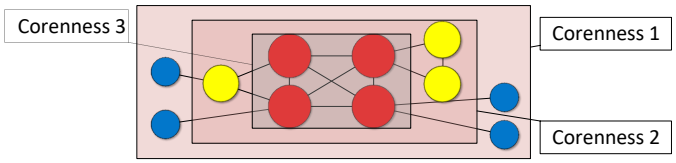

FIGURE 2.12. A K-core centrality example.

- *μ-Power Community Index (μ-PCI)* characterizes a node for the density in connections of both itself and its μ-hop neighbors [38]. The PCI index of a node v is equal to k, such that there are up to k nodes in the 1-hop neighborhood of v with degree greater than or equal to k, and the rest of the nodes in that 1-hop neighborhood have a degree less than or equal to k (see Figure 2.13). It has been a subject of intensive study in the present dissertation as it is an effective; low computation cost; and efficient method for detecting potent nodes that can communicate with a large subset of network nodes in large complex networked systems.

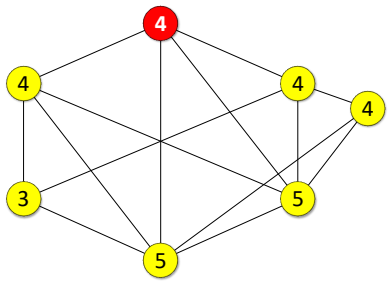

FIGURE 2.13. A PCI centrality example. The red node has a PCI = 4 as it is connected to 4 nodes and each one of them has a degree = 4.

Besides these representatives of the basic concepts of centrality, a plethora of other centralities has been defined over the years (see, e.g., [17, 46, 73, 102]) which, e.g., enable the integration of edge weights or of directional connections or are suitable for specific applications and network types. Usually, these centralities represent modifications or enhancements of the already discussed centralities and thus are not elaborated in more detail here. The interested reader is referred to [1, 35, 114] and references therein to have a more concrete view about centrality measures.

Centrality concepts have been exploited widely in ad hoc networking for purposes of cooperative caching [38], service deployments [80], access control [40], security [77], routing [76], in many areas of delay tolerant networking [81], and so on.

Multilayer networks [16] are a particularly hot research area of network science. In [12] several centrality measures were developed for helping in the identification of influential spread-

ers [71] in multilayer networks. Multilayer network literature has not been exploited widely yet in the area of ad hoc networking even though several of its advances can be applied there. Calculating connected dominating sets with the purpose of operating as backbones in wireless multilayer ad hoc networks was studied so far only in [92]; it was proved there that some peculiarities [61, 124] of the problem make existing solutions either not appropriate or not efficient.

CHAPTER 3

BACKBONE FORMATION IN MILITARY MULTILAYER AD HOC NETWORKS USING COMPLEX NETWORK CONCEPTS

3.1 Introduction

Tactical ad hoc networks encompass some unique characteristics that differentiate them in terms of requirements, expectations, needs and constrains from the respective commercial. Those characteristics are related to dynamic topology, scarcity of bandwidth and excessive delay. Tactical wireless networks built with the Joint Tactical Radio System (JTRS) [69] in mind have layers of subnets. There is the Soldier Radio Waveform (SRW) tier. It can have two subtiers, one for soldier-to-soldier communications and one for networking sensors. Above that, there is the Wideband Networking Waveform (WNW) tier, which has two subtiers; one forms local subnets for vehicle-to vehicle communications, and the other is for global connectivity, to generate a single subnet over the entire theater. There is also the Joint Airborne Network - Tactical Edge (JAN-TE) stub network that supports the tactical airborne domain of weapons platforms.

All in all, we consider these "island" subnetworks as being the layers of a single, large network, which we call a *multilayer* communication network. Abstracting all the specifics, we illustrate such a multilayer network in Figure 3.1 composed by a subnetwork (layer) of soldiers, a subnetwork (layer) of helicopters and a subnetwork (layer) of drones.

Fast, information spreading across the whole network is vital to many battle and intelligence operations. Since each layer is an ad hoc wireless network, the goal is to construct a VBN

Related publication **[C3]:** Dimitrios Papakostas, Pavlos Basaras, Dimitrios Katsaros, Leandros Tassiulas. "*Backbone Formation in Military Multilayer Ad Hoc Networks Using Complex Network Concepts*", **Proceedings of the 35th IEEE Military Communications Conference (MILCOM)**, Baltimore, Maryland, USA, November 1-3, 2016.

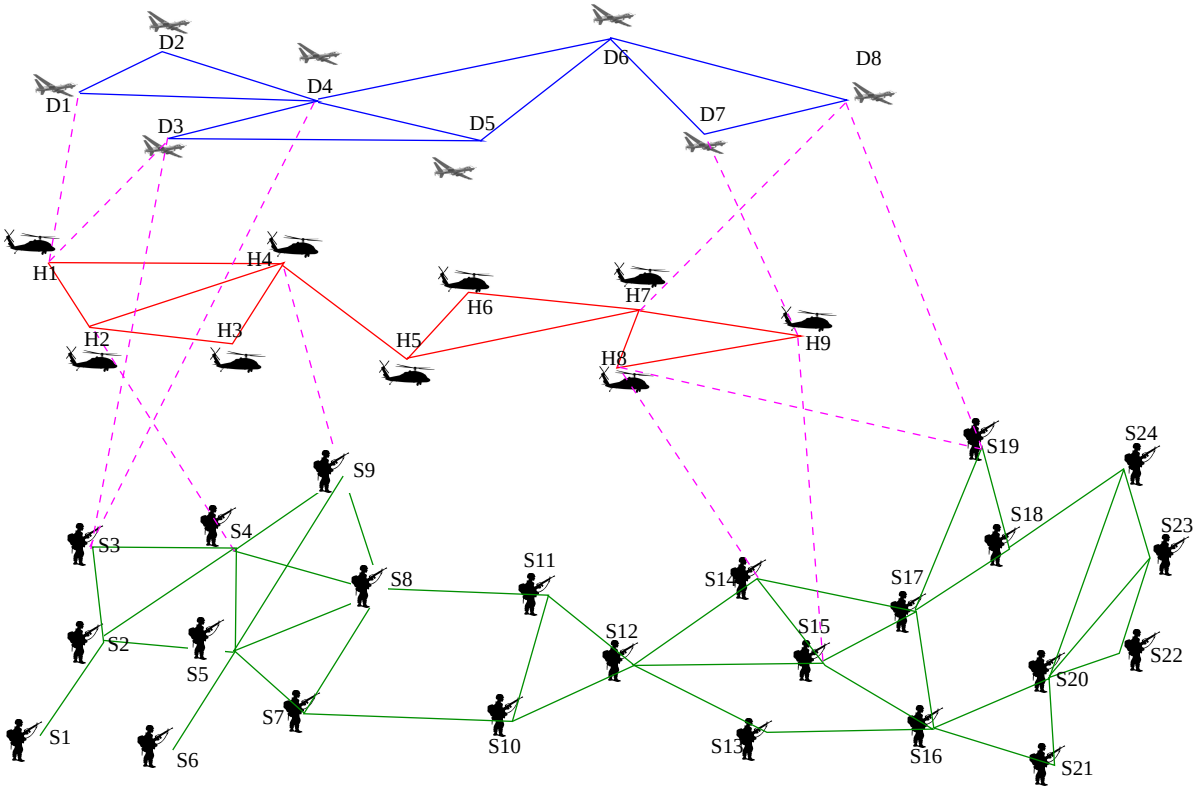

FIGURE 3.1. Abstraction of a multilayer ad hoc network (for the purposes of illustration, physical obstacles have been removed, and the entities have been projected into the two dimensional space): The first layer is comprised of soldiers, the second of helicopters, and the third of drones. Solid (green/red/blue) color links denote communication links among entities of the same layer (soldiers/helicopters/drones). Dashed, purple links denote links among entities belonging to different layers.

connecting all these layers in such a way that efficient and effective information dissemination can take place. According to § 2.2.3 backbones based on connected dominating sets [39] are a fine solution which combines flexibility (considerations for backbone diameter, mobility management, energy efficiency, different transmission ranges, interference, etc) and incorporation of even social-cognitive techniques [38].

Notably, the problem of constructing connected dominating set-based backbone for multilayer ad hoc networks has not been considered in the literature so far, mainly because there is no supporting technology. However, the recent advances in the Micro Electro-Mechanical Systems (MEMS) technology provide the framework in which some state of the art wireless communication technologies such as Multiple-Input and Multiple-Output (MIMO), SDR and CR make the homogenization of a multilayer ad hoc a reality. In this context, the present chapter makes the following contributions:

- It introduces the problem of calculating minimum connected dominating sets in multilayer networks.
- It defines measures of *node importance* which help to identify those nodes "strategically" positioned in the multilayer network that will act as dominators.
- It develops a distributed algorithm for calculating the connected dominating set.
- It experimentally evaluates the proposed method for constructing connected dominating set against a robust competitor.

The rest of this chapter is organized as follows: section 3.2 provides background information, it explains why earlier methods for dominating set calculation can not work, and formally defines the investigated problem; section 3.3 presents the proposed measures and the distributed algorithm for dominating set construction; section 3.4 evaluates the performance of the proposed algorithm, and section 3.5 concludes the chapter.

3.2 Backbone formation for multilayer ad hoc networks

Before we formulate the problem we will first make some comments on Figure 3.1. Using the Definitions from §2.2.3, it is easy to deduce that the MCDS for the drone-layer includes only the nodes $\{D4, D6\}$; the respective MCDS for the helicopter-layers includes the nodes $\{H4, H5, H7\}$, and a (there exist more than one) MCDS for the soldier-layer includes the nodes $\{S2, S5, S7, S10, S12, S15, S17, S16, S20\}$. Finding the MCDS of a graph in the centralized setting (i.e., having knowledge of the complete network topology) belongs to the class of NP-complete problems [47]; it is understood that efficient (heuristic) distributed solutions to the same problem – which are those preferred for ad hoc networks – are much harder to devise [143].

Definition 3.1. *A multilayer network comprised of n layers is a pair* (G^{ML}, E^{ML})*, where* $G^{ML} = \{G^i, i = 1, \ldots, n\}$ *is a set of networks* (G_i, E_i) *as defined earlier, and a set of inter-layer links* $E^{ML} = \{E_{i,j} \subseteq G_i \times G_j; i, j \in \{1, \ldots, n\}, i \neq j\}$.

In Figure 3.1, $G_1 = \{S_i, i = 1, \ldots, 24\}$, $G_2 = \{H_i, i = 1, \ldots, 9\}$, $G_3 = \{H_i, i = 1, \ldots, 8\}$, and E^{ML} is the set of all links denoted by dashed lines, e.g., (S_3, D_4).

3.2.1 Problem formulation

Suppose that we are given an undirected (links are bidirectional), unweighted (no weights on links/vertices) network comprised of multiple (i.e., more than one) layers denoted as (G^{ML}, E^{ML}). Then, this chapter studies the **ML-MCDS** problem from a distributed perspective and it also develops a heuristic approximation to the **ML-MCDS** problem.

Definition 3.2 (ML-MCDS problem)**.** *Solve the Minimum-Connected Dominating Set for a multilayer network in a distributed fashion, i.e., determine the set $MCDS^{ML}$ comprised of the minimum number of nodes (belonging to any layer) such as:*

(a) Their induced subgraph is connected (with intra- and/or inter-layer links) and the rest of the nodes (not belonging to $MCDS^{ML}$) are adjacent to at least one node belonging to $MCDS^{ML}$.

(b) The number of dominators in each layer is the minimum one.

(c) Having only knowledge of the k-hop neighborhood around each node. Here, we set $k = 2$.

Constraint (a) ensures connectivity of the backbone, and (c) enforces a distributed only approach. Constraint (b) needs some further discussion. We could have simply described it as "the total number of dominators is the minimum one". It is obvious that such a formulation does not imply the one we have used in Definition 3.2, but the reverse is true. Thus we have strived for a stronger formulation which can alleviate problems arising from multilayer networks when their relative size (measured in number of nodes) is highly skewed. Therefore, our definition strives for locating efficient "cross-layer" dominators.

It is easy to prove that our problem is NP-complete [47]. Apparently, solving the same problem for directed (i.e., unidirectional links) and/or weighted (energy considerations on links) versions of networks is also very interesting and subject to solutions not unlike the ones proposed here. Similarly, the problem of stability or incremental maintenance of a discovered ML-CDS (in cases of attacks to nodes, or due to nodes' departures/moves) is also very significant [78], but for the interest of space will not be discussed here.

We admit that we have provided a completely abstract formulation of the problem without taking into account practical constraints/considerations such as types of military formations, physical obstacles and so no; these are reflected in an abstract way into the resulting network topology. Nevertheless, we feel that such considerations will certainly provide optimizations opportunities worth examining in a separate chapter.

3.2.2 Decomposition-based and aggregation-based approaches for DS calculation in multilayer networks will not work

A method that calculates (in a centralized or a distributed fashion) a CDS for each layer separately – thus applying a *decomposition approach* – and then trying to connect them, is clearly a suboptimal solution. This approach is a characteristic case of the problem where we have calculated an unconnected dominating set of a network and we need to find a set of dominatee nodes in order to connect the dominator nodes. In this case, as Theorem 3.1 tells us, the number of nodes that need to be added to the DS in order to become a CDS can be (in the worst case) equal to two times the size of the DS.

Theorem 3.1. *Any (unconnected) dominating set of size $|DS|$ can be turned into a CDS by adding $2 \times |DS|$ additional nodes in the DS in the worst case.*

[Sketch of] Proof.
Firstly, we will state a corollary that results immediately from the domination property, and then we will define the concept of *neighboring dominators* of a dominator v.

Corollary 3.1. *In any dominating set, the closest (in terms of hops) dominator to any dominator can be found at one, two or three hops away, i.e., at most three hops away.*

Definition 3.3. *A neighboring dominator u of a dominator v is any dominator which is at most three hops away from v.*

A dominator v can have more than one neighboring dominators, but the exact number depends on the network topology. Combining Corollary 3.1 and Definition 3.3, we can recognize only three cases that describe the topology between a dominator and its neighboring dominators:

C1 A dominator has at least one neighboring dominator one hop away (dominator S1 – and S7 of course – in Figure 3.2).

C2 A dominator has at least one neighboring dominator two hops away, and none of the rest dominators in one hop distance away (dominator S17 in Figure 3.2).

C3 A dominator has at least one neighboring dominator three hops away, and none of the rest dominators in one or two hops distance away (dominators S10 and S14 in Figure 3.2).

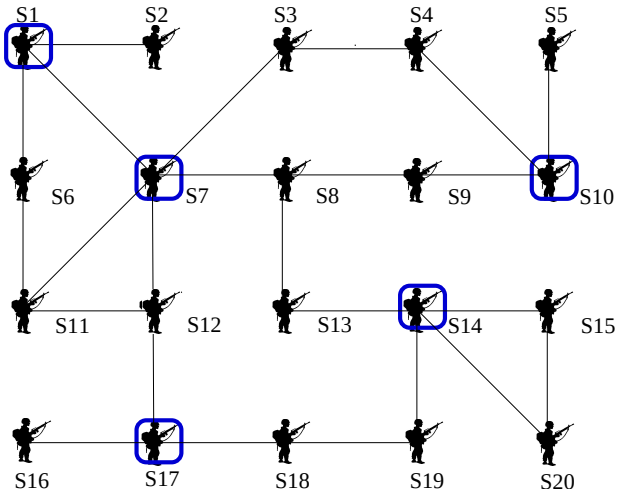

FIGURE 3.2. A dominating set (composed of blue nodes) which exhibits all possible relative locations of neighboring dominators.

If $[C1]$ holds for each and every dominator, the DS is a CDS. If $[C2]$ holds for some dominator v, then we need to include one more dominatee into the DS in order to connect v to its nearer neighboring dominator. Finally, if $[C3]$ holds for some dominator v, then we need to include two more dominatees into the DS in order to connect v to its nearer neighboring dominator. Thus, in the worst case, for every dominator we need to include two more nodes in the DS in order to make it a CDS. The worst case occurs for dominating sets as that shown in Figure 3.3. ⊣

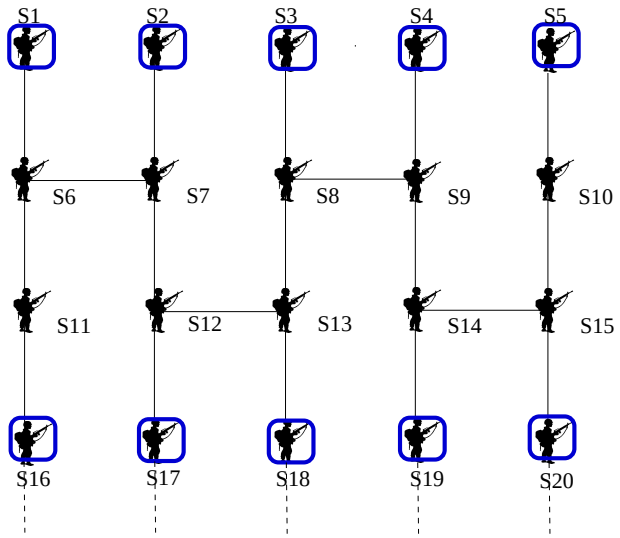

FIGURE 3.3. A dominating set (composed of blue nodes) which requires the maximum number of dominatees that must become dominators in order that the resulting DS is a CDS.

Even though Theorem 3.1 applies in the worst case only, it is quite possible that the military formations in battlefields will make topologies such as that of case $[C3]$ to appear quite frequently. Thus the decomposition-based approaches will create long-and-skinny CDS instead of "bushy" ones, resulting in large communication latencies.

On the other hand, if we apply an *aggregation approach* treating all links the same even though some of them may connect nodes belonging to different layers, will cause other types of problems; looking again at Figure 3.1 and following an aggregation-based method, some algorithm might decide to include node $S4$ into the dominating set, because it is the most connected node in its neighborhood. However, a wiser decision would be to include node $S3$ into the dominating set, because that node connects to node $D3$ and $D4$, with the former providing links to the helicopter-level and the latter being the most connected to its level, thus better facilitating information dissemination across all layers.

Therefore, neither a decomposition- nor an aggregation-based approach would provide efficient solutions to our problem.

3.3 Identifying (efficient) cross-layer dominators

The discussion in the previous section highlighted the significance of assessing and exploiting a node's intra- and inter-layer links in order to be considered as a candidate dominator. Assessing the significance of a node must be quick (in small computational complexity) and cheap (in small communication complexity, which practically implies small energy consumption as well). Therefore, we need to devise a method that will rely on connectivity information from the node's "local" neighborhood (one or two hops away at most), and without computing sophisticated functions for assessing the value of the node. For the case of single-layer networks, a node's degree [48, 99] is a measure that complies with the above requirements, but has several drawbacks [143].

Our plain intuition for selecting nodes that will eventually be "efficient" (i.e., they will cover as much of a network area as possible) dominators is that these nodes should be strategically located in dense areas of the (multilayer) network. In [38] we showed how to identify such nodes in a single-layer network by defining the *Power Community Index* (PCI), which is a node centrality measure.

Definition 3.4 (Power Community Index (PCI))**.** *The PCI index of a node v is equal to k, such that there are up to k nodes in the* 1*-hop neighborhood of v with degree greater than or equal to k, and the rest of the nodes in that* 1*-hop neighborhood have a degree less than or equal to k.*

Now turning to our multilayer network case, we can straightforwardly generalize it for multilayer networks by ignoring(!) the existence of layers; then we get the *Layer-agnostic PCI (laPCI)* defined as follows:

Definition 3.5 (Layer-agnostic PCI (laPCI) [13])**.** *A node has $laPCI$ equal to k, if it has k one-hop neighbors with a number of links towards any layer greater than or equal to k, and the rest of its one-hop neighbors have a number of links towards any layer less than or equal to k.*

laPCI gives credit to a node whose neighbors have many connections in different layers, however, it makes no distinction on how those connections are distributed over the layers, which is problematic. We can cure this, by taking into account the existence of layers:

Definition 3.6 (Minimal-layers PCI (mlPCI) [13])**.** *A node has $mlPCI_n$ equal to k, if it has k one-hop neighbors with the number of links towards at least n layers greater than or equal to k, and the rest of its one-hop neighbors have a number of links toward at least n layers less than or equal to k.*

mlPCI_n characterizes a node for its connectivity in a predefined number of layers. We further combine mlPCI_n values for all n bringing *mlPCI* for a node v in its final form:

$$mlPCI(v) = \sum_{i=1}^{\#layers} mlPCI_i(v) \tag{3.1}$$

$mlPCI$ categorizes as "good" nodes those who are well connected in many layers compared to those who are well connected in a few layers.

A disadvantage of the original PCI (and thus of *laPCI* and *mlPCI*) is that it is mainly based on the connectivity of the nodes that participate in the definition of *PCI*; the connectivity of the rest of the nodes is ignored. We should somehow incorporate this missed topological information into our definitions. We do this for a single layer as follows: we calculate the *PCI* index of a node as usual (using Definition 3.4) and then – after excluding the nodes that contributed to this *PCI* value – we compute a new *PCI* value with the remaining nodes, and add the two *PCI* values. We perform this computation for every layer, and add the resulting indices; we call the obtained number *Exhaustive PCI* (*xPCI*). We calculate *xPCI* for node $S4$ in Figure 3.1 by observing that only nodes $S5$, $S8$ and $S9$ are responsible for defining *PCI*($S4$) at the "Soldiers" layer and node $H2$ at the "Helicopters" layer, thus *xPCI(*$S4$*)*= 5. *xPCI* is not satisfactory as a ranking mechanism because it creates a lot of ties. To this end, for those k nodes that participate in the *xPCI* index, we calculate the number of unique links between them in order to form the final index (actually, we multiply each *PCI* value by log_2 of the number of links to obtain reasonable numbers even for large networks). We call this new measure *Cross-layer PCI (clPCI)*.

3.3.1 Distributed CDS in multilayer networks

Here, we describe a distributed CDS generation protocol which makes use of any of the proposed measures (for illustration purposes, we use *clPCI* in the pseudocode).

Gathering Data:

- Nodes via the exchange of "Hello" messages gather their 1-hop ($N(u)$) and 2-hop ($N^2(u)$) neighborhood connectivity.
- Each node calculates and broadcasts its *clPCI* index. Hence, each node u is aware of the *clPCI* values in $N(u)$.

Node Selection:

- For any node u, nodes in $N(u)$ are sorted in decreasing order of their *clPCI* values.
- Since the algorithm is executed in a distributed fashion, node u first selects as its relays those 1-hop neighbors that have already been selected as dominators by other neighboring nodes (if any).
- While there are still nodes in $N^2(u)$ which are not neighbors to any node in the set of u's relays, select and include in the set of u's relays the next node from $N(u)$ with the largest *clPCI* index that covers at least one new node in $N^2(u)$

It is easy to prove the correctness (i.e., it computes a CDS) of the algorithm. Clearly, the computation complexity of this algorithm is dominated by the sorting process – $O(m \times log_2 m)$, for a node with where m neighbors for the computation of the measures. The communication complexity (per node) is constant (only 2 messages), i.e., $O(n)$ for a n-node network.

3.4 Experimental evaluation

We performed a simulation-based performance evaluation of the proposed methods in MATLAB.

3.4.1 Experimental settings

3.4.1.1 Competitors

As mentioned in the introduction, there is no prior work on our topic; therefore we used as baseline competitors source-initiated versions of the degree [48] and OLSR [99] in the "aggregated" complex network where all layers have been collapsed into a single one. We also include source-initiated versions of all the proposed ones, namely *laPCI*, *mlPCI* and *clPCI*.

3.4.1.2 Performance measures

We use the size of the resulting connected dominating set as the measure that quantifies the performance of the competing algorithms in Figures 3.4 and 3.5, and the same measure per layer in Figure 3.6. Apparently, a small CDS implies small energy consumption, and in most cases it also implies a short latency; the latency depends also of the "shape" of the dominating set (long versus bushy ones).

3.4.1.3 Datasets

Due to the lack of publicly available, real-world military multilayer networks, we developed a generator for multilayer networks in MATLAB. Our aim was to build a generator that could generate in an algorithmic way a variety of multilayer network topologies, so as to be able to explain the obtained results afterwards with respect to the topology. The generator was developed and described in detail in [12], but here, we will present its basic features.

In our topologies each network layer consists of a set of wireless nodes distributed in a two-dimensional plane. Each node has the same *maximum* transmission range R. By proper scaling, we set that all nodes have the same maximum transmission range equal to one. Every pair of nodes whose Euclidean distance is equal to or less than this maximum transmission range are assumed to be connected, i.e., they form a Unit Disk Graph (UDG). So in this way the actual location of nodes is taken into account when computing the connectivity. Moreover, in order to better approach reality where obstacles prohibit the direct communication between adjacent nodes, we used non-uniform intra-layer models to distribute the nodes on the two-dimensional plane, the same way it was done in [89]. The construction of a multilayer network is controlled by the link density in each layer which is expressed by the average degree (D) of each node, by the number of nodes per layer (i.e., size of the layer), and the number of layers (L).

The task of interconnecting the different layers was done with the aid of two parameters: the number of links a node has towards nodes in different layers, while the second parameter involves the distribution of interconnections towards the nodes within a certain layer.

Finally, we may require "coverage" (preference) for a certain layer, that is, nodes generating most of their interconnections towards a specific layer, e.g., most interlinks from the drones layer are generated towards the soldiers network, etc. With the above considerations we apply the Zipfian distribution for our interconnectivity generator. The desired skewness is managed by parameter $s \in (0,1)$. We apply three distinct Zipfian laws, one per parameter of interest:

- $s_{degree} \in (0,1)$ in order to generate the frequency of appearance of highly interconnected nodes,
- $s_{layer} \in (0,1)$ in order to choose how frequently a specific layer is selected,
- $s_{node} \in (0,1)$ in order to choose how frequently a specific node is selected in a specific layer.

In the next section we will represent the values of these parameters (which collectively will be called *topology skewness*) as a sequence of three floats, e.g., 0.5/0.5/0.1, meaning that $s_{degree} = 0.5$, $s_{layer} = 0.5$ and $s_{node} = 0.1$. Finally, in a multilayer network the relative size of the layers would clearly have an impact on the performance of the algorithms. Thus, we equipped our topology generator with the ability to create multilayer topologies where each can be a percentage (10%, 20%, 40%, 70%) larger than the previous one. So we may have topologies with relatively equi-sized layers (10%), or topologies with huge layer inequalities (70%).

Table 3.1 records all the independent parameters of our topology generator, their range of values, and their default values.

TABLE 3.1. Experimentation parameters values.

parameter	**range**	**default**
avg. node degree (D)	6, 8, 10, 12, 16	10
network diameter (H)	7, 15, 30, 40	7
# network layers (L)	2, 3, 4, 5, 6, 7	5
topology skewness	0.1/0.1/0.1 0.1/0.9/0.1 0.1/0.5/0.5 0.5/0.5/0.1 0.9/0.5/0.1	0.1/0.5/0.5
relative size of one layer to its previous	10%, 20%, 40%, 70%	40%

3.4.2 Experimental results

3.4.2.1 Impact of topology density

Firstly, we wish to evaluate the impact of topology density on the performance of the algorithms. To keep the experiment controlled, we vary the density of a single layer keeping the rest unaltered to the extent possible. The results are illustrated in the bottom plot of Figure 3.4.

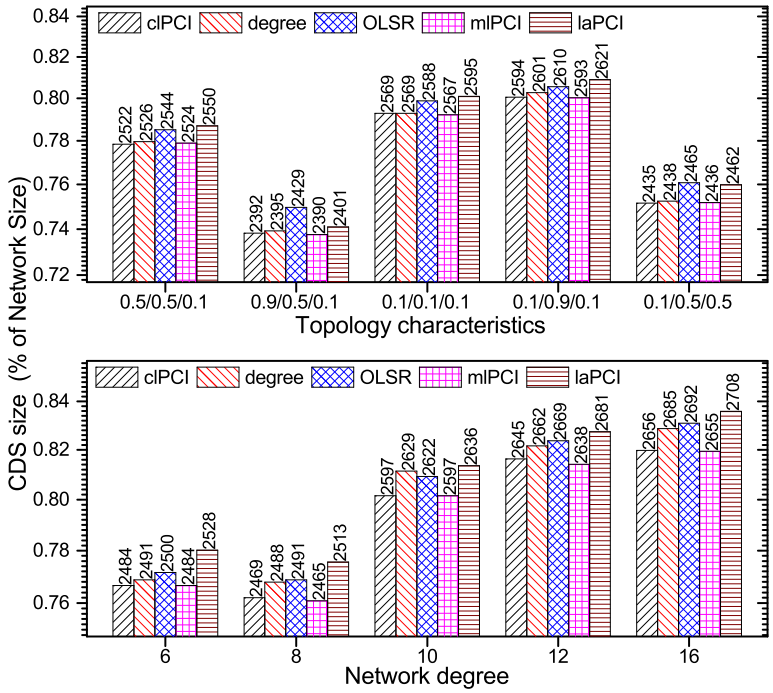

FIGURE 3.4. Impact of network density and topology skewness on the size of CDS.

Our first observation concerns the size of the generated CDS formed by each competitor; one would expect that a higher network density would decrease CDS size, but here we observe that CDS size increases. This is due to the fact that the inter-layer links are spread more uniformly among layers, and thus there are no 'hub nodes' whose existence will result in a decreasing CDS size for increasing density. Concerning the performance of the competitors, $clPCI$ and $mlPCI$ produce the smallest CDSs for all D values, but none of them are better than the other. Moreover, their performance gap from the third best performing algorithms widens with increasing density, which is due to the fact they exploit the inter-layer links to identify cross-layer dominators. Now looking at the upper part of Figure 3.4, where the champion algorithms remain the same as before, we observe the generic trend that CDS size for all competitors increases when the topology skew is small, i.e., s_{degree} and s_{node} have small values. Only, when there are "hub nodes" i.e., $s_{degree} = 0.9$ the size of CDS is small.

3.4.2.2 Impact of network diameter

In Figure 3.5 we evaluate the effect of the multilayer network diameter in the size of the CDS. At this point we need to say, the each layer is composed of around 500 nodes, and n-layered network is composed of the previous $n-1$ layers plus one more layer. As the network diameter increases the size of the constructed CDS for all methods decreases. The decrement of the diameter is the result of sparser vicinities, i.e., fewer links between the network nodes. In other words, fewer, longer (in hops), and more distinct paths towards the nodes of the multilayer network, which renders the election of those nodes that cover the N^2 neighborhood more discrete, and hence fewer nodes are recruited. Focusing on the evaluation of the competitors, we observe that the difference in their performance is minimum when $H = 7$. This is due to fact that when nodes are relatively "close" to each other, there is significant overlapping in the selected CDSs.

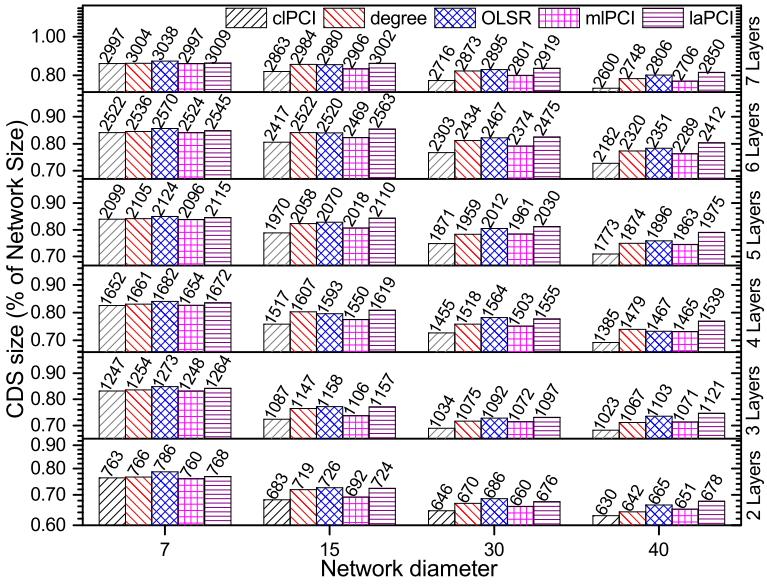

FIGURE 3.5. Impact of network diameter on the CDS size.

3.4.2.3 Impact of increasing the layer size

Figure 3.6 illustrates the impact of the number and size of layers on CDS size; as expected, the generic trend is that the size of CDS increases with more layers or more variability in the relative layer size. We need to say here, that the top layer (Layer 5) in each *5-layered* network is composed of 500 nodes and the remaining layers have increased size with respect the previous layer as depicted in the x-axis. The purpose of Figure 3.6 is to clarify that the performance of the proposed methods is the result of selecting a minimum dominating set in each respective layer, which due to a careful selection of key intra- & interconnected nodes, results in an interconnected dominating set, i.e., an $MCDS^{ML}$. Evidently, as the size of each layer increases, so does the

cardinality of the elected CDSs for all methods. The competitors's ranking obtained from the previous subsection has remained unchanged, i.e., $clPCI$ selects the smallest CDS in all layers, and thus the overall minimum $MCDS^{ML}$, which highlights the effectiveness of the proposed technique.

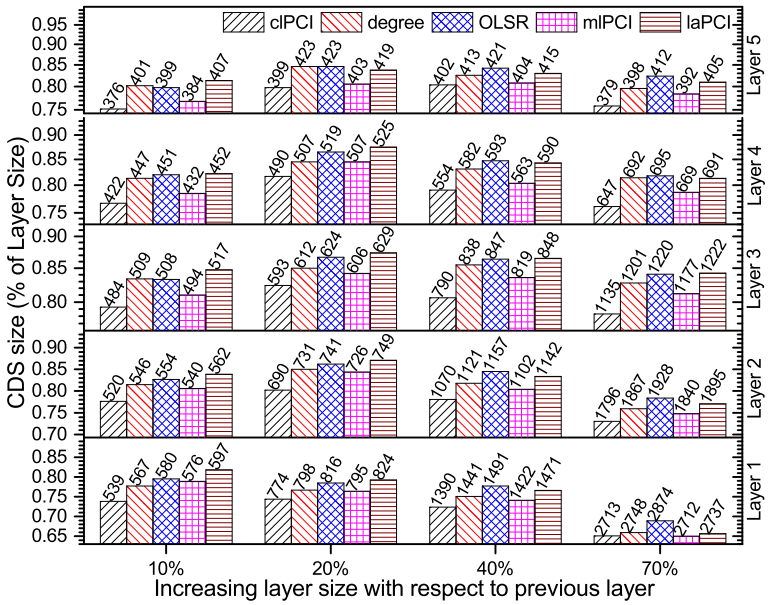

FIGURE 3.6. Impact of network size on the CDS size.

3.5 Conclusions

We considered the problem of backbone formation for modern military ad hoc networks which are composed by multiple subnetworks ("layers" in this chapter's terminology). We investigated the possibility of forming the backbone in terms of connected node dominating sets, and subsequently we defined – for the first time in the literature – the problem of minimum connected node dominating set for multilayer networks. We recognized the significance of determining "efficient" cross-layer dominators, and proposed a set of measures (based on network theory concepts) for detecting them, namely $laPCI$, $mlPCI$ and $clPCI$. Then, we proposed a distribution algorithm for backbone formation based on those measures. We performed a simulation-based evaluation of the proposed techniques against baseline methods (i.e., degree, OLSR) and showed that the distributed algorithm based on *clpci* shows (almost always) the best performance.

CHAPTER 4

ENERGY-AWARE BACKBONE FORMATION IN MILITARY MULTILAYER AD HOC NETWORKS

4.1 Introduction

In this chapter we delve deeper into the world of military multilayer ad hoc networks and explore their unique characteristics compared to the traditional wireless ad hoc networks [45, 50]. Apart from the broadcast-nature of the wireless communication medium and mobility which are very common features, these ad hoc networks are usually very large in terms of the number of participating nodes. Therefore, more critically than for "plain" ad hoc networks, we need to ensure protocol scalability in the number of nodes. Moreover, we must carefully consider for reduced delays and for the scarce energy resources. Additionally, due to the dynamic topology, protocols must be based on primitives that are feasible and efficient to compute in a distributed manner, and also to engage only computations based on localized information. Most important though is to consider the nature of the network itself which usually consists of "subnetworks".

In the context of the military multilayer ad hoc networks we take into consideration the JTRS multilayer network paradigm [109] to build upon. Thus, we consider "island" subnetworks as being the layers of a single, large network, which we call a multilayer communication network. To make clear this nature, we show in Figure 4.1 a mixed military unit consisting of a tank platoon belonging to some tank company, and an infantry squad belonging to some infantry platoon.

Related publication **[J2]:** Dimitrios Papakostas, Soheil Eshghi, Dimitrios Katsaros, Leandros Tassiulas. "*Energy-Aware Backbone Formation in Military Multilayer Ad Hoc Networks*", **Ad Hoc Networks (Elsevier)**, vol.81, pp.17-44, December, 2018.

These two units communicate wirelessly via an ad hoc network and advance in the battlefield pursuing some common operational goal.

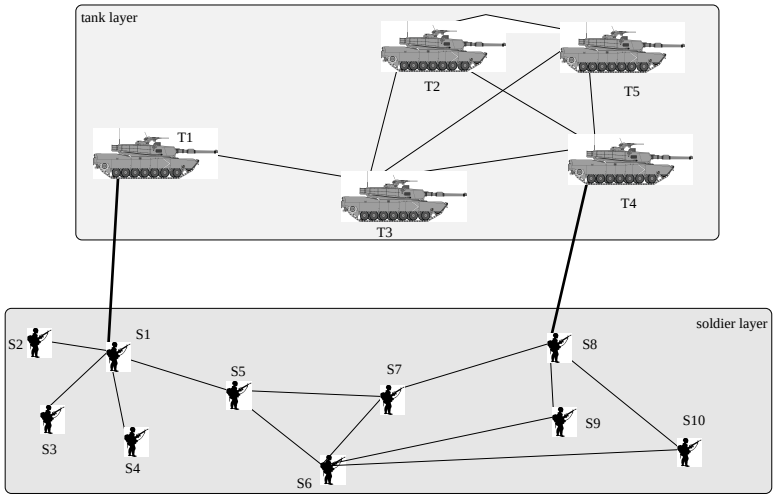

FIGURE 4.1. Abstraction of a military multilayer ad hoc network comprised by 2 layers. Physical obstacles have been removed, and the entities have been projected onto the 2D space.

In this wireless network we would recognize two "subnetworks", namely the tank layer and the soldiers' layer; for various reasons related to military strategy and hierarchy and terrain topology, the only links are those shown in the figure. Earlier methods did not allow a node to participate into two "networks" at the same time, but recent progress in networking could sustain such situations. In [92] we used terminology from complex networks literature to describe the topology of such ad hoc networks, and we will use here the same terminololy. Thus, we recognize two network *layers*, *intra-layer* connections (the thin ones) connecting entities of the same type, and *inter-layer* links (the tick ones) connecting entities belonging to different layers.

Diverse types of military units participate in modern battlefields that are energy-constrained, such as soldier, drones, still sensors, and so on. Therefore, VBN construction protocols must be energy-aware. In this chapter, we investigate the topic of energy-efficient VBN construction for military multilayer networks using CDSs constructed in a distributed fashion. In principle, any efficient algorithm for calculating a MCDS seeks to detect nodes strategically positioned in the topology in order to include them into the DS and thus decrease the size of the obtained DS, because they "dominate" over a large number of other nodes. For instance, some algorithms for single layer networks achieve this by looking at the degree of each node [48]. Moreover, if while searching for nodes to include in the DS we include as criterion apart from their strategic position, their residual energy, then we can develop energy-aware algorithms for DS construction.

4.1.1 Motivation and contributions

The architecture of the multilayer networks poses some unique challenges which make topology control a "tricky" task. We proved in [92] that solutions based on decomposition and/or aggregation of the entire multilayer network are not efficient; there is significant room for improvement if we take into account the multilayer structure. Secondly, assigning different weights on intra-layer versus inter-layer links can not help transform our problem at hand into that of dealing with the calculation of a weighted DS of the multilayer network, because there is no algorithmic method yet for the determination of the relative weights so as to produce an efficient backbone for multilayer networks. Finally, energy-related issues have not been investigated for DS-based VBN construction algorithms for multilayer networks, even though there is work on energy-agnostic protocols [92].

In this context, the present chapter makes the following contributions:

- it investigates the issue of energy-aware connected dominating set-based backbones for multilayer networks;
- it generalizes an earlier proposed centrality measure for identifying nodes with high residual energy and central position within the multilayer network;
- it develops a distributed algorithm, namely *E2CLB* which is based on the aforementioned centrality measure for identifying dominating nodes;
- it analyzes the algorithm's performance both from a computational/communication complexity perspective and an experimentation-based perspective;
- it compares exhaustively the proposed algorithm against relevant and baseline competitors, because there is no prior work on the chapter's subject.

The rest of this chapter is organized as follows: Section 4.2 introduces in formal terms the problem of constructing energy-efficient connected dominating sets for multilayer networks; Section 4.3 proposes a locally computable measure to assess the significance of a node in participating in an energy-efficient connected dominating set; Section 4.4 develops a distributed algorithm for calculating the energy-efficient connected dominating set; Section 4.5 provides performance evaluation results comparing the proposed algorithm against competitors, and finally Section 4.6 concludes the chapter.

4.2 Problem formulation

The main elements of the architecture of a multilayer ad hoc network are the following: the network consists of a set of nodes, each one beloning to some layer, and having an amount of energy associated with it which is described by a scalar quantity. Each node has a non-empty set

of connections towards (some) nodes belonging to the same layer (intra-layer links), and it has a (possibly empty) set of connections towards nodes belonging to other layers (inter-layer links). All links are assumed to be bidirectional.

We will now describe our setting using graph-theoretic terms. A multilayer network which consists of n layers is a pair (G^{ML}, E^{ML}), where $G^{ML} = \{G^i, i = 1, \ldots, n\}$ is a set of "networks" (G_i, E_i) ($|G_i|$ nodes belonging to layer i, and $|E_i|$ edges connecting nodes belonging to layer G_i), and a set of inter-layer links $E^{ML} = \{E_{i,j} \subseteq G_i \times G_j; i, j \in \{1, \ldots, n\}, i \neq j\}$. Moreover, each node is annotated with a scalar quantity which represents its residual energy. Then, the problem of finding an energy-efficient backbone network based on dominating sets for multilayer networks in a distributed fashion can be described as follows:

Definition 4.1 (The Multilayer Maximum Energy MCDS (ML-MEMCDS) problem)**.** *The problem of calculating a Maximum Energy MCDS (MEMCDS) for a multilayer network* (G^{ML}, E^{ML}) *consists of finding a subset* $MEMCDS$ *of its nodes such that the following conditions hold:*

1. *Each node of* G^{ML} *either belongs to* $MEMCDS$ *or is adjacent to (in one hop distance from) a node belonging to* $MEMCDS$*.*

2. *The cardinality of set* $MEMCDS$ *is the minimum possible.*

3. *The nodes comprising* $MEMCDS$ *are connected to each other, i.e., there is path from any node* $i \in MEMCDS$ *to any node* $j \in MEMCDS$, $\forall i, j$*. [Intra-layer or inter-layer links may comprise that path.]*

4. *The sum of energies of nodes belonging to* $MEMCDS$ *is the maximum possible.*

5. *Knowledge of only the closest* k*-hop neighborhood of a node is permitted.*

Corollary 4.1. *The problem ML-MEMCDS is NP-complete.*

The proof is trivial [47] and thus we omit it. Versions of the problem with directed links, with incremental maintenance of its solutions in cases of nodes/links additions/removals will be examined in subsequent works.

We proved in [92] (Theorem 1) that finding a minimum connected dominating sets for every layer and then trying to connect them does not provide efficient solutions in terms of minimizing the cardinality of the dominating set. Similar observations were made in [92] for methods based on ignoring the layer information and calculating connected dominating sets in the "aggregated" network. It is easy to extend those results for our case where energy issues are present. Thus, in the next two sections, we will present an efficient heuristic solution to this problem that considers the layering information.

4.3 Identifying energy-rich cross-layer relay nodes

In this section we will introduce a locally computable measure to identify prominent nodes to be included in the $MEMCDS$. We build upon the PCI family of centrality measures [12] and provide a generalization of $clPCI$ to solve the problem. To elaborate, in [92] we developed a backbone construction algorithm based on clPCI which was energy-agnostic, i.e., all nodes were assumed to somehow replace the energy they deplete, e.g., by fuel, solar panels, etc. However, in the generic case, energy issues do need to be considered for ad hoc connectivity, especially in battlefields [122]. Thus, we provide here the $EnergyawareclPCI(EclPCI)$ which, for a node u with energy equal to $E(u)$ is defined as follows:

$$EclPCI(u) = E(u) \times clPCI(u). \tag{4.1}$$

When energy is not an issue, then clearly $EclPCI \equiv clPCI$. Algorithm 4.1 presents a distributed algorithm for the calculation of $EclPCI$ of node u.

Algorithm 4.1: $EclPCI$ index value calculation.

precondition : Known 1-hop ($N(u)$) and 2-hop ($N^2(u)$) neighbor connectivity info (ID) of node u
postcondition: Calculation of the $EclPCI$ index value of node u
remarks : m = number of layers in the multilayer network, $layer(u)$ = network layer that node u is situated, $E(u)$: residual energy of node u, S = node set, $PCI(u)$, $xPCI(u)$, $clPCI(u)$, $EclPCI(u)$: indexes related to node u

```
1  for layer i ← 1 to m do
2      PCI(u) = xPCI(u) = 0;
3      Build S = u_1, u_2, ..., u_n | u_{k (1≤k≤n)} ∈ N(u), layer(u_{k (1≤k≤n)}) = i;
4      while S ≠ empty do
5          Calculate PCI(u) for S;
6          Calculate unique links (Links_unique) of nodes participating in PCI(u);
7          xPCI(u) += PCI(u) * log2(Links_unique);
8          Remove nodes that participated in PCI(u) from S;
9          PCI(u) = Links_unique = 0;
10     end
11     clPCI(u) += xPCI(u);
12 end
13 EclPCI(u) = E(u) * EclPCI(u);
```

Note that because $EclPCI$ is calculated on a *per layer* basis it is possible to present some sort of preference to one or more layers. For example, if the multilayer network incorporates a layer with nodes with no energy issues, then it might be a wise decision to have many relay nodes in that layer. This capability however is out of the scope of the present work and it will not be examined further. In case of ties due to Equation 4.1, the selection of relay nodes may be random, or in an application-dependent way, e.g., prefering energy-rich nodes over well connected for resource-scarce environments.

Proposition 4.1. *The computation complexity of $EclPCI$ index calculation is $O(\Delta^2)$ in the worst case, where Δ is the maximum node degree in the network.*

Proof. The worst case regarding the computation complexity of the $EclPCI$ index calculation is when a host u has Δ neighbors and each one of them has Δ neighbors too; i.e $PCI(u) = \Delta$. In such case and during the calculation of the unique links among neighbors, a host u needs to compare its neighbor set with Δ neighbors and the neighbor set comparison has a complexity of $O(\Delta)$. $\dashv$

Energy-related augmentation can be applied to $mlPCI$ as well, and in that case we get the $EmlPCI$ measure. Now, armed with a method to identify energy-rich nodes whose connections span many nodes in many layers, we are ready to describe a distributed algorithm for calculating an energy-efficient connected dominating set.

4.4 Distributed energy-efficient backbone formation algorithm

$EclPCI$ which was described in previous section is actually a *centrality* measure that identifies those nodes of the network which have high energy levels and at the same time maintain a strategic/central position among the network layers. In principle, any efficient heuristic algorithm for calculating a minimum connected dominating set seeks to detect such strategically positioned nodes in order to decrease the size of the obtained dominating set. Some algorithms for single layer networks achieve this by looking at the degree of each node [48].

Thus, we exploit the $EclPCI$ measure and incorporate it into a distributed algorithm for computing an energy-efficient connected dominating set. The algorithm will be called $Energy-EfficientCrossLayerBackbone(E2CLB)$ formation algorithm. In principle, a backbone based on the formation of a CDS whose elements are such well-connected nodes can turn them into hotspots. There are several solutions proposed in the literature [140] that can alleviate these kinds of problems, e.g., role rotation, movement control and so on; in general it is a well addressed problem and therefore we will refrain from replicating the details of such mechanisms here. Additionally, we need to mention that making $E2CLB$ able to work for unidirectional links, or weighted links (e.g., using the weight to depict variation in energy consumption during communication) is straightforward by simply incorporating direction/weight in the calculation of $EclPCI$. Such adaptations are abundant in the literature for centrality measures [41], and thus we skip the relative discussion here. Before delving into technical details of the proposed algorithm, we will first provide a brief overview of the algorithm, and then we will describe its constituent parts. In $E2CLB$, there are mainly three phases, which are the following:

- CDS construction phase.
- Redundant relay node pruning.
- Mediator phase.

Before these three steps take place, one more procedure evolves that is typical and common in (almost) all distributed algorithms for ad hoc networks with non Global Positioning System (GPS)-enabled nodes. During this process, each node learns the topology of its neighborhood, and also other interesting features (e.g. residual energy) of its neighbors. For $E2CLB$, each node learns the connectivity and residual energy of all its neighbors up to its 2-hop neighborhood $N^2(u)$; this preparatory phase will not be described in details since it is very common.

The CDS construction phase is based on a *source-initiated* relay node selection process that is executed by every node u. Because this selection process produces many redundant CDS nodes, a pruning phase follows. Finally, in order to exploit the connectivity among nodes that belong to the same relay node set and improve the minimum residual energy level of a node in the set, one more phase called the mediator phase is employed which is based on some heuristic rules.

Algorithm 4.2: Relay node set election.

precondition : Known 1-hop ($N(u)$) and 2-hop ($N^2(u)$) neighbor connectivity info (ID) of node u
postcondition : Elected relay node set ($R(u)$) of node u
remarks : $EclPCI(u)$: index related of node u, $M(u)$: status of node u with regards to being [True (T)] or not [False (F)] a relay node

1 Calculate and broadcast own $EclPCI$ index value;
2 Gather the $EclPCI$ index values of the nodes in $N(u)$;
3 Sort nodes in $N(u)$ in decreasing order of their $EclPCI$ index values;
4 **repeat**
5 | Select the node from $N(u)$ with the largest $EclPCI$ index value that covers at least one new node in $N^2(u)$;
6 | Include the selected node in $R(u)$;
7 **until** *each node in* $N^2(u)$ *has at least one neighbor in* $N(u)$;
8 Broadcast $R(u)$;
9 **if** *selected as a relay node and* $M(u) = F$ **then**
10 | $M(u) = T$; /* node becomes a relay node */
11 | Broadcast status change;
12 **end**
13 Update 1-hop neighborhood node status (if req);

4.4.1 CDS construction phase

CDS construction (Algorithm 4.2) is divided into two tasks, namely *neighbor prioritization* and *construction* task. During neighbor prioritization task, every node u calculates its own $EclPCI$ index and it broadcasts its value in a single message to its neighbours. By mutuality of the distributed protocol, it receives its neighbors' $EclPCI$ values. Then, it sorts the nodes in $N(u)$ in non-increasing order of their $EclPCI$ value. In the construction stage, each node u selects from $N(u)$ and includes in its relay node set $R(u)$ the nodes with the largest $EclPCI$ index value that cover at least one new node in the $N^2(u)$ neighborhood. Using the proof methodology of [37,

Theorem 4.2], we can easily prove that the resulting relay node sets of all the network nodes form a CDS.

Proposition 4.2. *The computation complexity of the relay node set election process is* $O(\Delta^3)$*, where* Δ *is the maximum vertex degree in the network.*

Proof. The prioritization phase involves neighbor sorting based on $EclPCI$ value, which is a $O(\Delta * log\Delta)$ operation. The worst case regarding the construction phase results when a host u has Δ neighbors and each one of them contributes Δ nodes to the coverage of the 2-hop neighborhood of u. In this case, host u needs to run once over its neighbor set of size $O(\Delta)$ and "erase" those nodes of the 2-hop neighborhood of u (which has maximum size $O(\Delta^2)$) covered by the specific neighbor; therefore, this operations costs $O(\Delta^3)$, i.e., the total cost progresses as follows: $\Delta^2 + (\Delta^2 - \Delta) + (\Delta^2 - 2\Delta) + \cdots + (\Delta^2 - (\Delta - 1)\Delta)$. ⊣

Algorithm 4.3: The pruning phase.

precondition : Completed relay node set election process from 1-hop neighbors
postcondition: Node updated status
remarks : $T_{pruning}$: a timer, $S_{constrained}$: a node set, $M(u)$: status of node u with regards to being [True (T)] or not [False (F)] a relay node

```
1  Start T_pruning;
2  Build S_constrained = u_1, u_2, ..., u_n | u_k (1≤k≤n) ∈ N(u) ∧ N^2(u),
     M(u_k (1≤k≤n)) = T, EclPCI(u) < EclPCI(w_k (1≤k≤n));
3  if S_constrained is subject to N(u) ⊂ N(u_1) ∪ N(u_2) ... ∪ N(u_n) and u_1, u_2, ..., u_n form a connected graph
     then
4      Wait for expiration of T_pruning;
5      if M(u_k (1≤k≤n)) = T then
6          M(u) = F ;                                  /* node becomes a plain node */
7          Broadcast status change;
8          Exit pruning stage;
9      else
10         Restart pruning stage;
11     end
12 else
13     M(u) = T ;                                      /* node remains a relay node */
14     Broadcast status;
15     exit pruning stage;
16 end
```

4.4.2 Pruning phase

It is known that distributed, source-initiated dominating set construction algorithms produce DSs with many redundant nodes [100],[116]. For our needs, we design a distributed pruning

phase, which is executed by relay nodes only. Each candidate relay node is aware of its status, i.e., being a relay or not due to step 8 of Algorithm 4.2. Each relay node waits until all its one-hop neighbors decide their "relay status" before it enters the pruning phase (Algorithm 4.3).

Moreover, in order to confront the case where more than one relay nodes enter the pruning phase simultaneously, we "prioritize" the execution of the pruning rules in such a way that relay nodes with smaller residual energy level execute it earlier than relay nodes with larger residual energy level. In order to do that each node i whenever is selected as a relay node it calculates a backoff time according to Equation 4.2:

$$T_{pruning} = \frac{E(i)}{|R(u)|} + \ell, \tag{4.2}$$

$E(i)$ is the residual energy of node i, and $|R(u)|$ is the cardinality of the relay node set of node u that node i participates in; ℓ is a unique pseudo-random number calculated by node i in the range $[0, 0.1]$ that is used in order to solve any ties between relay nodes in $|R(u)|$ with the same residual energy level.

So, each relay node before starting to execute the pruning rules waits first for the backoff time to expire. In the case where more than one nodes have selected the same node i as a relay node, that relay node will calculate more than one backoff times, i.e., a backoff time of each different relay node set that i participates in, but use during the pruning stage only the smaller backoff time between them. It is interesting to notice that the backoff time formula favors the elimination of relay nodes that have either small residual energy and/or belong to a relay node set with many participants. To achieve a good balance between efficiency and overhead in our work we make use of the *restricted pruning Rule k* as this self pruning scheme, in general, is more efficient in reducing the relay node set than several existing schemes that ensure the broadcast coverage [133]. This rule can be implemented with knowledge of either 2-hop or 3-hop neighborhood [33]. In the pruning rule we make use of connectivity as quantified by $EclPCI$ as priority value in order to establish a total order among nodes that participate in the CDS. Connectivity has been proved to be the most efficient priority under all circumstances [133]. The complete pruning phase is depicted in Algorithm 4.3.

Proposition 4.3. *The computation complexity of the pruning phase is $O(\Delta^3)$, where Δ is the maximum vertex degree in the network.*

Proof. A relay node u in order to decide if it will act as a relay node or not it needs to calculate the coverage capability of a connected graph composed of both 1-hop and 2-hop neighbors. Thus, each relay node u compares its neighbor set with Δ^2 neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. ⊣

4.4.3 The mediator phase

The central idea of this phase is to further reduce the relay node set by examining if a particular relay node can be accessed through another relay node; we call this relay node a *mediator* (Algorithm 4.4). The mediator heuristic is employed sequentially to relay nodes of the same set, in increasing order of their $EclPCI$ value. Thus, a relay node i with smaller $EclPCI$ index value than other nodes from the same relay node set will be examined first if it can be reached through another relay node, and if so, it will be removed from the respective relay nodeset iff it has less residual energy from the relay node that will act as a mediator.

Algorithm 4.4: The mediator phase.

precondition : Completed pruning process from 1-hop relay nodes
postcondition: Updated relay node set
remarks : $T_{mediator}$: backoff timer, S_{relays}: node set, $R(u)$: relay node set of node u, $M(u)$: status of node u with regards to being [True (T)] or not [False (F)] a relay node

1 Start $T_{mediator}$;
2 Update $R(u) = u_1, u_2, ..., u_n \mid u_{k\ (1\leq k\leq n)} \in N(u), M(u_{k\ (1\leq k\leq n)}) = T$;
3 Sort nodes in $R(u)$ in increasing order of their $EclPCI$ index value;
4 Broadcast $R(u)$;
5 Set $S_{relays} = R(u)$;
6 Sort nodes in S_{relays} in increasing order of their residual energy;
7 Wait for expiration of $T_{mediator}$;
8 **repeat** $\forall$ node $v_{k\ (1\leq k\leq n)}$ in $R(u)$, in increasing order of their $EclPCI$ index value
9 **repeat** $\forall$ node $w_{k\ (1\leq k\leq n)}$ in S_{relays}, in increasing order of their residual energy level
10 **if** $E(w_k) > E(v_k)$ *and* $v_k \in R(w_k)$ **then**
11 remove v_k from $R(u)$;
12 Broadcast $R(u)$;
13 Set w_k as a mediator to get to v_k;
14 break;
15 **else**
16 select the next node from S_{relays};
17 **end**
18 **until** *Until all nodes in* S_{relays} *are checked*;
19 select the next node from $R(u)$;
20 **until** *Until all nodes in* $R(u)$ *are checked*;

Moreover, with the intention of avoiding race conditions regarding a relay node that is included in more than one relay node sets we resorted to prioritizing the removal of the relay nodes in such a way that nodes who have smaller $EclPCI$ index value take higher priority to decide about their relay node sets than other nodes that have larger $EclPCI$ index value. In order to do that, each node u calculates a backoff time and executes the mediator heuristic right after the expiration of the respective $T_{mediator}$ timer.

The mediator backoff time is calculated with Equation 4.3.

$$T_{mediator} = \frac{EclPCI(u)}{|R(i)|}. \tag{4.3}$$

$E(u)$ is the residual energy of node u and $|R(i)|$ is the cardinality of the relay node i that is under consideration to be removed (it is used for normalization purposes). All in all, the mediator heuristic is an indirect approach to sustain as long as possible the number of alive nodes in the network [75] [56] or equally the fraction of alive nodes [136]. Next, we present the pseudocode of the mediator heuristic.

Proposition 4.4. *The computational complexity of the mediator phase is* $O(\Delta^2 \times log\Delta)$ *in the worse case, where* Δ *is the maximum degree in the network.*

Proof. In the worst case, a node with degree equal to Δ will have Δ relays. Thus, after sorting them (with cost $\Delta \times log\Delta$) a serial scan over them takes place with cost $O(\Delta)$ and while scanning each, a new sorting over the rest relays is performed with cost $O(\Delta \times log\Delta)$. ⊣

4.4.4 Communication overhead of E2CLB

The following theorem presents the communication overhead and latency (in terms of information exchange) of the proposed algorithm.

Proposition 4.5. *In bidirectional networks, the execution of* $E2CLB$ *algorithm requires* 7 *rounds to complete.*

Proof. The 2-hop information used by the relay node set election process can be collected via two rounds of information exchanges. In round 1, each node advertises its ID and residual energy level and builds its 1-hop neighbor set based on the advertisement of its neighbors. In round 2, each node advertises its 1-hop neighbor set and identifies links among 1-hop neighbors. These two rounds are present in any distributed protocol where the nodes need to become aware of their neighborhood. In round 3, each node calculates its $EclPCI$ index value and advertises it together with its 2-hop neighbor set. Then it identifies links among 2-hop neighbors. In round 4, each node calculates and advertises its own relay node set and updates 1-hop neighbor status. In round 5, the restricted Rule k is applied to each relay node and each one of them advertises its status. In round 6 each node advertises its updated relay node set and applies the mediator heuristic to each one of the participating relay nodes. Finally, in round 7 the composition of the updated relay node set is advertised (if needed). ⊣

4.5 Performance evaluation

In this section we will present the details of the evaluation setting and illustrate the results. In particular, in subsection 4.5.1 we present the competing algorithms, and in subsection 4.5.2

we give the performance measures of the comparison. In subsection 4.5.2.1 we describe the network topologies used in our simulation, and in subsection 4.5.3, we present and comment on the obtained results.

4.5.1 Competing algorithms

The first thing to note is that, instead of $EclPCI$, we can use in its position the $EmlPCI$ measure and thus get the Energy-Efficient MultiLayer Backbone (E2MLB) formation algorithm ; or we can use the $clPCI$ measure – which does not take the residual energy of a node into account – and get the Energy-Unaware Cross Layer Backbone (EUCLB) formation algorithm which is actually the algorithm proposed in [92]. These two algorithms along with some baseline ones that will be described in the next paragraph will be used as competitors to $E2CLB$.

Degree-based CDS construction is a very popular technique, and thus we looked for generalizations of degree centrality in multilayer networks. We call the respective competitor as Energy-Efficient Weighted Degree Backbone (E2WDB) which uses a generalized notion of degree found in [90]. This algorithm uses the same mechanics as $E2CLB$ to create the CDS; in particular it uses 2-hop connectivity information and it incorporates the pruning phase. However, in its plain version it does not include the mediator heuristic, but its enhanced version called $E2WDB$* does include this heuristic.

The next competing algorithm is based on Tang et al. algorithm to form a $MCDS$ [120], namely $EMCDS$. This algorithm is not localized, as it requires global information to compute the relay node set. However, it can produce a near-optimal forward node set. Here we use it as a substitution of a "perfect" algorithm that produces the optimal result both in terms of the size of the CDS constructed and the energy efficiency. It is emphasized that $EMCDS$ is based on an 1-hop connectivity info in order to build the CDS. The impact of the mediator heuristic on the performance of $EMCDS$ is presented separately under the algorithm $EMCDS$*.

4.5.1.1 Analytic computation and communication complexity of the competitors

Aparently, $E2MLB$ and $EUCLB$ present the same communication overhead with $E2CLB$, because they use the same heuristics during the CDS construction. On the other hand, $EMCDS$ communication overhead varies according to the position of the most energy efficient node in the network [120]. To detect this in a distributed fashion, we need $O(n * logn)$ [129] messages by constructing some spanning tree, and then we need $O(diameter)$ rounds for termination where each node sends $O(1)$ messages. Therefore, $EMCDS$ has $O(n * logn)$ message complexity, and $O(diameter)$ delay. $E2MLB$ and $EUCLB$ present the same computation complexity with $E2CLB$; $E2WDB$ variations and $EMCDS$ variations have $O(\Delta)$ cost.

4.5.2 Performance measures

So far the evaluation of an energy-efficient backbone construction algorithm in a routing protocol-independent way is done according to one of the following ways: i) the first node to die, ii) the number (or fraction) of alive nodes, iii) the time until the network fails to construct a backbone, iv) the fraction of connected dominating set nodes that remain alive, v) the time until the packet delivery ratio drops "drastically". In this work we employ several detailed – and not simply gross – generalized performance measures which are described in the sequel. Competing algorithms are compared in terms of the size of the CDS, the mean *per node* minimum node energy in the relay set, the mean cardinality of each relay node set, and the message complexity to build each relay node set. We say an algorithm is more efficient than another algorithm if it generates a smaller CDS [120, 143]. Additionally, an algorithm that manages to establish *per node* a relay set with larger minimum residual energy level is considered to be more energy efficient than another algorithm whose *per node* relay set includes relay nodes with less residual energy; this measure is a direct approach to define the network lifetime. Moreover, we use the size of the relay set as another performance measure, as the smaller the relay set *per node*, the smaller the volume of broadcast message transmissions in the network is, which subsequently translates into a reduction in node interference, bandwidth usage, and energy savings for the non-relay nodes.

4.5.2.1 Datasets

Due to the lack of publicly available, real world military multilayer networks, we created multi-layer weighted networks in MATLAB [12]. The alternative of developing our solutions over an "programmable" network, such as Software-Defined Networking (SDN) – although appealing at first glance – suffers mainly from the fact that it does not allow for an extensive experimentation with large scale topologies with varying connectivity. Existing wireless testbeds, e.g., NITOS [1] although being useful experimental facilities, they can not serve our aforementioned goals.

Note that, in this chapter, in order to have full control over the network topology and on the way energy (i.e., weights) is distributed among nodes we apply four distinct *Zipfian* distributions (compare with §3.4.1.3), which can produce from uniform to highly skewed distributions for every parameter of interest. The desired skewness is managed by parameter $s \in (0,1)$. We apply :

- $s_{degree} \in (0,1)$ in order to generate the frequency of appearance of highly interconnected nodes,
- $s_{layer} \in (0,1)$ in order to choose how frequently a specific layer is selected,
- $s_{node} \in (0,1)$ in order to choose how frequently a specific node is selected in a specific layer,
- $s_{weight} \in (0,1)$ in order to choose how much uniformly weights are distributed in the multilayer network.

[1] `https://nitlab.inf.uth.gr/NITlab/nitos`

Moreover, we use two different approaches to apply the $Zipfian$ laws; i.e., by selecting nodes either in increasing or decreasing order of their degree. We selected a default setting for each of the parameters of interest and created various datasets that we used to evaluate the efficiency of each competing algorithm. Collectively, we call these parameters as the *topology skewness*, and represent it as a sequence of four floats, e.g., 0.5 – 0.5 – 0.5 – 0.5, meaning that $s_{degree} = 0.5$, $s_{layer} = 0.5$, $s_{node} = 0.5$ and $s_{weight} = 0.5$ (which are the default settings we used to create the datasets). We perform experiments and present the performance of the competing algorithm when using datasets which differ in the topology skewness settings. In a multilayer network the relative size of the layers clearly has an impact on the performance of the algorithms. Thus, we equipped our topology generator with the ability to create multilayer topologies where each layer can be a percentage (10%, 20%, 30%, 50%, 70%) larger than the previous one. So we have topologies with relatively equi-sized layers (10%), or topologies with huge layer inequalities (70%). Table 4.1 records all the independent parameters of our topology generator, their range of values, and their default values.

TABLE 4.1. Experimentation parameters values.

parameter	**range**	**default**
avg. node degree (D)	4, 6, 10, 12, 16	6
network diameter (H)	5, 10, 20, 40, 70	10
#network layers (L)	2, 3, 4, 5, 7	4
size of a layer relative to its adjacent layers	10%, 20%, 30%, 50%, 70%	-

4.5.3 Simulation results

We performed a simulation-based performance evaluation of the competing algorithms in MATLAB. We repeated each experiment 5 times, and recorded the variation in the performance, but each result was so tightly concentrated around the mean that the error bars are hardly recognizable in the plots.

4.5.3.1 Impact of topology density

Throughout this section, we consider the impact of topology density on the performance of each competitor. Firstly, in Figure 4.2 we evaluate the *per layer* size of the CDS that each competitor creates. The first observation is that the size of the CDS is almost a decreasing function with respect to the node density, which is consistent with the existing results previously obtained in [119]. That is due to the fact that the higher the network density the greater the coverage capability of the multilayer network nodes, and thus the smaller the size of the CDS. It is interesting that the distribution of the CDS nodes among the layers is almost uniform for $EMCDS$ (up to 5% variance) and for both $E2CLB$, $EUCLB$ (up to 10% variance), while it increases in each layer for $E2MLB$ (approximately from 5% up to 10%). The aforementioned

behavior has to do with the different way that each competitor creates the CDS. In $EMCDS$, each node that is selected to participate in the CDS, selects recursively its own nodes for the CDS which result to the uniform distribution of the CDS nodes on the multilayer network nodes.

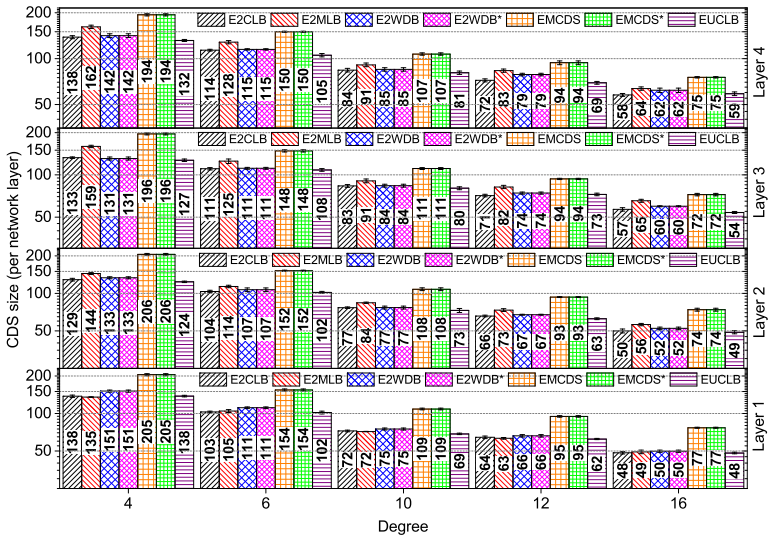

FIGURE 4.2. Impact of network density on the size of CDS.

On the other hand, both $EUCLB$ and $E2CLB$ calculate per layer the CDS. The unique behavior of $E2MLB$ is justified by the fact that it multiplexes different layers in order to calculate the $EmlPCI$ value. In $EMCDS$ the size of the CDS is from 27% (best case) up to 60% (worst case) larger than the best performing algorithm, which is $EUCLB$. The high performance of $EUCLB$ regarding the size of the CDS has been shown in [92]. The second best performing algorithm is $E2CLB$ (generally, both algorithms present almost the same performance, but in some cases $E2CLB$ presents up to 9% worse performance e.g., at layer 4 when degree = 6), third is $E2WDB$ (up to 10% worse performance)and then follows $E2MLB$ (up to 20% worse performance). Focusing on all competitors, we observe that the difference in their performance is minimum when degree = 16. This is due to fact that when nodes are relatively "close" to each other, there is significant overlapping in the selected $CDSs$. $E2WDB^*$ and $EMCDS^*$ performance is not considered here as the mediator heuristic does not affect the total number of $ECDS$ nodes in the network. Therefore these two improved algorithms present the same efficiency regarding the size of the $ECDS$ with their "clean" versions.

Next, in Figure 4.3 we evaluate the *mean per layer node* minimum relay node energy. The first observation is that compared to the previous experiment, now $EUCLB$ presents the worst performance (in most cases). That is expected as $EUCLB$ is unaware of the residual energy of each of the selected nodes for the CDS. However, we see that in some cases $EUCLB$ presents even better performance than $EMCDS$ does (e.g., when degree = 4), but this is due to the smaller CDS it creates. The second observation is that generally the competitors create *per layer node* more efficient CDS as the network density increases. This is because the larger network

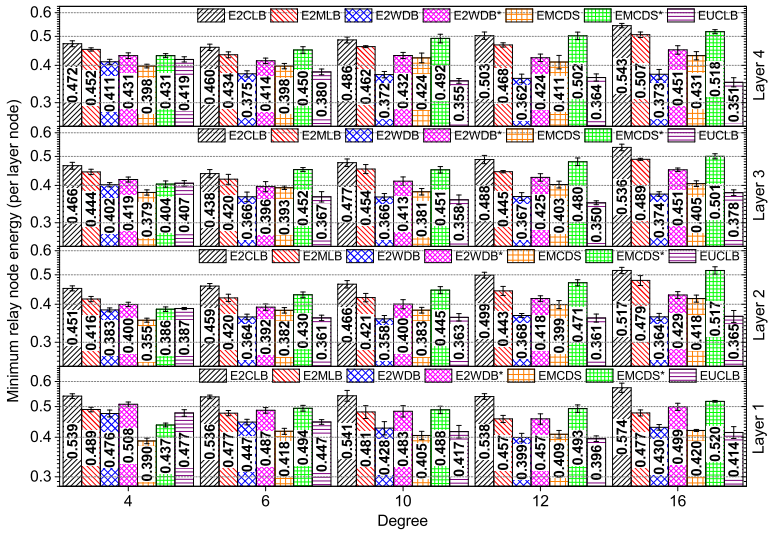

FIGURE 4.3. Impact of network density on the energy level of each relay node (on the average the worst case scenario).

density presents more opportunities for nodes with smaller residual energy level to be substituted by more energy efficient nodes. The best performing algorithms are $E2CLB$ and $E2MLB$ (we examine the performance of $E2WDB^*$ and $EMCDS^*$ right afterwards), with the first being from 4% (when considering relatively sparse networks) up to 20% (when considering relatively dense networks) more efficient than the second one. The performance gap when considering networks with different density is due to the fact that in dense networks both the pruning process and the mediator stage work more efficiently; i.e., in dense networks it is more likely to find nodes with less energy and exclude them from the CDS or reach them through other nodes which have a larger residual energy level. The third best performing algorithm is $EMCDS$ and last comes $E2WDB$. However, $E2WDB$ performs better than $EMCDS$ when degree = 4, which is justified by the fact that $EMCDS$ creates a large CDS (more than 35% larger than the CDS of $E2WDB$) and consequently many nodes with less energy are likely to participate in the CDS. This however does not exist in denser network topologies (except for the Layer 1 when degree = $6, 10, 16$). The next observation concerns the efficiency of the mediator heuristic. Both $E2WDB^*$ and $EMCDS^*$ present better performance compared to their version that lacks the heuristic. More specifically, $E2WDB^*$ is from 4% (when considering relatively sparse networks) up to 21% (when considering relatively dense networks) more efficient than $E2WDB$. For $EMCDS^*$, the corresponding figures are better compared to $EMCDS$ from 9% up to 24% (in most cases is even better than $E2MLB$ when degree > 4). The mediator heuristic is more effective in $EMCDS^*$ because it creates a relay node set with larger cardinality, thus the likelihood to be removed those nodes who participate in the CDS and have less residual energy increases.

Next, in Figure 4.4 we evaluate the *mean per layer node* relay node set cardinality. The first observation is that all competitors (except from $EMCDS$ and $EMCDS^*$ which use a different approach in order to calculate the CDS) create small *per layer node* relay sets. This is something desirable in order to reduce the number of redundant messages in broadcasting situations [116].

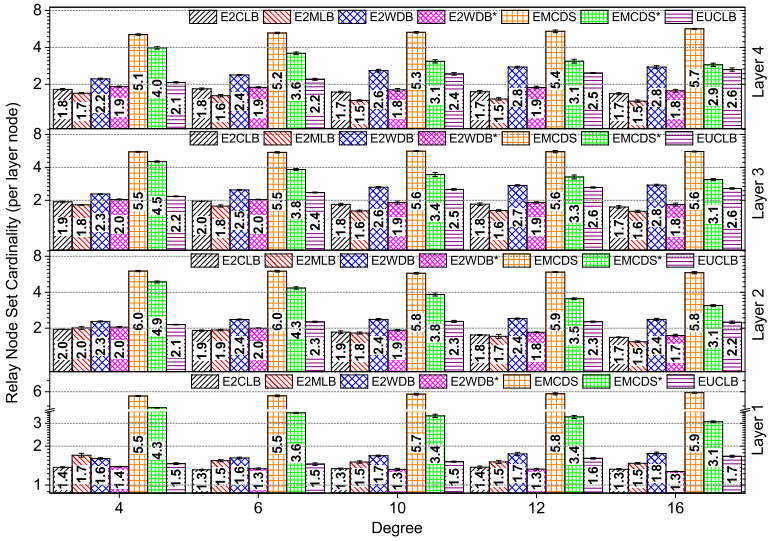

FIGURE 4.4. Impact of network density on the size of the relay node set of each network node(on the average).

The best performing algorithm is $E2MLB$ (which interestingly is presenting the larger CDS) and then follows $E2CLB$, $E2WDB^*$, $E2WDB$ and finally $EUCLB$ (which presents the smaller CDS). The second observation is that the mediator heuristic for one more time improves the efficiency of $E2WDB$ and $EMCDS$ regarding the *per layer node* relay node set cardinality (on the average) by 14% up to 55% for the $E2WD^*$ and by 28% up to 90% for the $EMCDS^*$. The higher efficiency of the mediator heuristic in $EMCDS^*$ is justified by the larger CDS that $EMCDS$ produces compared to $E2WDB$.

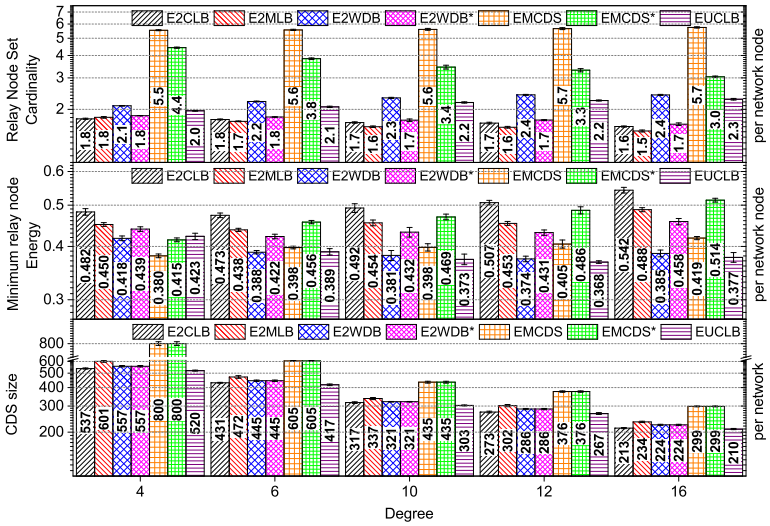

FIGURE 4.5. Impact of network density on the performance of each algorithm.

Finally, in Figure 4.5 we summarize the aforementioned *per layer* results and present them in one diagram in order to have a better overview of the impact of the topology density on the performance of each competitor. From the bottom plot we conclude that the size of the CDS decreases as the network density increases, for every algorithm. From the middle plot we

conclude that generally the CDS efficiency (in terms of the minimum energy that each CDS node has) is proportional to the network density. Finally, from the upper plot we conclude that the *per network node* size of the relay node set is irrespective to the network density.

4.5.3.2 Impact of network diameter

In this section, we consider the impact of network diameter on the performance of each competitor. Firstly, in Figure 4.6 we evaluate the *per layer* size of the CDS that each competitor creates. The first observation is that as the network diameter increases the size of the constructed CDS for all algorithms increases. The increment of the CDS is the result of sparser vicinities, i.e., fewer links between the network nodes. In other words, fewer, longer (in hops), and less distinct paths exist towards the nodes of the multilayer network, which renders the election of those nodes that compose the backbone and ensure the overall network connectivity less discrete, and hence more nodes are recruited.

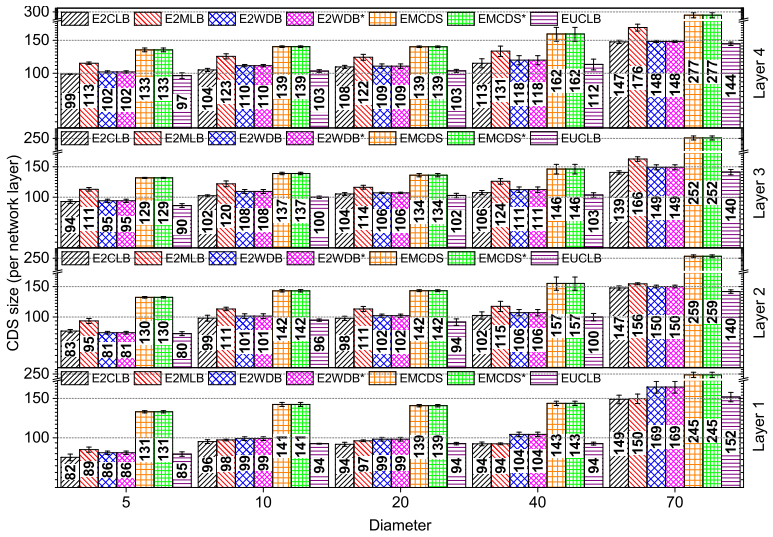

FIGURE 4.6. Impact of network diameter on the size of CDS.

It is interesting that while all the competitors manage to keep the *per layer* size of the CDS under control for the various diameter parameter settings (from 5% up to 20% CDS increase per diameter setting until when diameter = 40), they fail to do that when diameter = 70 and the *per layer* size of the CDS increases uncontrollably by approximately 60%. At that point is less prominent to find the best situated nodes in the network and therefore more nodes are selected to participate in the CDS. Focusing on the evaluation of the competitors, their performances follow the same pattern as in that of previous subsection. To elaborate, $EUCLB$ still remains the champion algorithm but now is closely followed by $E2CLB$ (or even loses by him e.g., in Layer 1 when diameter = 5 and when diameter = 70, or in Layer 3 when diameter = 70). In $MCDS$ the size of the CDS is from 35% (best case) up to 92% (worst case) larger than that of $EUCLB$. The larger *per layer* differences in the CDS size are noted when diameter = 70. The reason for this

is twofold. First, larger settings in the diameter parameter result in sparser vicinities in the network. Second, the sparser vicinities make the pruning process in $EMCDS$ less efficient when only 2-hop neighborhood information is used. As about $E2WDB$ it presents an almost equivalent performance with $E2CLB$ when diameter = 5, 10 and 20 (up to 5% worse performance)and worse performance compared to $E2CLB$ when diameter = 70 (from 10% up to 20%). Finally, the $E2MLB$ CDS is up to 18% larger than that of $E2CLB$.

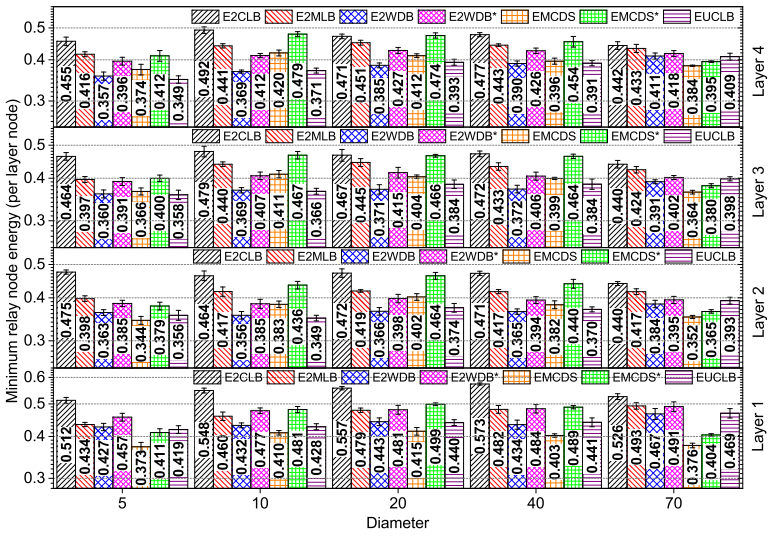

FIGURE 4.7. Impact of network diameter on the energy level of each relay node (on the average the worst case scenario).

Figure 4.7 illustrates the impact of network diameter on the *mean per layer node* minimum relay node energy. As expected, the first observation is that the generic trend is for $EUCLB$ to present the worst performance among all the competitors. Interestingly, however it is even better than $EMCDS^*$ when diameter = 70. This is due to the extremely larger CDS that $EMCDS^*$ creates compared to $EUCLB$ in conjunction with the sparser vicinities that exist in the network when diameter = 70. The best performing algorithm in this experiment is $E2CLB$. It presents comparable performance to $E2MLB$ (approximately 5% better performance) when diameter = 70, which is getting even better for smaller settings of the diameter (up to 19% better performance when diameter = 5). Definitely, $E2CLB$ can better distinguish between nodes that are situated relatively "close" to each other (smaller settings of the diameter),and select for the CDS those who have the larger residual energy. However, this positive performance gap diminishes in larger settings of the diameter, where sparser vicinities result to more nodes to be selected in the CDS. On the other hand $E2MLB$ is better than $E2WDB$ (up to 20% better performance) and $EMCDS$ (up to 20% better performance when diameter $\leq$ 40 and up to 31% better performance when diameter = 70). The worse performance of $EMCDS$ is noted when diameter = 5 and when diameter = 70. In both cases, the root of the problem is the myopic look that $EMCDS$ has that adds in the CDS many nodes with little energy in dense (diameter = 5) or sparse (diameter = 70) topologies compared to $E2WDB$. Concerning the impact of the mediator heuristic on the

performance of $E2WDB$ and $EMCDS$, it is noteworthy that it improves the mean performance of $E2WDB^*$ and $EMCDS^*$ by 10% and 15% respectively. However, this performance improvement diminishes (it drops to approximately 5% for both cases) when the network topology is getting sparse (diameter = 70). Nevertheless, $E2MLB$ presents better performance than $E2WD^\star$ (up to 8% better performance in all layers except for Layer 1 where $E2WD^*$ performance improves and gets up to 5% better than that of $E2MLB$).

Regarding the impact of network diameter on the *mean per layer node* minimum relay node set cardinality which is depicted in Figure 4.8 we notice that it is negligible. All the competitors which take into account *2-hop* connectivity info for their calculations manage to decompose efficiently the network structure (e.g., skinny or bushy shape) and continue to create small *per layer node* relay sets. Notably, this time the champion performing algorithms are both $E2CLB$ and $E2MLB$.

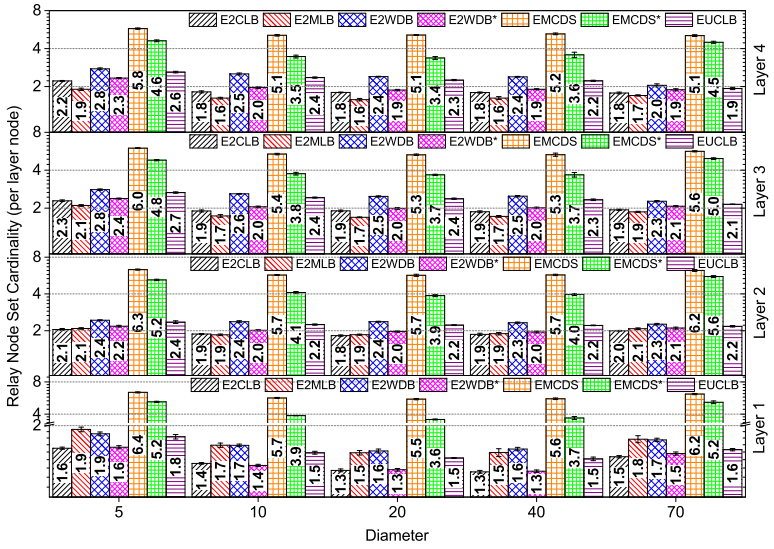

FIGURE 4.8. Impact of network diameter on the size of the relay node set of each network node(on the average).

Finally, in Figure 4.9 as a brief statement of the most important information in a piece we summarize the aforementioned *per layer* results and present them in one diagram. From the bottom plot we conclude that generally the size of the CDS increases as the diameter parameter settings increase, for every algorithm. From the middle plot we conclude that generally the algorithms efficiency (in terms of the *mean per network node* minimum relay node energy) is considered irrespective to the network diameter. Finally, from the upper plot we conclude that the *per network node* size of the relay node set is irrespective to the network diameter.

4.5.3.3 Impact of number of layers

In this section, we consider the impact of the number of network layers on the performance of each competitor. Firstly, in Figure 4.10 we evaluate the *per layer* size of the CDS that each

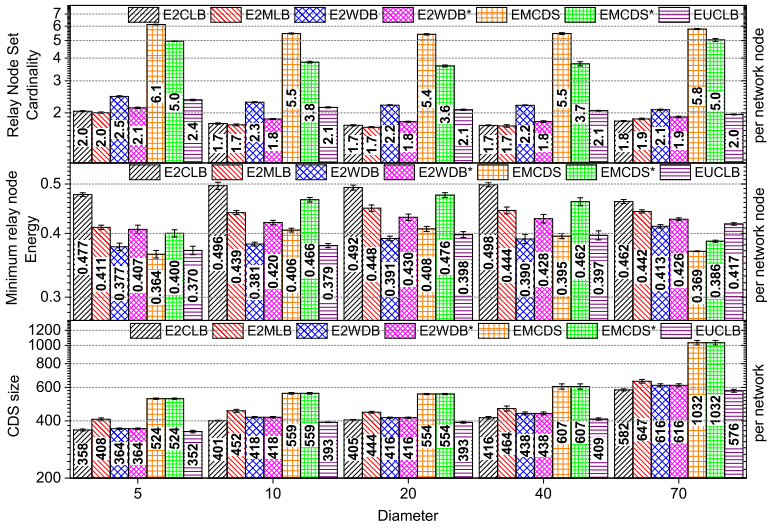

FIGURE 4.9. Impact of network diameter on the performance of each algorithm.

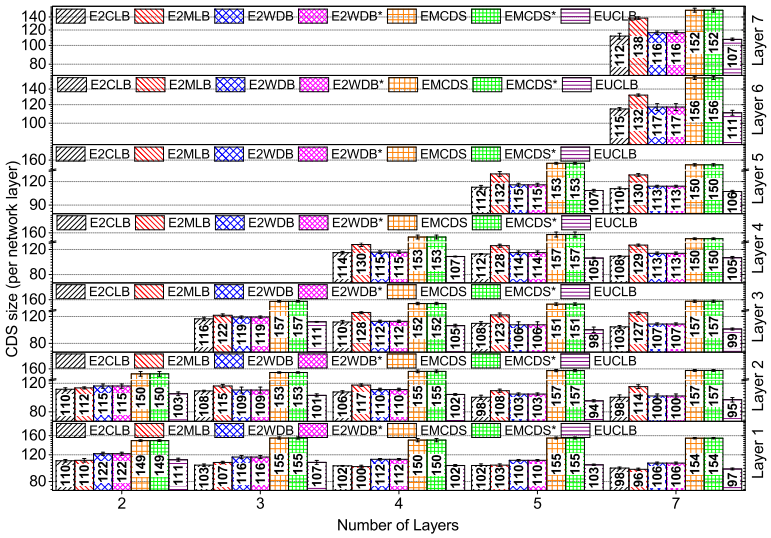

FIGURE 4.10. Impact of the number of network layers on the size of CDS.

competitor creates. First, we note that the size of the CDS is a decreasing function with respect to the number of layers. This happens because as the number of layers increases it increases the number of interlinks among layers. Thus, the coverage capability of nodes that communicate with nodes in other layers increases which result to the reduced CDS. Focusing on the evaluation of the competitors, we observe that $EUCLB$ remains the champion algorithm regarding the size of the CDS, followed by $E2CLB$ (up to 10% worse performance), by $E2WDB$ (up to 13% worse performance), by $E2MLB$ (up to 29% worse performance) and finally by $EMCDS$ (up to 71% worse performance).

Figure 4.11 illustrates the impact of the number of network layers on the *mean per layer node* minimum relay node energy. The first observation is that the *mean per layer node* minimum relay node energy is a decreasing function with respect to the number of layers (e.g., in layers

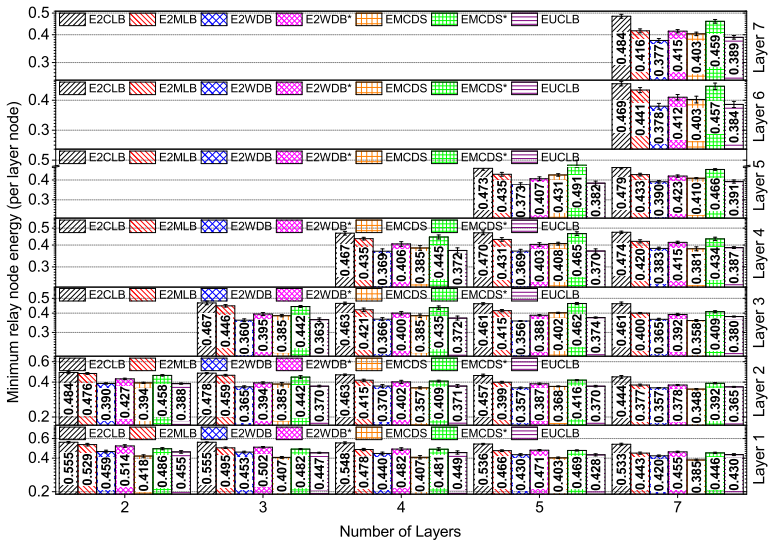

FIGURE 4.11. Impact of the number of network layers on the energy level of each relay node (on the average the worst case scenario).

1,2,3). This is justified by the reduced *per layer* size of the CDS when considering an increasing number of network layers. As the size of the CDS is reduced it is reduced the likelihood of the less energy efficient nodes to be substituted by other more energy efficient nodes. The best performing algorithm is $E2CLB$, followed by $E2MLB$ (up to 15% worse performance), next by $EUCLB$ (up to 29% worse performance), next by $E2WDB$ (up to 31% worse performance), and finally by $EMCDS$ (up to 36% worse performance). The good performance of $EUCLB$ compared to $E2WDB$ and $EMCDS$, while it is unaware of the residual energy of the network nodes is due to the fact that energy-rich nodes are centrally situated in the network. Finally, note that the mediator heuristic improves the performance of $E2WDB$ and $EMCDS$ by up to 12% and 24%, respectively.

Next, in Figure 4.12 we evaluate the *mean per layer node* relay set cardinality. The best performing algorithm is $E2CLB$ and then follows $E2MLB$, $E2WDB^*$, $E2WDB$, $EUCLB$. The mediator heuristic improves the efficiency of $E2WDB$ and $EMCDS$ regarding the *per layer node* relay node set cardinality (on the average) by 14% up to 31% for the $E2WD^*$ and by 34% up to 56% for the $EMCDS^*$.

Finally, in Figure 4.13 we summarize the aforementioned *per layer* results and present them in one diagram. From the bottom plot we conclude that the size of the CDS decreases as the number of layers increases, for every algorithm. It is straightforward that the larger the number of layers is the larger the need for more nodes to participate in the CDS becomes. From the middle plot, we conclude that generally the CDS efficiency (in terms of the *mean per network node* minimum relay node energy) is inversely proportional to the number of network layers. This is due to the selection of nodes for the CDS is driven primarily from the network topology, i.e., to establish network connectivity, and not from the energy level of each network node. The performance decrease approximately is 9% for $E2CLB$, 20% for $E2MLB$, 11% for $E2WDB$ (14%

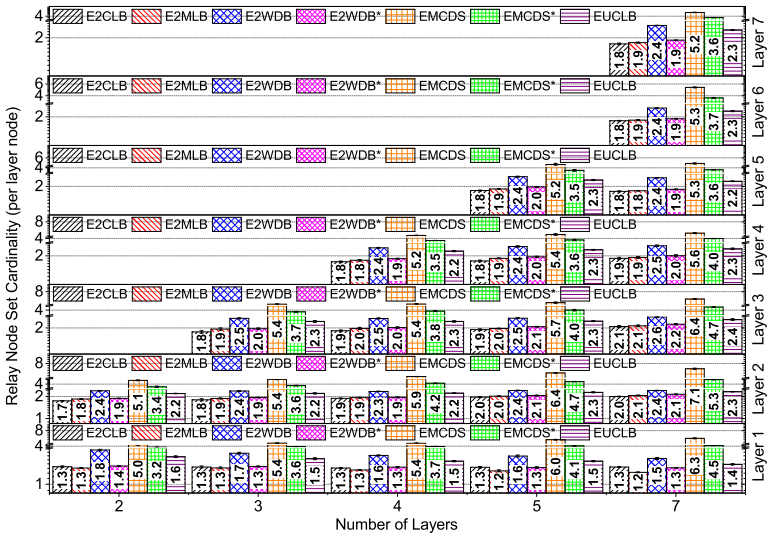

FIGURE 4.12. Impact of the number of network layers on the size of the relay node set of each network node(on the average).

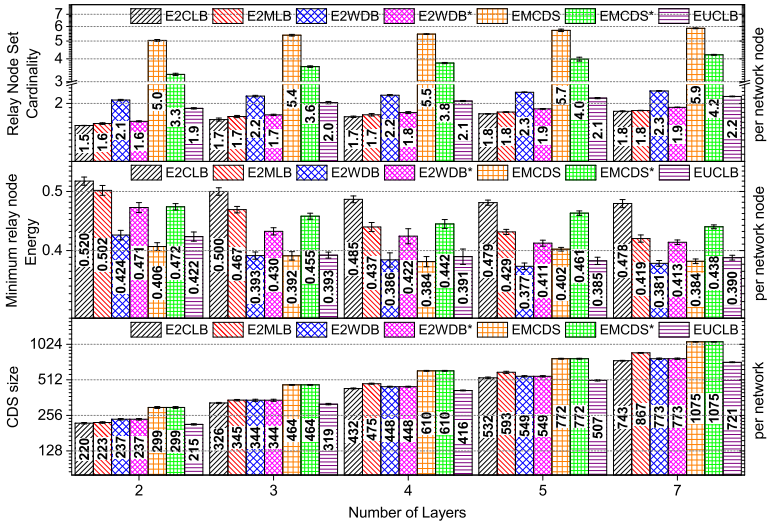

FIGURE 4.13. Impact of the number of network layers on the performance of each algorithm.

for $E2WDB^*$), 6% for $EMCDS$ (8% for $EMCDS^*$) and 8% for $EUCLB$. Finally, from the upper plot we conclude that the *per network node* size of the relay node set is irrespective to the number of network layers. Note that the mediator heuristic improves the efficiency of $E2WDB$ and $EMCDS$ regarding the *per network node* relay node set cardinality by 21% up to 31% for the $E2WD^*$ and by 40% up to 52% for the $EMCDS^*$.

4.5.3.4 Impact of increasing the layer size

In this section, we consider the impact of increasing the layer size on the performance of each competitor. Firstly, in Figure 4.14 we evaluate the *per layer* size of the CDS that each competitor

creates. Note that the size of the CDS is an increasing function with respect to the increasing layer size (except for Layer 1). This happens because as the size of each layer increases it increases the need for more nodes to act as connectors and thus for more nodes for the CDS. Focusing on the evaluation of the competitors, we observe that in the majority of cases $EUCLB$ remains the champion algorithm regarding the size of the CDS, followed by $E2CLB$ (up to 6% worse performance), by $E2WDB$ (up to 8% worse performance), by $E2MLB$ (up to 21% worse performance), and finally by $EMCDS$ (up to 59% worse performance).

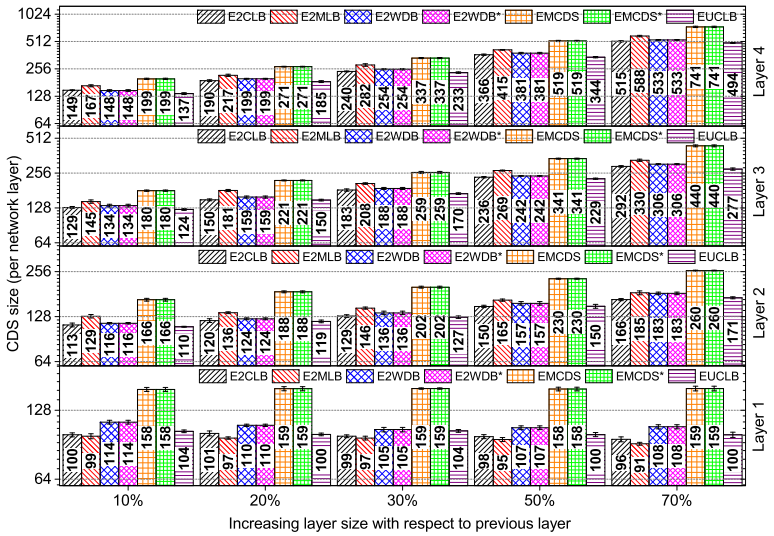

FIGURE 4.14. Impact of the network size on the size of CDS.

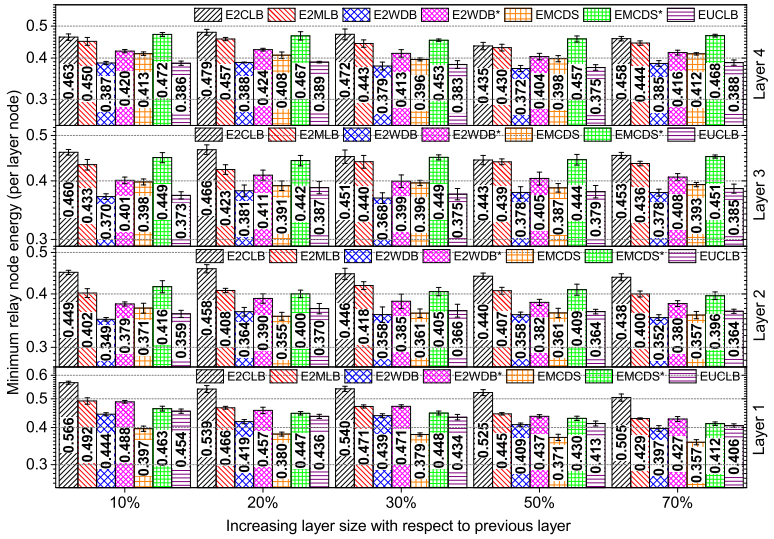

FIGURE 4.15. Impact of the network size on the energy level of each relay node (on the average the worst case scenario).

Figure 4.15 illustrates the impact of increasing the layer size on the *mean per layer node* minimum relay node energy. The first observation is that the *mean per layer node* minimum

relay node energy is irrespective to the increasing layer size. This is justified by the increased *per layer* size of the CDS when considering an increasing number of network layers. As the size of the CDS is increased it is more likely that the less energy efficient nodes to be substituted by other more energy efficient nodes. The best performing algorithm is $E2CLB$, followed by $E2MLB$ (up to 18% worse performance), next by $EUCLB$ (from 16% up to 25% worse performance), by $EMCDS$ (from 12% up to 31% worse performance), and finally by $E2WDB$ (from 19% up to 29% worse performance). Once again the weight distribution on the topology is responsible for the better performance of $EUCLB$ compared to $E2WDB$ and $EMCDS$, (energy efficient nodes are centrally situated in the network). Moreover, note that with the mediator heuristic the performance of $E2WDB^*$ and $EMCDS^*$ is improved compared to their "clean" versions by up to 10% and 18%, respectively.

Next, in Figure 4.16 we evaluate the *mean per layer node* relay node set cardinality. In this experiment both $E2CLB$ and $E2MLB$ compete for presenting the best performance (without though having a clear winner), followed by $EUCLB$ and $E2WDB$. Note that, the mediator heuristic improves the efficiency of $E2WDB^*$ and $EMCDS^*$ regarding the *per layer node* relay node set cardinality (on the average) by 5% up to 26% for the $E2WD^*$ and by 11% up to 53% for the $EMCDS^*$.

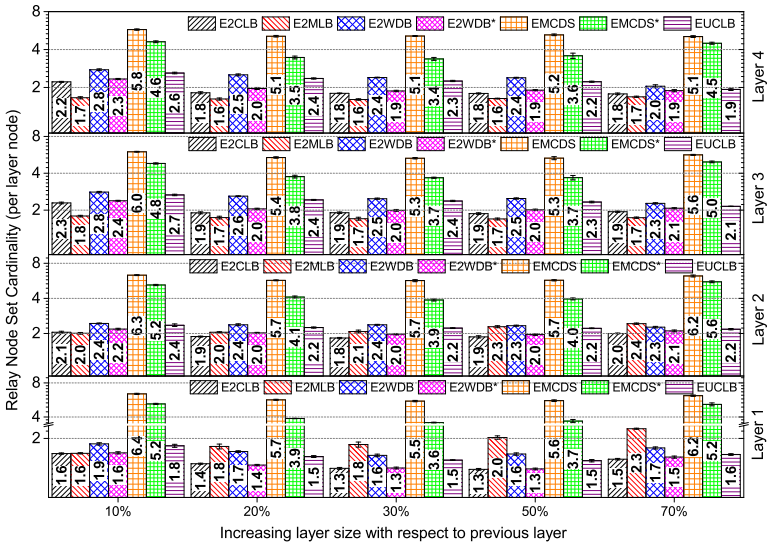

FIGURE 4.16. Impact of the network size on the size of the relay node set of each network node(on the average).

In Figure 4.17 we summarize the aforementioned *per layer* results and present them in one diagram. From the bottom plot we conclude that the size of the CDS increases with increasing (with respect to the previous layer) layer size, for every algorithm. From the middle plot we conclude that generally the algorithms' efficiency (in terms of the *mean per network node* minimum relay node energy) is considered irrespective to the increasing layer size. Finally, from the upper plot we conclude that the *per network node* size of the relay node set is irrespective to the increasing layer size.

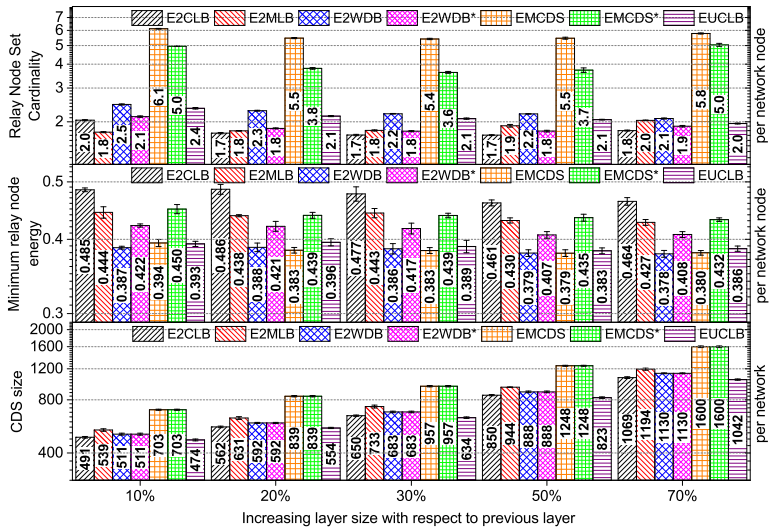

FIGURE 4.17. Impact of increasing the layer size on the performance of each algorithm.

4.5.4 Evaluation of network load

In this section we evaluate the network load on nodes included in the CDS. In each experiment, we examined the simultaneous communication among distinct pairs of nodes (randomly selected) in a dozen of topologies with the same characteristics and measured the average queue length. Here, we include a small indicative subset of the obtained results. In particular, we show the results which concern the simultaneous communication between 200 pairs of nodes, and record the queue length of each CDS node. The overall conclusion is that all queues remain bounded, and in particular only a couple of node queues reach a size of maximum eleven messages.

4.5.4.1 Network load in sparse networks

Figure 4.18 illustrates the network load on the CDS nodes when considering sparsely connected multi layer networks. Namely, we utilized networks consisting of 4 equi-sized layers, with a total number of nodes equal to 2000 and layer diameter equal to 70. The generic observation is that none of the competing methods presents any likely-overflow buffer phenomenon. Moreover, the $E2CLB$ algorithm manages to have the largest number of nodes with the least number of messages, i.e., around 800 nodes whose queue accommodates on the average one message.

4.5.4.2 Network load in dense networks

Next, in Firure 4.19 we examine the network load on dense multilayer networks. The setting is same as previously, but now with a diameter of each layer equal to around 50. We observe the same pattern of performance as in the previous experiment, and we see – for all competitors – fewer nodes with larger queues which is to be expected since in dense networks more paths are available to serve traffic.

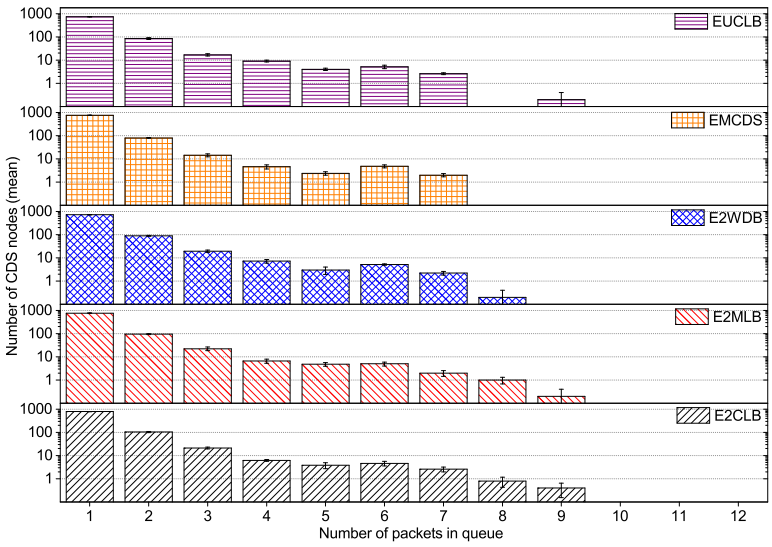

FIGURE 4.18. Network load in sparse networks.

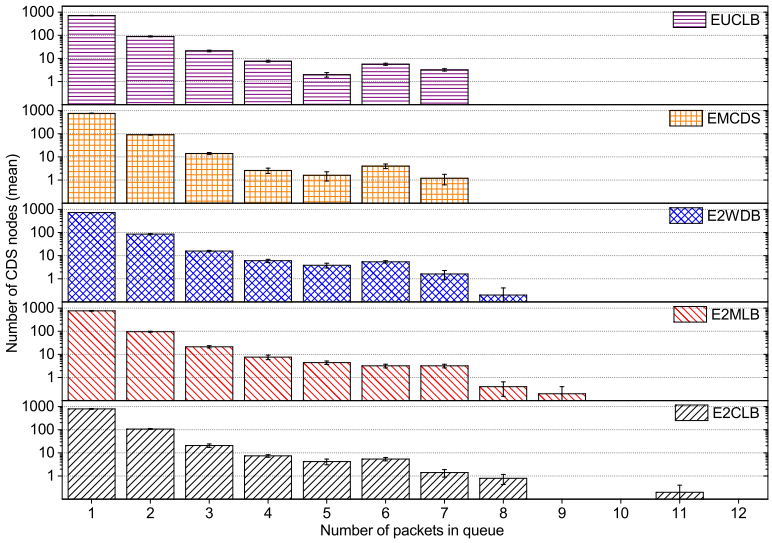

FIGURE 4.19. Network load in dense networks.

4.5.4.3 Network load in networks with more layers

Next, in Firure 4.20 we examine the load on networks with more layers, namely with 7 equi-sized layers (now the number of nodes is 3500). The results are alike the previous experiment, since now the communicating pairs are spread more sparsely among the set of nodes.

4.5.4.4 Network load in networks with non equi-sized layers

Finally, in Figure 4.21 we examine the network load on when the layers differ in their size. The setting is as the original one, but we have networks with 4 layers, and the adjacent layers differ in the number of nodes by 20%, (the total number of nodes is 2184). The difference in queue lengths is that we observe an increase in the number of nodes with moderate queue size, because some CDS nodes which belong to layers with less nodes are selected for message routing.

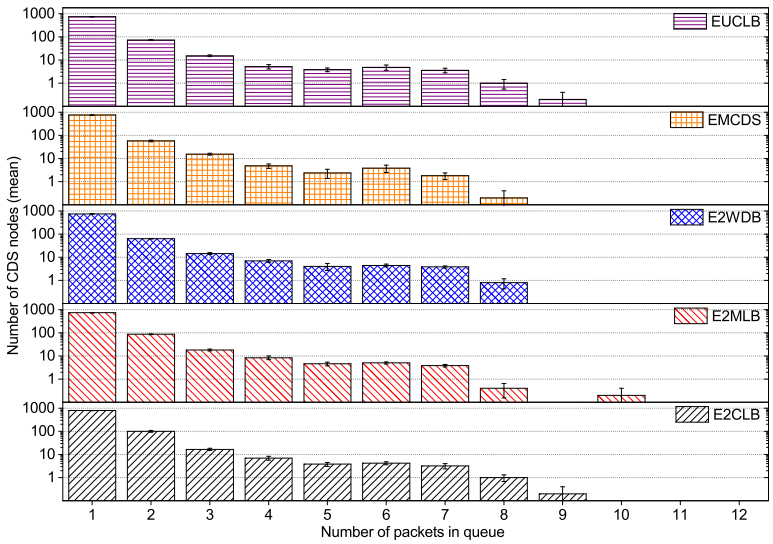

FIGURE 4.20. Network load in networks with more layers.

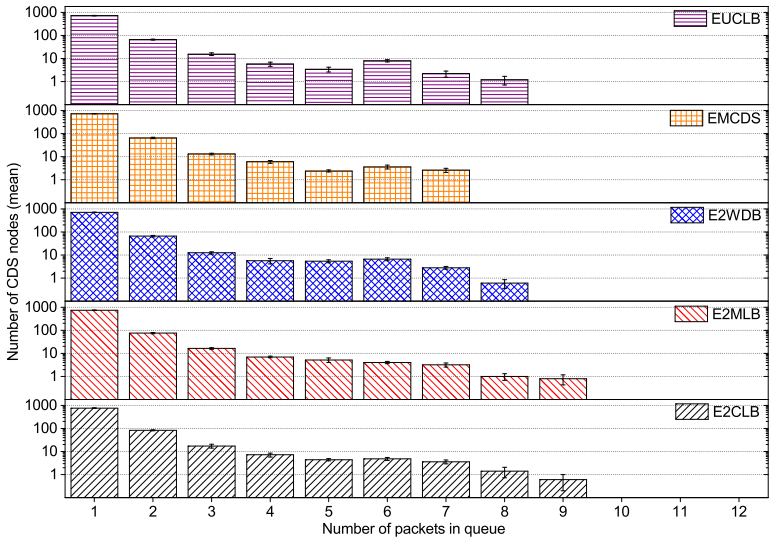

FIGURE 4.21. Network load in networks with unequal layer sizes.

4.5.5 Results with skewness-varying topologies

Here we evaluate the efficiency of the competing algorithms when considering networks whose topologies and weight skewness settings varying across a range of settings. Each multilayer network is composed by 4 layers, each one of them containing 500 nodes (mean degree = 6). In Table 4.2 we present for the three different settings of the topology skewness (Low, Medium, and High) the values of the respective parameters of interest.

In Figures 4.22–4.24 the results concern the case where the skewness is towards high degree nodes, and in Figures 4.25–4.27 the results concern the case where the skewness is towards low degree nodes. The generic observation is that the performance differences between competitors

remain almost stable regardless of the topology and weight skewness. This justifies the fairness of the settings we used in our earlier experimentation.

TABLE 4.2. Experimentation parameters values.

Topology Skewness	s_{degree}	s_{layer}	s_{node}
Low	0.1	0.1	0.1
Medium	0.5	0.5	0.5
High	0.9	0.9	0.9

4.5.5.1 Skewness to high degree nodes

In Figure 4.22 we evaluate the impact of the various settings of the topology and weight skewness on the performance of the competing algorithms regarding the size of the CDS when skewness is towards high degree nodes. The first observation is that the size of the CDS increases for larger settings of the topology skewness with respect to the same weight skewness setting. That is something we expected to happen as larger settings of the topology skewness result to non uniform distribution of the interlinks among the mlNetwork layers, the appearance of some hub nodes in the mlNetwork and consequently drives to more *per layer* nodes selected for the CDS in order to guarantee the network connectivity.

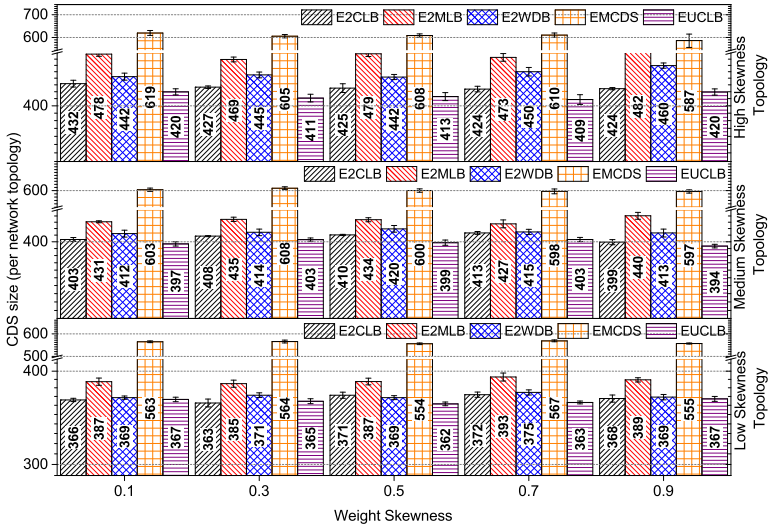

FIGURE 4.22. Algorithms performance (CDS size) with skewness to high degree nodes.

The second observation concerns the impact of the weight skewness on the size of the CDS with respect to the same topology skewness settings and should be considered in combination with the respective results of Figure 4.23. To elaborate, note that the weight skewness has negligible impact on the size of the CDS for the same topology skewness settings. That is happening because the algorithms decide about the CDS primarily based on the existing network topology (establish network connectivity). The residual energy is taken into account only when there is some coverage

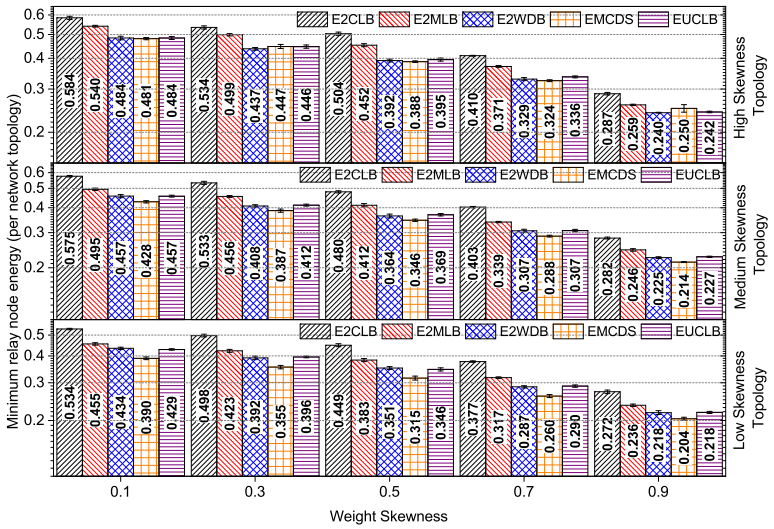

FIGURE 4.23. Algorithms performance (min energy) with skewness to high degree nodes.

redundancy between the mlNetwork nodes(establish network connectivity first and then strive to substitute the less energy efficient nodes). This observation justifies the case in Figure 4.23 where the *mean per network node* minimum relay node energy decreases for larger settings of the weight skewness as larger settings of the weight skewness result to less uniform distribution of the weights in the mlNetwork and consequently to the selection of some less energy efficient nodes in the CDS.

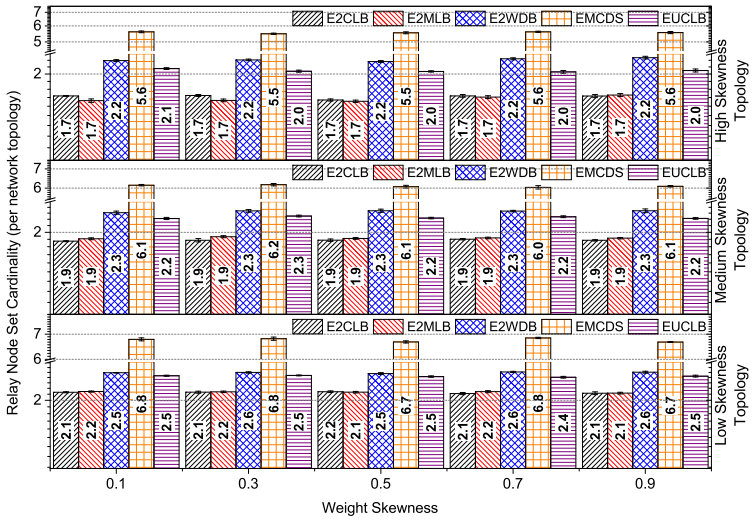

FIGURE 4.24. Algorithms performance (relay node set cardinality) with skewness to high degree nodes.

Finally, in Figure 4.24 we observe that the weight skewness has negligible impact on the *mean per network node* size of the relay node set which is justified by the fact that each algorithm strives for the minimum possible *per network node* relay node set as this guarantees smaller volume of broadcast message transmissions in the network. We observe also that for

larger settings of the topology skewness the *mean per network node* cardinality of the relay node set decreases which is justified by the larger CDS with these settings and thus the greater coverage capability of the selected relay nodes.

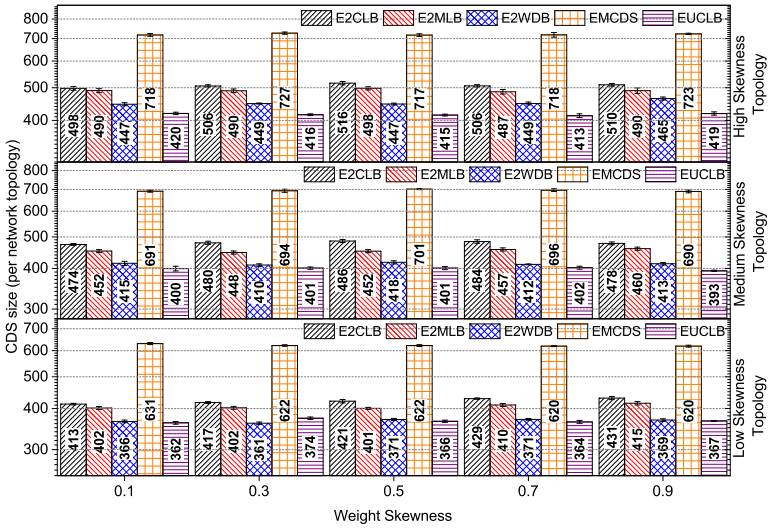

FIGURE 4.25. Algorithms performance (CDS size) with skewness to low degree nodes.

4.5.5.2 Skewness to low degree nodes

In Figure 4.25 we evaluate the impact of the various settings of the topology and weight skewness on the performance of the competing algorithms regarding the CDS when the skewness is towards low degree nodes. The observations of Figure 4.22 regarding the size of the CDS when using larger settings of the topology skewness still apply. Nevertheless, the performance of both *E2CLB* and *E2MLB* worsens compared to the respective performance of *EUCLB* and *E2WDB*. That is because of the attributes of the mlNetwork; i.e low degree nodes take priority over high degree nodes in getting the interlinks which makes them good choices for the CDS. However, the reduced coverage of the low degree nodes in combination with the residual energy of the participating nodes in the CDS which we should take into consideration explains the larger CDS of *EUCLB* and *E2WDB*. All in all, that is acceptable to happen as long as both *E2CLB* and *E2MLB* create energy efficient CDSs.

In Figure 4.26 we evaluate the impact of the various settings of the topology and weight skewness on the performance of the competing algorithms regarding the *mean per network node* minimum relay node energy when the skewness is towards low degree nodes. The observations of Figure 4.23 still apply; i.e the *mean per network node* minimum relay node energy decreases for larger settings of the weight skewness. Moreover, the relative performance among competing algorithms compared to when the skewness is towards high degree nodes still apply except for *E2WDB* which presents worse performance by *EMCDS*.

Finally, in Figure 4.27 we observe that the weight skewness has negligible impact on the

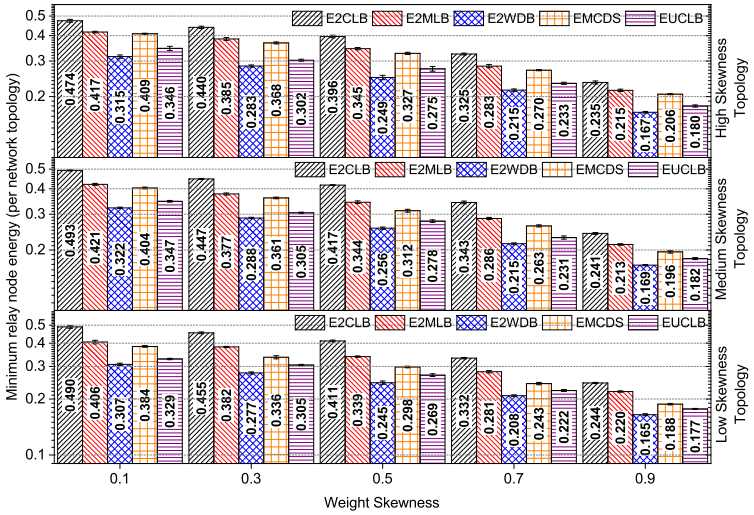

FIGURE 4.26. Algorithms performance (min energy) with skewness to low degree nodes.

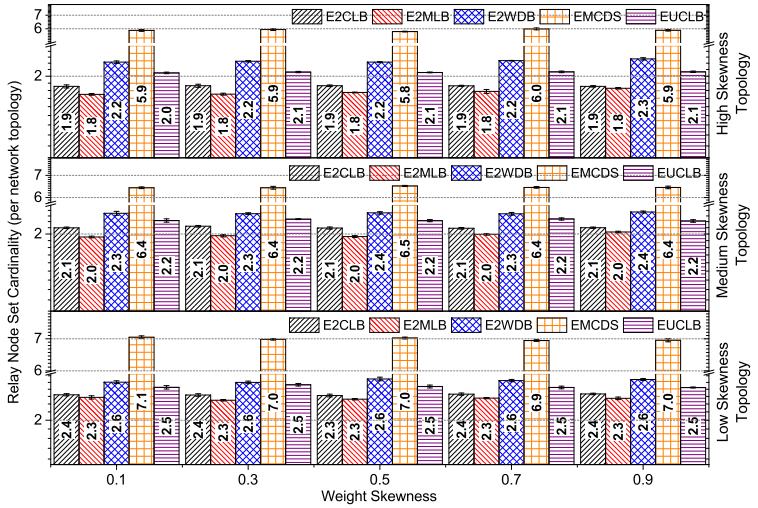

FIGURE 4.27. Algorithms performance (relay node set cardinality) with skewness to Low degree nodes.

mean per network node size of the relay node set. We observe also that for larger settings of the topology skewness the *mean per network node* cardinality of the relay node set decreases which is justified by the larger CDS with these settings; i.e the larger CDS is a by product of larger relay node sets which results in increased likelihood that a less efficient relay node to be substituted by an energy efficient relay node.

4.5.6 Pruning Rule k efficiency

In this section we evaluate the efficiency on using more connectivity information in reducing the size of CDS during the pruning phase.

4.5.6.1 Impact of topology density

The results presented in Figure 4.28 study the impact of increasing node degree on the performance measures when using 2-hop information, and Figure 4.29 when using 3-hop information for all algorithms but *EMCDS*. The results are intuitive and confirm the findings of the main chapter. Denser connectivity (higher average degree) means smaller CDS, equal or larger relay node sets per node. Utilizing more information, i.e., 3-hop information can decrease these quantities by a factor of 2 or 3. Champions algorithms are as before.

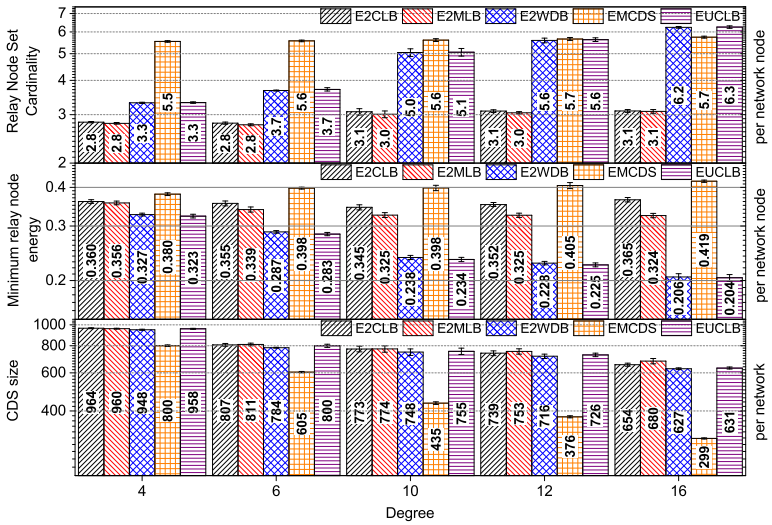

FIGURE 4.28. Impact of network density on the performance of each algorithm by using 2-hop neighborhood information.

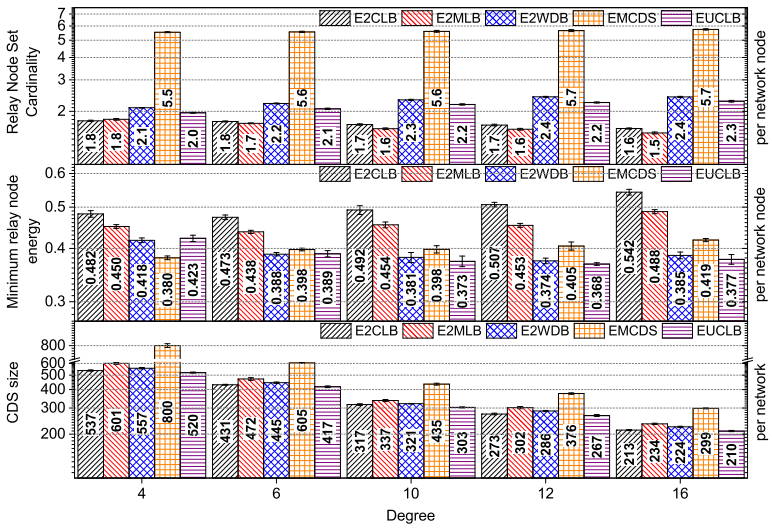

FIGURE 4.29. Impact of network density on the performance of each algorithm when using 2-hop neighborhood information for *EMCDS* and 3-hop neighborhood information for the rest.

4.5.6.2 Impact of network diameter

The results shown in Figure 4.30 investigate the impact of increasing diameter on the performance of the competitors when exploiting 2-hop information or 3-hop information (Figure 4.31). The performance patterns are similar to those reported in the previous pair of graphs. Using such rich imformation every algorithm can improve its performance concerning CDS size 3 times from small and medium diameter valus, and 2 times for larger diameter values.

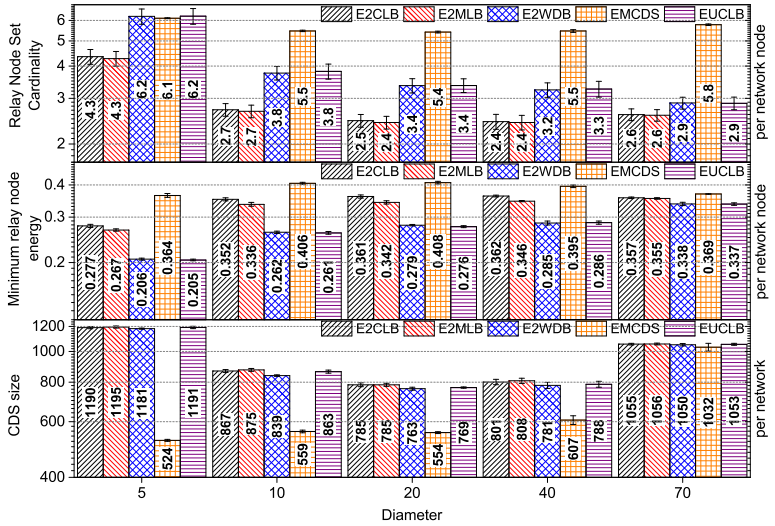

FIGURE 4.30. Impact of network diameter on the performance of each algorithm when using 2-hop neighborhood information.

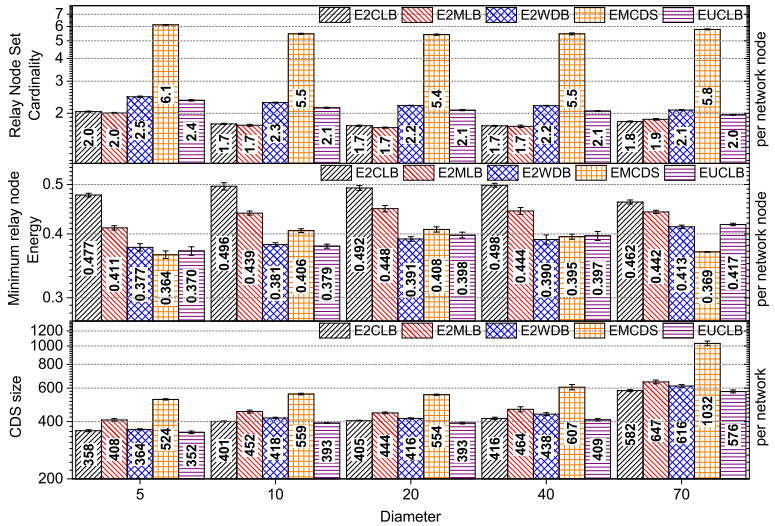

FIGURE 4.31. Impact of network diameter on the performance of each algorithm when using 2-hop neighborhood information for *EMCDS* and 3-hop neighborhood information for the rest.

4.5.6.3 Impact of number of layers

The results shown in Figure 4.32 investigate the impact of increasing the number of network layers on the performance of the competitors when exploiting 2-hop or 3-hop information (Figure 4.33). Here the performance gains are smaller and every competitor improves itself at a factor of 2 concerning CDS size, and at a factor of 1.5 concerning relay set size and residual energy.

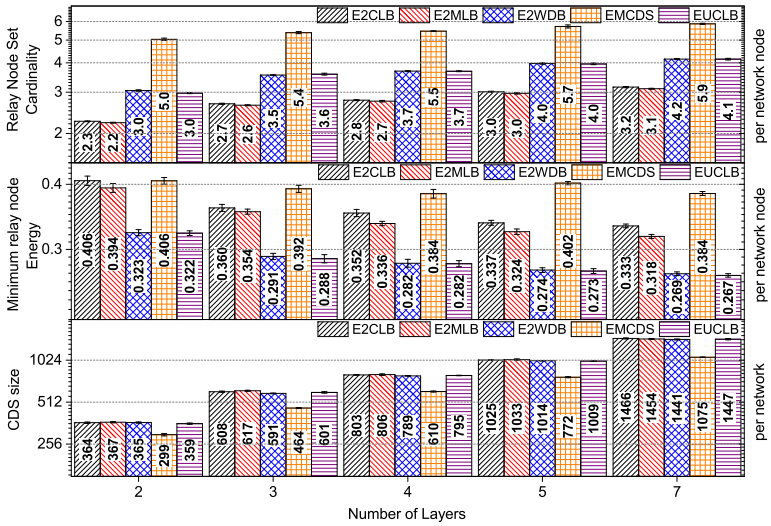

FIGURE 4.32. Impact of the number of network layers on the performance of each algorithm when using 2-hop neighborhood information.

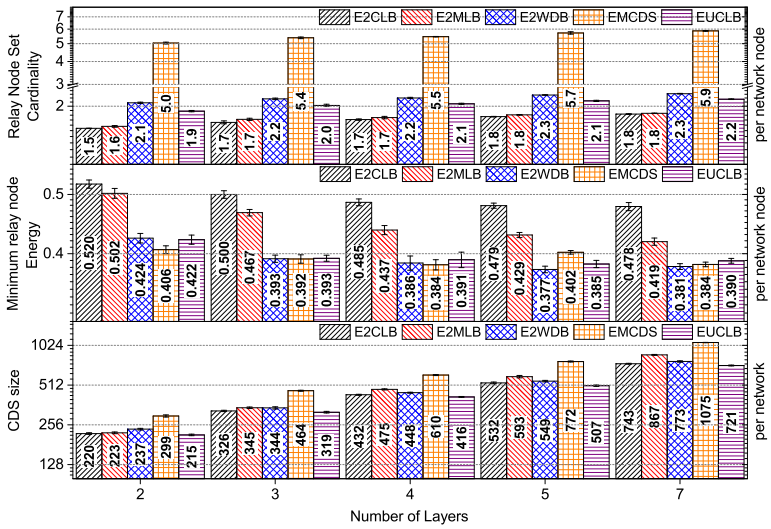

FIGURE 4.33. Impact of the number of network layers on the performance of each algorithm when using 2-hop neighborhood information for *EMCDS* and 3-hop neighborhood information for the rest.

4.5.6.4 Impact of increasing layer size

The results shown in Figure 4.34 investigate the impact of increasing the number of network layers on the performance of the competitors when exploiting 2-hop information or 3-hop information (Figure 4.35). The results are alike those observed in the previous pair of plots.

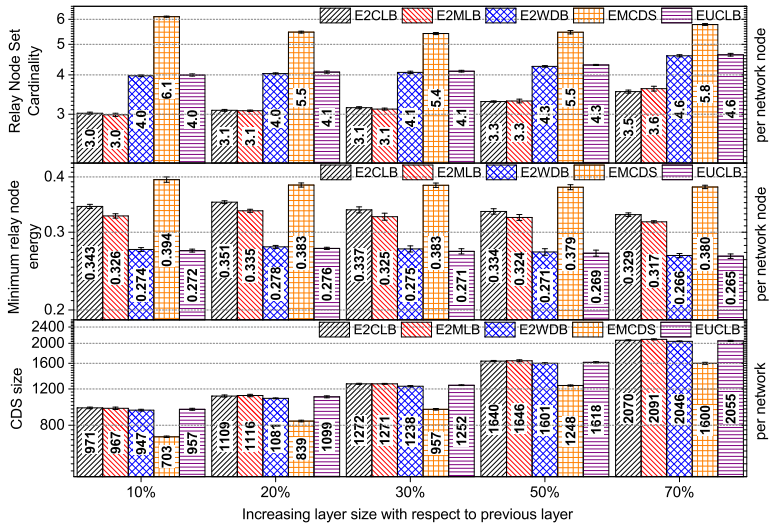

FIGURE 4.34. Impact of increasing the layer size on the performance of each algorithm when using 2-hop neighborhood information.

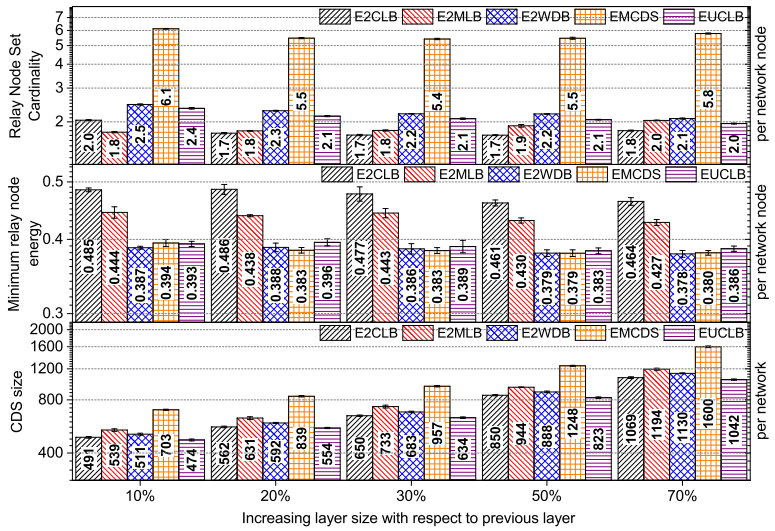

FIGURE 4.35. Impact of increasing the layer size on the performance of each algorithm when using 2-hop neighborhood information for *EMCDS* and 3-hop neighborhood information for the rest.

4.6 Conclusion

Multilayer wireless ad hoc networks arise in several modern settings and pose some new challenges around effective and efficient communication capability among their entities. This chapter investigates the problem of constructing energy-efficient backbones for such network types utilizing the notion of connected dominating sets (CDS). Improving on from our earlier work, which proved the insufficiency of traditional algorithms for addressing this problem, we proposed a new centrality measure, namely $EclPCI$ which can identify energy-rich and at the same time "central" to the topology nodes. Then, the chapter developed a distributed algorithm, namely $E2CLB$ for calculating an energy-efficient connected dominating set based on the proposed centrality measure.

The proposed algorithm was evaluated analytically by establishing its computational and communication complexity, and experimentally in an exhaustive manner. The experimental evaluation was done with respect to independent parameters that quantify the structure of the topology, i.e., density and shape (diameter), the size of the multilayer network in terms of the number of nodes and layer. The performance measures quantified the overall (and per layer) size of the dominating set, and the residual energy. Even though there is no prior related work on this subject, we employ as competitors six other algorithms; some of them stem from the present work and others are straightforward extensions of traditional well-known algorithms. In all experiments, the proposed $E2CLB$ proved to be the winning algorithm in the sense that it could trade a very small increase or no increase at all at the dominating set size in order to offer significant gains in terms of residual energy of the CDS nodes. Interesting extensions of the present work are the investigation of this problem for unidirectional connectivity or the incremental maintenance of the CDS in cases of topology changes.

CHAPTER 5

DISTRIBUTED ALGORITHMS FOR MULTILAYER CONNECTED EDGE DOMINATING SETS

5.1 Introduction

In this chapter we investigate the problem of distributed computation of a resilient network overlay for communication link monitoring in the context of multilayer ad hoc networks. The distributed nature of modern networks and the limited processing power of networked sensors and embedded systems used in Internet of Things (IoT) applications has led to new security vulnerabilities [14]. Intruders can inject malicious communications between any two networked elements without aiming to have the message propagated to any further target. The increasing variety, capability, and complexity of network elements has increased this risk. Furthermore, the evolution of these networks leaves them vulnerable to errors and compatibility issues when new elements are added to the network. While these issues may be sensed by the communicating network elements, their limitations do not allow them to compute remedies, necessitating communication to elements with more processing power. In this work, we present a framework for monitoring network failures using connected edge dominating sets in multilayer networks, and then we provide efficient distributed algorithms for their computation.

Related publication **[J1]:** Dimitrios Papakostas, Soheil Eshghi, Dimitrios Katsaros, Leandros Tassiulas. "*Distributed Algorithms for Multilayer Connected Edge Dominating Sets*", **IEEE Control Systems Letters**, vol.3, pp.31-36, January, 2019.

Related publication **[C1]:** Dimitrios Papakostas, Soheil Eshghi, Dimitrios Katsaros, Leandros Tassiulas. "*Distributed Algorithms for Multilayer Connected Edge Dominating Sets*", **Proceedings of the 57th IEEE Conference on Decision and Control (CDC)**, Miami Beach, FL, USA, December 17-19, 2018.

Consider the case where we wish to be able to monitor all the communication taking place among nodes of a wireless ad hoc network such as the one shown in Figure 5.1. It is assumed that any pair of nodes can initiate an exchange of packets and the routing may follow any path of the network, e.g., not only the shortest-path route between the communicating nodes. In principle, this task requires us to recognize a set of edges (communication links) such that every other edge is adjacent to at least one edge belonging to this set; then, by placing monitoring devices at the endpoints of each edge belonging to this set we can achieve our goal. Such a set of edges is termed an Edge Dominating Set (EDS) in graph-theoretic terms. Due to cost considerations, we are interested in identifying such sets with minimum cardinality, i.e., we seek Minimum Edge Dominating Set (MEDS). However, as it is often the case for ad hoc networks, the set of monitoring devices must be able to output any intercepted information; therefore the $MEDS$ must be *connected* [Minimum Connected Edge Dominating Set (MCEDS)], and, moreover, must be computed in a distributed fashion. Looking at Figure 5.1, we can confirm that the set of blue edges constitutes a $MCEDS$, and also the set of green edges constitutes a $MCEDS$.

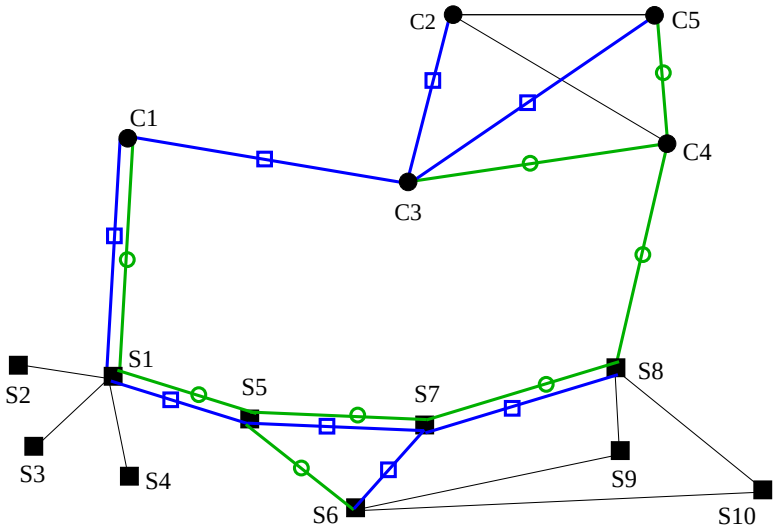

FIGURE 5.1. Two minimum connected edge dominating sets: the blue (with square marks) includes one inter-layer edge, and the green (with circular marks) includes two inter-layer edges.

The concept of *network layers* can capture the diversity in the capabilities of network elements, as well as their differing roles. For example, although traditional ad hoc networks are treated as single layer networks, military tactical ad hoc networks [92] are considered to be *multilayer* networks due to the existence of different types of units (infantry, vehicles or airborne units), where nodes belong to different layers, i.e., groups. For instance, in Figure 5.1 the node set $C1$–$C5$ comprise one layer and nodes $S1$–$S10$ comprise another layer.

Finding an $MCEDS$ for multilayer networks is somewhat more complicated than calculating $MCEDS$ for single layer networks, both for technical reasons (cf. Theorem 5.2), and for

application-specific reasons, e.g., robustness. Looking again at the blue and green $MCEDS$'s in Figure 5.1, we observe that the green one includes two edges that connect the different layers (inter-layer edges), whereas the blue only has one such edge. Increasing the number of inter-layer edges can improve the network's resiliency to failures in any particular layer [92].

In this chapter, we cast our monitoring problem for multilayer networks, which entailed finding an $MCEDS$ in a distributed manner containing many inter-layer edges, into a new form of generic domination problems. We name this problem the Multi-Colored MCEDS (MCMCEDS) problem, and we will describe it here in terms of calculating the minimum multi-colored edge dominating set. The framework and the algorithms proposed can be used for efficiently detecting and avoiding interference conditions in large wireless IoT networks, or even in more specialized setting such as those enabling $DynamicFrequencySelection(DFS)$ where radar signals must be detected and protected against interference from 5GHz radios; DSs concepts have been used in the past for monitoring problems [54, 84].

The contributions of the present chapter are as follows:

- It introduces the novel problem of finding a $MCEDS$ in multilayer networks with the additional goal of including many inter-layer links into the EDS (§5.2). This problem extends ideas related to those developed in [113].
- It analyzes its computational complexity (§5.3).
- It proposes three heuristic distributed algorithms for it (§5.4).
- It proves an analytic result that relates the cardinality of an Independent Edge Dominating Set (IEDS) to the cardinality of a corresponding Connected Edge Dominating Set (CEDS) (§5.4.2).
- It conducts a performance evaluation of the proposed algorithms against two baseline competitors (§5.5).

We define the $MCMCEDS$ problem in §5.2. We then present results on the complexity of $MCMCEDS$ computation in §5.3, and discuss our approaches to computing heuristics and their rationale §5.4. We present extensive simulation results in §5.5. We survey related work on §5.6 and finally, in Section 5.7 we conclude the present work.

5.2 The MCMCEDS problem

5.2.1 Edge domination in traditional settings

Firstly, we will provide some basic definitions on dominating sets [54] before we formulate this chapter's problem.

Definition 5.1. *An $EDS(G)$ of a network (G,E) (G is the set of nodes, and E is the set of edges) is any subset of E such that any edge $e \in E$ is either a member of $EDS(G)$ (it is a dominating edge) or it has one common endpoint with at least one dominating edge (it is a dominated edge).*

Let x_e be an indicator variable representing whether $e \in E$ is included in $EDS(G)$. Therefore, Definition 5.1 is equivalent to saying that for each $e \in E$: $x_e + \sum_{e' \in N(e)} x_{e'} \geq 1$, where $N(e)$ is the set of neighboring edges of edge e (i.e., those with one common endpoint). Note that in the *line-graph* $L(G)$ of graph G, in which every edge is replaced with a vertex and vice versa, and the incidence relationship between edges and vertices is preserved [25], an edge dominating set in G, $EDS(G)$, is translated to a $DS(G)$.

Definition 5.2. *An $IEDS(G)$ of a network (G,E) (also referred to as a maximal matching [123]) is any EDS of G such that no two edge dominators share an endpoint.*

Definition 5.3. *A $CEDS(G)$ of a network (G,E) is any EDS of G such that the set of dominating edges along with their endpoints comprise a connected network.*

The line-graph ($L(G)$) preserves connectivity [25], so in translation, $CEDS(G)$ becomes a $CDS(G)$ of the line-graph.

Definition 5.4. *A $MCEDS(G)$ of a network (G,E) is any $CEDS$ of G with the additional property that it contains the least possible number of dominating edges.*

At this stage, the link between the $CEDS$ and its equivalent in the line-graph is broken: an $MCEDS(G)$ will translate to a CDS with the minimum number of nodes, and not edges, in the line-graph. Therefore, the problems of finding the $MCDS$ [134] and the $MCEDS$ are not linked in a straightforward manner. So an $MCDS$ in $L(G)$ will be a $CEDS$ in G, but there is no guarantee that its cardinality will be minimal.

5.2.2 Edge domination in multi-layered network settings

Definition 5.5. *A multilayer network comprised of n layers is a pair (G^{ML},E^{ML}), where $G^{ML} = \{G^i, i = 1,\ldots,n\}$ is a set of networks (G_i,E_i), as defined earlier, and $E^{ML} = \{E_{i,j} \subseteq G_i \times G_j; i,j \in \{1,\ldots,n\}, i \neq j\}$ is a set of inter-layer edges.*

Definition 5.6. *A $MCEDS$ of a multi-layered network $MCEDS(G^{ML})$ includes the minimum set of edges such that their induced subgraph is connected and edges not in this set are adjacent to at least one edge within it.*

In Figure 5.1, $G_1 = \{Si, i = 1,\ldots,10\}$, $G_2 = \{Ci, i = 1,\ldots,5\}$, and E^{ML} is the set of all edges connecting them e.g., $\langle S1, C1\rangle$.

Definition 5.7. *An edge-multicolored multi-layer network (see Figure 5.2) is a multi-layer network with these two properties:*

p-1) all edges e whose endpoints both belong to a single (any) layer, i.e., $e \in E_i, \forall i \in \{1,\ldots,n\}$ *have the same color (black). E.g., black edges in Figure 5.2.*

p-2) all edges l whose endpoints belong to different layers i.e., $l \in E_{i,j} \subseteq G_i \times G_j, i,j \in \{1,\ldots,n\}, i \neq j$ *will have the same color, which is different from the color of edges* $c \in E_{x,y} \subseteq G_x \times G_y, x,y \in \{1,\ldots,n\}, [x,y] \neq [i,j]$*. E.g., red edges in Figure 5.2.*

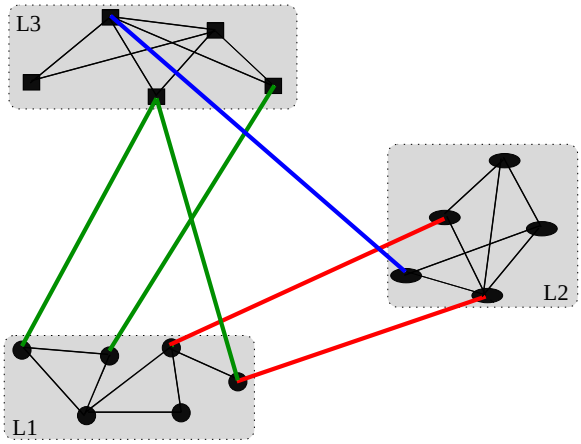

FIGURE 5.2. A multicolored multi-layer network with 3 layers (L1, L2, L3).

Definition 5.8. *A MCMCEDS of a multilayer network* $MCMCEDS(G^{ML})$ *is an* $MCEDS(G^{ML})$ *with the maximum number of colorful (i.e., non-black) edges.*

Problem 5.1 (dist-$MCMCEDS$)**.** *We seek to find an* $MCMCEDS(G^{ML})$ *for a multi-layer network* G^{ML} *in a distributed fashion, i.e., having only knowledge of the k-hop neighborhood around each node. Here, we set* $k = 2$*.*

5.3 Complexity of the MCMCEDS problem

Theorem 5.1. *The MCMCEDS problem is NP-hard.*

Proof. Assume we have a single-layer graph $G = (V,E)$ and we seek to find its $MCEDS$. Now, create a 2-layer network (G^{ML}, E^{ML}), where $G^{ML} = \{G^i, i = 1,2\}$ have the same vertices as V, and a set of inter-layer edges $E^{ML} = \{E_{i,j} \subseteq G_i \times G_j; i,j \in \{1,\ldots,n\}, i \neq j\}$ by assigning one edge in E to E_{ML} and the rest to E_1, and E_2 uniformly at random. If $MCMCEDS$ for such a (G^{ML}, E^{ML}) was not NP-hard, we could use it at most $|E|$ times (varying the edge assigned to E_{ML}) to find a solution to $MCEDS$, a known NP-complete problem [87, p. 102, Lemma 4.4.3]. ⊣

5.4 Heuristics for the MCMCEDS problem

Since our problem is NP-hard, we wish to design heuristic algorithms that can encapsulate the idea of including as many inter-layer edges as possible into the EDS. In our previous work [12, 92] we have introduced the family of the PCI centrality measures for multilayer networks, namely $mlPCI$ and $clPCI$, whose purpose is to assign a value to each node which depicts its connectivity both to its layer and to other layers. In [92] we used $clPCI$ and $mlPCI$ for the purpose of establishing a VBN for multilayer ad hoc networks based on the calculation of a node dominating set. Note that a simple application of these algorithms to create a CDS is insufficient, as it may leave some edges "undominated". We will not repeat the definitions, but instead give the distributed algorithms for the calculation of the edge dominating sets, and calculate their computational complexities as a function of Δ, the maximum node degree in the network.

Algorithm 5.1: IEDS

postcondition : Completed $IEDS$ election process

remarks : multilayer network $G = (V, E)$ where V and E are vertex & edge set, $M(u)$ / $M(w_{i,j}^{edge})$: True(T) / False(F) indicator for node u / edge $w_{i,j}^{edge}$ being a DS node / edge, $S_{(u)}^{edge}$: edges incident to u.

1 Identification of 1-hop ($N(u)$) and 2-hop ($N^2(u)$) neighborhood via distributed beaconing and calculation of $clPCI$ indexes of the nodes;

2 Build local edge adjacency matrix $E_{(u)}^{mat}$ with $N(u)$ & $N^2(u)$;

/* $\exists\ e_{(i,j)} \in \mathrm{E} \iff i \in N(j) \wedge j \in N(i)$ */

3 Add weights $w_{i,j}^{edge} = clPCI(i) * clPCI(j)$ to $E_{(u)}^{mat}$;

4 Build $S_{(u)}^{edge} = w_{u,l_1}^{edge}, \ldots, w_{u,l_m}^{edge} \mid w_{u,l_k}^{edge} \in \mathrm{E},\ l_k \in \mathrm{N}(u)\ \forall_{k \le m}$;

5 **if** $\exists\ w_{u,l_k\ (1 \le k \le m)}^{edge} \in S_{(u)}^{edge}$ *not attached to DS edge* **then**

6 Select the edge with the largest weight and set M(u) = T;

7 M($w_{u,l_k\ (1 \le k \le m)}^{edge}$) = T; /* EDS election */

8 Announce status change;

9 **end**

10 Collect all edges (across the network) with a status=T;

5.4.1 PCI approaches

In Algorithm 5.1, lines (1)–(9) are distributed and executed by every node u in order to select which of the edges incident on it (i.e., on u) will be included in the $IEDS$. The selection is based on such a multilayer centrality measure. Since the centrality measure has been defined for nodes and not for edges, we use the product "value" of each edge's end-nodes to define the edge's value. The fictitious operation of line (10) unites every node's selection in order to construct the final $IEDS$. The proof of algorithm's correctness, in the sense that it constructs an $IEDS$ is very similar to that reported in [37, Theorem 4.2] and thus it will be omitted for all algorithms presented.

Proposition 5.1. *The computation complexity of $IEDS$ is $O(\Delta^2)$ in the worst case.*

Proof. The worst case computation complexity of $IEDS$ selection is when a node u has Δ neighbors and each one of them has Δ neighbors too. During the build-up of the edge adjacency matrix, node u needs to compare its 1-hop and 2-hop neighbor set with Δ^2 neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. The same computation cost applies to the population of the edge adjacency matrix node with the weight value $w_{i,j}^{edge}$ of each respective edge. The computation complexity of electing an edge as a DS edge is $O(\Delta^2)$, as node u needs to compare its 1-hop neighbor set with Δ neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. ⊣

Algorithm 5.2: MLEDS#1

precondition : Completed $IEDS$ election process
postcondition : Completed $MCEDS$ election process
remarks : $R(u)$: relay node set of node u,$M(u)$ / $M_(w_{i,j}^{edge})$: True(T) / False(F) indicator for node u / edge $w_{i,j}^{edge}$ being a DS node / edge.

1 **If** $M(u)$ = F **then** Return; /* not a DS node */
2 **repeat**
3 Add in $R(u)$ a node $l \in N(u)$ with the largest $clPCI$ index that covers at least one new node in $N^2(u)$;
4 M(l) = T; M($w_{u,l}^{edge}$) = T; /* $CEDS$ process */
5 **until** *each node in $N^2(u)$ is covered by node(s) in $R(u)$*
6 Announce status change;
7 Build $S_{(u)}^{edge} = w_{u,l_1}^{edge}, \ldots w_{u,l_m}^{edge} \mid w_{u,l_k\ (1\le k\le m)}^{edge} \in$ E, $l_k \in$ N(u), $M(l_k)$ = T;
8 Sort $S_{(u)}^{edge}$ in increasing order of the w^{edge} weights.
9 **repeat**
10 **if** $w_{u,l_k\ (1\le k\le m)}^{edge}$ *is dominated by connected* $w^{edges} \in E_{(u)}^{mat}$ *with larger weight* **then**
11 $M(w_{u,l_k\ (1\le k\le m)}^{edge})$=$F$; /* EDS Pruning */
12 Announce status change;
13 **end**
14 **until** *each* $w_{u,l_k\ (1\le k\le m)}^{edge} \in S_{(u)}^{edge}$ *has been considered*
15 Collect all edges (across the network) with a status=T;

The second algorithm, namely $MLEDS1$ (Algorithm 5.2), is the first that computes a $CEDS$; it starts from an $IEDS$ and connects it by adding edges that are bounded by DS nodes of the $IEDS$ and 1-hop relay nodes of them (those with the largest $clPCI$ index) who collectively cover their 2-hop neighborhood. Steps (1)–(14) are distributed and executed by each node u. Since adding edges in a distributed manner may result in redundant edge selection, $MLEDS1$ has a pruning phase (line 11). Line 15 is fictitious in order to fulfill the postcondition, i.e., it need not be run in practice.

Proposition 5.2. *The computation complexity of $MLEDS1$ is $O(\Delta^3)$ in the worst case.*

Proof. In order to connect the $IEDS$, each node u needs to check the status of its 1-hop neighbor set, which has a $O(\Delta)$ complexity. The computation complexity of the pruning phase is $O(\Delta^3)$, because a node u needs to calculate the coverage capability of a connected graph composed of both 1-hop and 2-hop neighbors in order to decide if it will act as a DS node or not. Thus, each node u compares its neighbor set with Δ^2 neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. ⊣

An improved version of the previous algorithm ($MLEDS2$) applies the more sophisticated pruning technique developed in [93] in order to reduce the size of the resulting connected edge dominating set. We omit its pseudocode and computational complexity here.

Algorithm 5.3: MLEDS#3

postcondition: Completed MCEDS election process

1 Identification of 1-hop ($N(u)$) and 2-hop ($N^2(u)$) neighborhood via distributed beaconing and calculation of $clPCI$ indexes of the nodes;
2 **repeat**
3 Add in $R(u)$ a node $l \in N(u)$ with the largest $clPCI$ index that covers at least one new node in $N^2(u)$;
4 **until** *each node in $N^2(u)$ is covered by node(s) in $R(u)$*
5 Announce $R(u)$;
6 **if** *selected as a relay node* **then**
7 $M(u) = T$; Announce status change;
8 Build $S_{(u)}^{constrained} = u_1, u_2, \ldots, u_n \mid u_{k\ (1\le k\le n)} \in N(u) \wedge N^2(u), M(u_{k\ (1\le k\le n)}) = T$, $clPCI(u) < clPCI(u_{k\ (1\le k\le n)})$;
9 **if** *$S^{constrained}$ is subject to $N(u) \subset N(u_1) \cup N(u_2) \ldots \cup N(u_n)$ and $u_1, u_2, \ldots, u_n$ form a connected graph* **then**
10 $M(u) = F$; Set M($w_{i,j}^{edge}$) = F any edge $w_{i,j}^{edge}$ incident to node u; /* CDS Pruning */
11 Announce status change; Return;
12 **end**
13 Build $S_{(u)}^{edge} = w_{u,l_1}^{edge}, w_{u,l_2}^{edge}, \ldots\ w_{u,l_m}^{edge} \mid w_{u,l_{k\ (1\le k\le m)}}^{edge} \in$ E, $l_k \in$ N(u), $M(l_k)$ = F;
14 **if** $\exists\ w_{u,l_{k\ (1\le k\le m)}}^{edge} \in S_{(u)}^{edge}$ *adjacent to a non DS edge and that edge is not incident to a DS node* **then**
15 M($w_{u,l_{k\ (1\le k\le m)}}^{edge}$)=T; /* $MCDS$ to $MCEDS$ */
16 Announce status change;
17 **end**
18 **end**
19 Collect all edges (across the network) with a status=T;

Finally, Algorithm 5.3 first creates a CDS and then computes a $CEDS$ through the addition of edges. Note that for such a node dominating set, all nodes are within one-hop of a selected node, so if we can judiciously add such connecting edges (between selected and non-selected nodes), we will have a $CEDS$. Steps (1)–(18) are executed in a distributed fashion by every node u.

Proposition 5.3. *The computation complexity of the relay node set election process is* $O(\Delta^3)$.

Proof. The prioritization phase involves neighbor sorting based on $clPCI$ value, which is a $O(\Delta * \log\Delta)$ operation. The worst case construction phase results when a node u has Δ neighbors and each one of them contributes Δ nodes to the coverage of the 2-hop neighborhood of u. In this case, node u needs to run once over its neighbor set of size $O(\Delta)$ and 'erase' those nodes of the 2-hop neighborhood of u (which has maximum size $O(\Delta^2)$) covered by the specific neighbor; this operation costs $O(\Delta^3)$. ⊣

Proposition 5.4. *The computation complexity of the pruning phase is* $O(\Delta^3)$.

Proof. A relay node u needs to check its 1-hop and 2-hop neighbors in order to decide if it will act as a relay node or not. Thus, each relay node u compares its neighbor set with Δ^2 neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. ⊣

Proposition 5.5. *The computation complexity of transforming the* $MCDS$ *to* $MCEDS$ *is* $O(\Delta^2)$ *in the worst case.*

Proof. The worst case computation complexity of the transformation process of the$MCDS$ to $MCEDS$ is when a non-DS node u has Δ non-DS neighbors and each one of them has Δ neighbors too. In such case node u needs to compare its 1-hop with Δ neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. ⊣

5.4.2 On the size relationship between IEDS and CEDS

Here we establish the relationship between the cardinality of an $IEDS$ and the cardinality of its corresponding[1] $CEDS$.[2].

Theorem 5.2. *Any* $IEDS$ *of size* $|IEDS|$ *can be turned into a* $CEDS$ *by adding* $2 \times |IEDS|$ *additional edges to the* $IEDS$ *in the worst case.*

Proof. We provide the proof sketch. Firstly, we will state a corollary that results immediately from the independent edge domination property, and then we will define the concept of *neighboring dominators* of an edge dominator e_v.

Corollary 5.1. *In any* $IEDS$*, the closest (in terms of hops) edge dominator to any edge dominator can be found one or two hops away, i.e.,* ≤ 2 *other edges are located in between these two edge dominators.*

[1] i.e., when $IEDS \subset CEDS$.

[2] Note that the claims of the theorem do not imply the relationship between the cardinality of the $IEDS$ and that of the graph's edge set.

Definition 5.9. *A neighboring edge dominator e_u of an edge dominator e_v is any edge dominator which is at most two hops away from e_v.*

An edge dominator e_v can have more than one neighboring edge dominator, but the exact number depends on network topology. Together, Corollary 5.1 and Definition 5.9 mean the topology between an edge dominator and its neighboring edge dominators must be one of the following:

C1 An edge dominator has at least one neighboring edge dominator one hop away. (e.g., edge dominator $\langle 1,2\rangle$ is one hop away from $\langle 7,9\rangle$ in Figure 5.3).

C2 An edge dominator has at least one neighboring dominator two hops away, and no dominators in one hop distance (edge dominator $\langle 1,2\rangle$ from $\langle 4,5\rangle$ in Figure 5.3 (LEFT)).

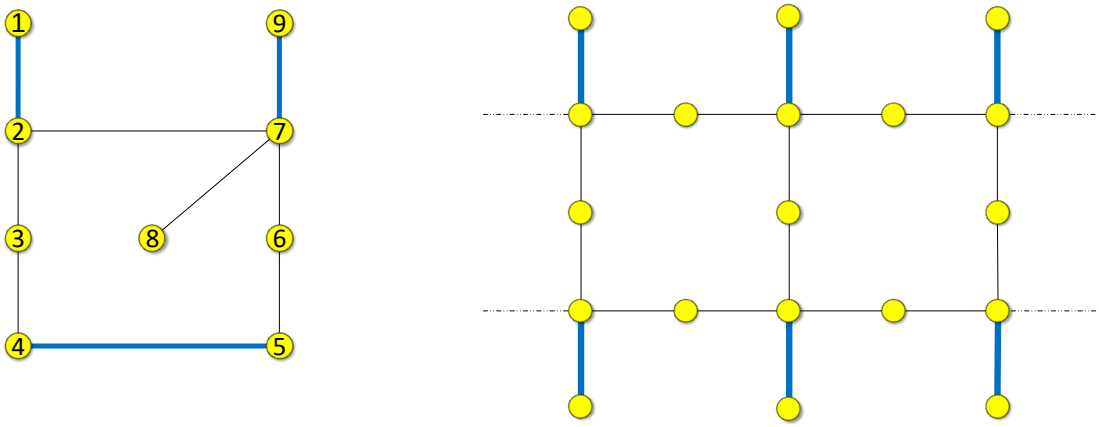

FIGURE 5.3. (LEFT) An IEDS (blue thick edges) which exhibits all possible relative locations of neighboring edge dominators. For instance, edge dominator $\langle 1,2\rangle$ is one hop away from $\langle 7,9\rangle$ and two hops away from $\langle 4,5\rangle$. (RIGHT) An IEDS (blue thick edges) which requires the maximum number of edge dominatees that must become dominators in order to get a CEDS. (Note that the graph extends infinitely to the left and to the right in the same pattern.)

If [C1] holds for some dominator e_v, then we need to include one more edge dominatee into the EDS in order to connect e_v to its nearer neighboring dominator. If [C2] holds for some dominator e_v, then we need to include two more edge dominatees into the EDS in order to connect e_v to its nearer neighboring dominator. Thus, in the worst case, for every edge dominator, we need to include two more edges into the EDS in order to make it a $CEDS$. The worst case occurs for $IEDS$'s as shown in Figure 5.3 (RIGHT). ⊣

5.5 Numerical results

We performed an evaluation of the algorithms in MATLAB. Since there is no prior work on our topic, we use as baseline algorithm (referred to as BASE) the very popular one proposed in [48] for node dominating sets, which we augment with a greedy heuristic to construct a $CEDS$. We have also developed a generator [12] to produce multilayer networks. We use the size (in percentages) of the resulting (connected) EDS as the performance measure. The champion algorithm will be the one that calculates the smallest size $CEDS$. The default value for average node degree is set to 10, for network diameter it is set to 8, and for the number of layers it is set to 4. Each figure encompasses four sets of plots aligned vertically, corresponding to four different settings for the number of nodes in each of the layers.

In Table 5.1 we present the impact of average network degree and diameter on the competitors' EDS size for default settings, and also on the number of inter-layer links included in the EDS as a resilience measure.

TABLE 5.1. Comparison of proposed algorithms to a baseline algorithm. For each competitor: the left column is the percentage of EDS size w.r.t. number of edges, and the right column is the percentage of inter-layer edges w.r.t. EDS size.

degree vs. (EDS size, # inter-layer links)										
deg	MLEDS# 1		MLEDS# 2		MLEDS# 3		BASE		IEDS	
3	0.54	0.66	0.34	0.45	0.29	0.42	0.58	0.39	0.14	0.19
6	0.34	0.61	0.23	0.51	0.21	0.53	0.42	0.25	0.11	0.21
10	0.19	0.37	0.15	0.46	0.15	0.49	0.26	0.21	0.07	0.22
15	0.11	0.25	0.11	0.47	0.11	0.49	0.15	0.18	0.05	0.21
20	0.09	0.21	0.08	0.48	0.08	0.53	0.12	0.12	0.04	0.23

diameter vs. (EDS size, # inter-layer links)										
diam	MLEDS# 1		MLEDS# 2		MLEDS# 3		BASE		IEDS	
3	0.21	0.39	0.21	0.48	0.16	0.51	0.25	0.29	0.08	0.21
5	0.32	0.54	0.32	0.47	0.20	0.50	0.36	0.33	0.10	0.20
8	0.33	0.57	0.33	0.45	0.21	0.47	0.39	0.38	0.10	0.20
12	0.46	0.64	0.46	0.43	0.25	0.44	0.51	0.41	0.12	0.19
17	0.55	0.66	0.55	0.45	0.32	0.42	0.62	0.42	0.15	0.20

We can see that the proposed algorithms succeed in including many inter-layer edges in the final $CEDS$; almost half of $CEDS$ edges are inter-layer ones. $MLEDS3$ in particular has stable behavior with respect to changes in network degree or diameter. On the other hand, BASE is the worst algorithm from the perspective of EDS size and this is consistent across all our experiments and therefore we refrain from presenting its performance in the sequel.

Figure 5.4 shows the performance of the algorithms as the average degree varies between 3 and 20. The immediate observation is that when the degree increases, the size of the EDS decreases for all competitors, which is to be expected given that in dense topologies a single

edge can dominate more edges. Also as expected are the observations that larger networks have relatively larger EDS's, as they must be sparser given that the average degree is fixed, and that the $IEDS$ algorithm leads to the smallest EDS, as it does not have to ensure connectivity. Among the algorithms that created connected EDS, $MLEDS3$ is the best performing algorithm, creating an EDS twice the size of that calculated by $IEDS$ which combined with Theorem 5.2 confirms that it is a good solution to our problem.

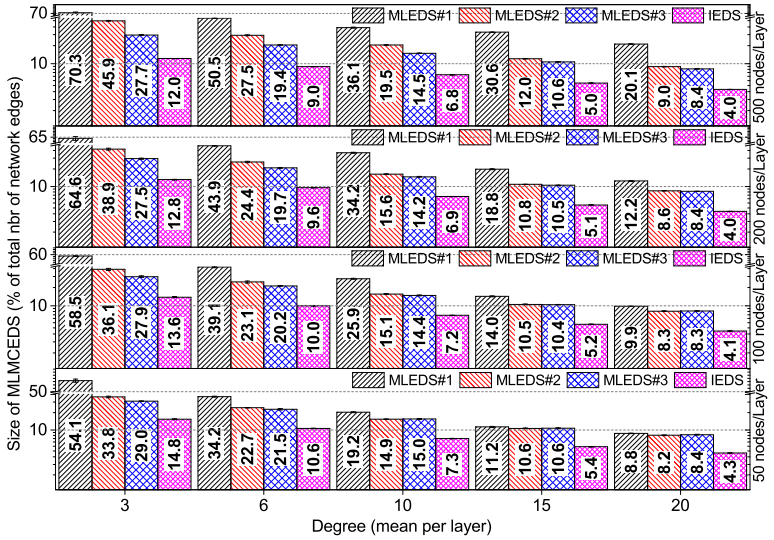

FIGURE 5.4. Impact of the average node degree on the size of CEDS.

Figure 5.5 shows the performance of the algorithms as the network diameter varies between 3 hops (so-called "bushy" networks) to 17 hops ('long and skinny' topologies). As expected, in "bushy" topologies, the resulting EDS's are smaller, whereas in the "long and skinny" topologies more dominating edges are needed.

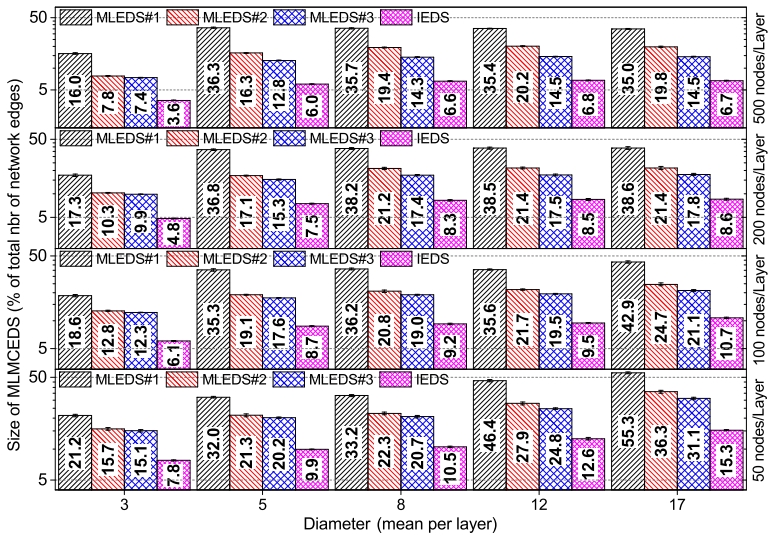

FIGURE 5.5. Impact of the network diameter on the size of CEDS.

As an analogy, in a star network (a "bushy" topology) a single edge can dominate all others, whereas in a line topology with k connections, the connected edge dominating set has cardinality $k-2$. Again, the best performing algorithm MLEDS3 is around 10% better than the second best algorithm on average. The performance gap reaches 25% for "longer and skinnier" topologies.

Figure 5.6 shows the performance of the algorithms as the number of layers varies. The increase in the number of layers causes the topology to become more connected, and as a consequence the size of the *EDS* reduces, but not as dramatically as when the diameter shrinks or when the density increases. Again, MLEDS3 is the best performing algorithm.

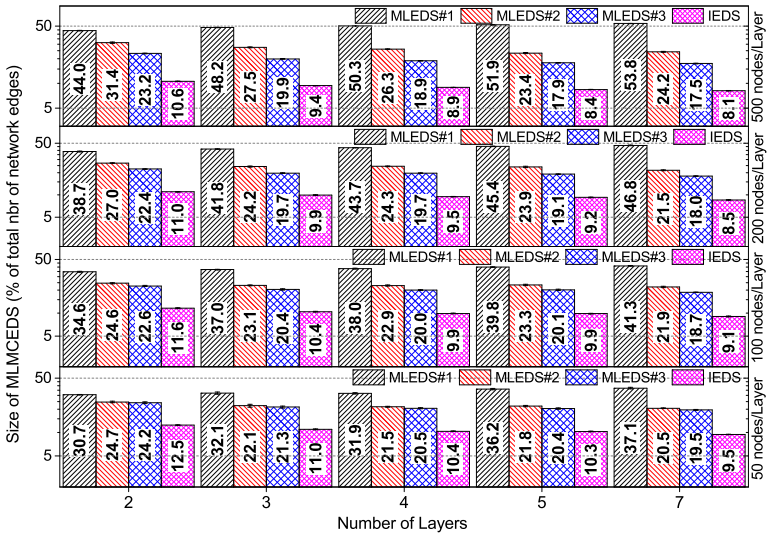

FIGURE 5.6. Impact of the number of layers on the size of CEDS.

5.6 Related work on MCMCEDS

The *MCMCEDS* problem, although novel per se, has connections to earlier work on finding minimum (connected) edge dominating sets. The *MEDS* problem has been shown to be NP-Complete in the single-layer case [139] in the centralized setting even for bipartite and planar graphs of maximum degree 3. Furthermore, even finding a 7/6−approximation of the optimal set has been shown to be NP-Hard [28]. The *MCEDS* problem has also been shown to be NP-Complete [87, p. 102, Lemma 4.4.3].

MCMCEDS generalizes the plain (without any colors and any weights) *EDS* problem [139], if we assume that all edges have the same color. However, *MCMCEDS* cannot be transformed into the plain *MCEDS* problem with weights on edges [18] by assigning a uniform small weight to all inter-layer edges, and a uniform large one to all intra-layer edges, as in this case we might end up including *all* inter-layer edges into the dominating set simultaneously, which is not necessarily the most efficient solution. This is significant; while a $3+\epsilon$ approximation exists for the weighted MCEDS problem [123], it will not apply to our *MCMCEDS* case. Problems related to *stratified domination* in graphs [21, 24] *ask for* a coloring of nodes, but in *MCMCEDS*, the

colors are provided as part of the input to the problem. Problems related to *chromatic transversal domination* [85] are also not related to $MCMCEDS$ for the same reason as stratified domination (in our case the colors are part of the input, and we do not seek a node coloring) and additionally because transversal domination demands that the dominating set's nodes should necessarily touch *all* color classes.

The problems most closely related to $MCMCEDS$ are those reported in [113], where color classes are given, but domination is defined such that *all or none* of the graph elements (edges in our case) of a color class should be included in the dominating set. However, the $MCMCEDS$ formulation allows for the inclusion of *any* number of edges belonging to *any* color class; therefore, the formulation is much more versatile (as compared to [113]) and encompasses a larger possible set of $MCEDS$s from which to choose from.

5.7 Conclusions

Motivated by applications in traffic monitoring in diverse communication systems, we presented distributed algorithms for the creation of connected Multi-colored Edge Dominating Sets in multilayer graphs. After showing that the underlying problem is hard to solve, we showed that a heuristic algorithm based on amending a connected node dominating set to create a connected edge dominating set provides the best performance. While our heuristics performed well over a range of scenarios, establishing approximability results for the $MCMCEDS$ problem represents an important line of future work.

CHAPTER 6

ENERGY-AWARE DISTRIBUTED EDGE DOMINATION OF MULTILAYER NETWORKS

6.1 Introduction

In this chapter we further explore the opportunities of maximizing the resiliency of a military multilayer ad hoc network by extending notions presented in some of our previous works [92, 94]. Modern military battlefields consist of an increasing array of entities with wireless communication and sensing capabilities. In [92], such heterogeneous allied systems have been modeled as multilayer ad hoc networks, in which each layer represents a type of battlefield entity (e.g., helicopters, UAVs, infantry); such multilayer tactical networks may arise in other settings as well, e.g., [105]. Designing networks with high numbers of inter-layer links immunizes the network to (possibly correlated) failures in any particular layer, allowing the design of resilient network overlays for purposes of either network management/monitoring or data forwarding, in the sense that the communication among different layers can not break easily (accidentally or due to malicious attacks).

While the increased number of layers and inter-layer links increases the resilience of such networks, it complicates network functions such as routing and scheduling. To accomplish these tasks successfully in the long run, the network must be able to sense changes in topology, and specifically link failures, efficiently, with the sensing system itself being resilient to such issues.

Related publication **[S3]:** Dimitrios Papakostas, Soheil Eshghi, Dimitrios Katsaros, Leandros Tassiulas. "*Energy-aware distributed edge domination of multilayer networks*", **Paper submitted for publication**.

The sensed information gathered (and possibly aggregated) by such a network overlay can be used in both centralized and decentralized control of the aforementioned network functions.

The problem of distributed computation of a resilient network overlay for communication link monitoring in single layer ad hoc networks has been studied in the past, e.g., [68]; it is a problem different from the traditional distributed approaches to ad hoc routing in that routing algorithms are designed aiming at minimizing latency (thus seeking shortest-paths) or increasing availability and preventing network choking (thus selecting multiple paths to destination).

The computation of a resilient network overlay for multilayer ad hoc networks is significantly harder to address, especially using distributed algorithms, because of coordination failures which can lead to loss of communication or over-dependence on a specific layer. A first effort towards achieving the goal of building resilient overlays for multilayer ad hoc networks introduced the use of custom-designed locally-computable centrality metrics [92]. Moreover, in [94] we proved that, in the context of multilayer networks, finding a $MCEDS$ with the additional goal of including many inter-layer links into the $MCEDS$ would improve its resiliency. However, neither the aforementioned works nor any prior work has considered the energy limitations of network entities in designing resilient overlays for link monitoring in multilayer ad hoc networks.

Even though some network entities might be energy-rich (e.g., vehicles) others can face severe energy needs, e.g., sensors, UAVs. Recharging batteries may be very difficult for entities deployed in a battlefield, as it is both time-consuming and detracts from attention to the mission at hand. Therefore, the overlay should be built in such a way that all included entities have enough energy to keep the overlay operational and connected for as long as possible. Such energy-aware overlay creation is the goal of this chapter. We emphasize that, the current problem calls for solutions that incorporate as many links among different layers as possible into the overlay with the goal of increasing resiliency. Thus we decided to extend the notions presented in [94], i.e., describe the problem in terms of calculating a $CEDS$, and take into consideration the energy state of each network node.

As an example, consider the two-layer network shown in Fig. 6.1, with the layers representing infantry and airborne units, and the node weights in parentheses denoting energy levels – the network is comprised of nodes S_i and C_i and the black (every link between nodes of the same layer) and red links (every link between nodes of different layers). It is assumed that communications between any two entities can traverse any path, and that monitoring the communications on an edge requires that we monitor at least one node of its end vertices.

All in all, our goal is to find a CEDS in such networks that is *small* in size, has many *inter-layer links*, and which is *energy-aware*, in a *distributed* manner. Each of these goals, which are driven by practical concerns, impose limitations on the set of acceptable network overlays, leading to possible trade-offs:

1. *Cardinality*: A smaller $CEDS$ limits the possible points of failure of the communication overlay.

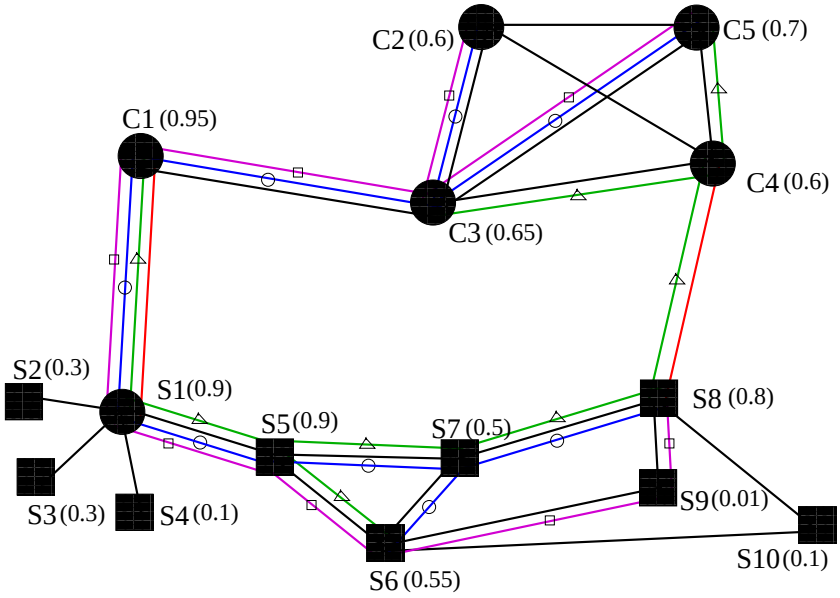

FIGURE 6.1. A multilayer network with (square and circular) nodes connected by black (intra-layer) and red (inter-layer) links. Three different connected edge dominating sets, namely set of purple edges (with small squares on the links), set of blue edges (with small circles on the links) and set of green edges (with small triangles on the links) are also depicted.

2. *Number of inter-layer links*: A resilient multilayer network overlay must limit dependence on any particular network layer while limiting the likelihood of "islanding", i.e., the loss of communication among network layers. Increasing the number of inter-layer links accomplishes these goals simultaneously.

3. *Energy awareness*: Communication and monitoring are both energy-intensive activities. The resulting energy depletion may exhaust the batteries of network elements, disconnecting the overlay and/or resulting in the loss of monitoring capabilities. In such settings, the $CEDS$ must be created anew, a time- and energy-intensive process. Thus, the distribution of energy among elements chosen in the overlay must be such that there are fewer low-energy elements.

4. *Distributed computation*: The aggregation of information and the implementation of centralized decisions are major challenges in such ad hoc networks. A distributed algorithm, in which the network elements make determinations on their presence or absence in the overlay using local information, allows the creation and maintenance of the overlay in battlefields.

For example, in Fig. 6.1, each of the purple (small squares on links), blue (small circles on links) and green (small triangles on links) sets of edges comprise a (minimum cardinality) $CEDS$. The blue and purple overlays have the same number of inter-layer links, yet the edges in the

blue overlay have end-points with higher energy-levels, making it a preferable option. The green overlay is the best among the three, because it has more inter-layer links and it is comprised of higher energy elements.

At this point, we wish to make clear the differences of the present work to our own research that initiated the study of domination for multilayer networks, and also to highlight the main chapter contributions. In [92, 93] we investigated analytically and experimentally the problem of node domination for multilayer networks, whereas here we study edge domination for multilayer networks. In [94] – which is the companion article of the present chapter – we introduced the problem of edge domination for multilayer networks, showed its relation to network control through the concept of maximal matching, gave complexity bounds and also, we developed heuristic algorithms. Here, we generalize the concepts of that paper by adding one more requirement, that of energy efficiency. This generalization presents challenges, because energy in not a feature of the edges, but of the network nodes, and thus new methods are needed to transform "node quantities" into "edge quantities". In summary, the present chapter contributes a generalization of the problem introduced in [94], presents its complexity, it provides simple and elegant methods to exploit node features in order to quantify edge significance, and it develops efficient heuristic algorithms to address the new problem.

The rest of the chapter is organized as follows: Section 6.2 formalizes the investigated problem; section 6.3 describes the proposed distributed heuristic algorithms to solve it; section 6.4 analyzes their performance and finally section 6.5 concludes the present work.

6.2 The EA-MCMCEDS problem

First, we build upon the notions presented in §5.2.1 regarding the domination concepts to describe the overlay in single layer settings [54], and then factor in the effects of a multilayered architecture and energy-awareness, which set our problem, henceforth called the *Energy-Aware Minimum Connected Multi-Colored Edge Dominating Set* (EA-MCMCEDS), apart from the existing literature. The subsections proceed in the order of the four goals outlined in §6.1.

6.2.1 Minimum cardinality connected edge domination

Given a single layer network $G = (V, E)$ and by following Definition 5.1 about EDS, for a vertex $v \in E$ one of the following two possibilities holds:

Case (a) v will be one of the two endpoints of an edge belonging to the EDS; in which case we will call it a *member of the overlay*, e.g., vertices $S6$ and $S7$ for either blue or green overlay in Fig. 6.1.

Case (b) v, a *non-member of the overlay*, will be at one hop distance from a member of the overlay, e.g., vertex $S2$ for either blue or green overlay in Fig. 6.1.

We can easily observe that *all* vertices at an one-hop distance from a non-member of the overlay will be overlay members. This means that placing monitoring devices on the vertices (endpoints) of EDS edges will allow the monitoring of *all* communication within the system.

Overlay vertices can further be categorized into two groups:

Case (a1) v will be a *core overlay member* if more than one EDS edges is incident to v; e.g., vertex $S7$ in Fig. 6.1 for either the blue or green overlay.

Case (a2) Otherwise, if exactly one EDS edge is incident to v, it will be a *peripheral overlay member*; e.g., vertex $S6$ in Fig. 6.1 for either the blue or green overlay.

A further refinement to the EDS concerns the notion of independence to the chosen edges. Based on Definition 5.2 about the $IEDS$ (also called a maximal matching [123]) any minimum-cardinality EDS will also be a MEDS. Note that, for an EDS of a certain cardinality (number of edges), the $IEDS$ will require the most number of monitoring devices. However, there may exist some $MEDS$s requiring fewer monitoring devices than the minimum-cardinality $IEDS$ [139].

While the EDS captures the goal of monitoring communications, it may not be able to provide the coordination and communication capability that is required of an overlay. For coordination within the chosen subset of edges, we need the EDS to be connected ($CEDS$).

6.2.2 Edge domination in multilayer networks

We now consider the equivalent of the single layer concepts described above in the context of multilayer networks, and describe how the relevant concepts described in §6.1 can be captured in the creation of an overlay in such networks. We first define (G^{ML}, E^{ML}) to represent a multilayer network, with $G^{ML} = \{G_1, G_2, \ldots, G_m\}$ such that:

- $G_i = (V_i, E_i)$ is a single layer network for all $1 \leq i \leq m$, with m being the number of layers.
- $E^{ML} = \{E_{ij} \subseteq V_i \times V_j | 1 \leq i, j \leq m, i \neq j\}$ is the set of existing inter-layer links.

It follows that:

Definition 6.1. *A $CEDS$ of the multilayer network (G^{ML}, E^{ML}) is a set of edges $E'' \subseteq (\bigcup_{i=1}^{m} E_i) \bigcup E^{ML}$ such that the induced single layer subgraph on the vertex-set $\bigcup_{i=1}^{m} V_i$ is a $CEDS$.*

The communication overlay should not be excessively reliant on any single layer, as it would be vulnerable to correlated failures. Thus, an ideal overlay would minimize the total number of edges within the overlay while at the same time maximizing the number of inter-layer links.

Definition 6.2. *A Multi-Colored Minimum Connected Edge Dominating Set (MCMCEDS) E'' of the multilayer network (G^{ML}, E^{ML}) is a $CEDS$ with minimum cardinality $|E''|$ that has the maximum number of inter-layer links, $|E'' \bigcap E^{ML}|$. Such a set is called multi-colored due to the practice of assigning a different non-black color to the elements of E^{ML} for each pair of layers connected by an edge, with all elements in $(\bigcup_{i=1}^{m} E_i)$ being considered black [94, Definition 7].*

6.2.3 Energy availability and constraints

Key results within the field of ad hoc networks have pointed to the importance of factoring in the energy distribution, and not just the total energy content, and moreover it has been shown that under certain circumstances, network lifetimes are maximized under equitable distributions of energy within the network and that optimal communication policies put more of the communication burden on the network elements with the most energy [22, 43]. Here, we assume that the energy available to each network element is known to that network element and can be communicated to its neighbors. Thus, we can adapt our overlay creation methods to be *energy-aware*:

Definition 6.3. *An MCMCEDS creation method is Energy-Aware (EA) if it utilizes the energy available to network elements to find an overlay with high energy elements.*

Ideally, all overlay nodes will have high energy, yet there is a possible trade-off between the cardinality and connectedness of the $MCMCEDS$ and the energy of the overlay elements. Given the results described from the ad hoc networks literature, our aim is to avoid elements with low energy in the overlay, while also having higher energy nodes at *core* nodes, as such nodes have to relay information as well as performing the sensing and communication required of all overlay nodes. We further explore these concepts in our simulations (§6.4.3).

6.2.4 Distributed energy-aware overlay generation

The nature of ad hoc networks demands the overlay creation algorithm to be distributed, with each element only having knowledge of their k-hop neighborhood. (Here, $k = 2$.) This adds the final element to the problem at the heart of this chapter:

Problem 6.1 (Distributed EA-MCMCEDS Computation)**.** *We seek to find an energy-aware distributed algorithm that computes an $MCMCEDS$ of a multilayer network given the energy available to network elements (i.e., vertices).*

Proposition 6.1. *Distributed EA-MCMCEDS computation is NP-hard.*

It is easy to prove Proposition 6.1 following the reasoning of [94, Theorem 1], and thus we will hence develop heuristic algorithms to solve the Distributed EA-MCMCEDS problem.

Algorithm 6.1: $CCEDS$

precondition : Known $EclPCI$ index values of nodes in $(N(u)) \wedge (N^2(u))$

postcondition : Completed $MCEDS$ election process

remarks : mlNetwork G = (V, E) where V and E are vertex & edge set, $R(u)$: relay node set of node u, $M(u)$ / $M(w_{i,j}^{edge})$: (T)rue / (F)alse indicators for node u / edge $w_{i,j}^{edge}$ being a DS node / edge.

repeat
  Add node $l \in N(u)$ with largest $EclPCI$ & which covers at least one new node in $N^2(u)$ to $R(u)$;
until *each node in* $N^2(u)$ *is covered by node(s) in* $R(u)$
Announce $R(u)$;
if *selected as a relay node* **then**
  $M(u) = T$; Announce status change;
  Build $S_{(u)}^{constrained} = u_1, u_2, \ldots, u_n \mid u_{k\ (1\leq k\leq n)} \in N(u) \wedge N^2(u)$, $M(u_{k\ (1\leq k\leq n)}) = T$, $EclPCI(u) < EclPCI(u_{k\ (1\leq k\leq n)})$;
  if $S^{constrained}$ *is subject to* $N(u) \subset N(u_1) \cup N(u_2) \ldots \cup N(u_n)$ *and* $u_1, u_2, \ldots, u_n$ *form a connected graph* **then**
    $M(u) = F$; Set M($w_{i,j}^{edge}$) = F any edge $w_{i,j}^{edge}$ incident to node u; `/* CDS Pruning */`
    Announce status change; Return;
  end
  Build $S_{(u)}^{edge} = w_{u,l_1}^{edge}, w_{u,l_2}^{edge}, \ldots\ w_{u,l_m}^{edge} \mid w_{u,l_k\ (1\leq k\leq m)}^{edge} \in$ E, $l_k \in$ N(u), $M(l_k)$ = F;
  if $\exists\ w_{u,l_k\ (1\leq k\leq m)}^{edge} \in S_{(u)}^{edge}$ *adjacent to a non DS edge and that edge is not incident to a DS node* **then**
    Add $w_{u,l_k\ (1\leq k\leq m)}^{edge}$ in the EDS ;
    Announce status change;
  end
end

6.3 Proposed distributed algorithms

Here we describe three energy-aware distributed algorithms that heuristically solve the $EA-MCMCEDS$ problem. The common principle in all the proposed algorithms is that when seeking which edges to include into the edge dominating set, these edges are selected based on their ability to a) dominate many other edges, b) connect different layers, and c) have energy-rich endpoints. Towards translating these goals into a heuristic rule for selecting edges, we use the local centrality measure we proposed in [92], $clPCI$, which we now enhance to take energy levels into account. Striving for simplicity and generalization, along the lines of earlier works (e.g., [36, 93], we adopt a plain generalization of $clPCI$, termed $EclPCI$, for a node u with energy $E(u)$, as follows:

$$EclPCI(u) := E(u) \times clPCI(u). \tag{6.1}$$

The computation complexity of $EclPCI$ index is $O(\Delta^2)$ in the worst case, where Δ is the maximum node degree in the network [93].

For the construction of the edge dominating set, the proposed algorithms either work with nodes to calculate node dominating sets and then turn them into edge dominating sets, or with edges (actually on their equivalent nodes in the *line graph* [54] of the original graph). The value of *EclPCI* is used to prioritize nodes/edges. Whenever link-weights are needed, we define them to be the product of the *EclPCI* values of the endpoints (vertices) of the link, so as to prioritize links whose endpoints are both energy-rich vertices and posses strategic position within the topology. In the interest of space, we give brief descriptions of the algorithms and provide their computation complexities and their pseudo-codes.

The first algorithm, $CCEDS$, calculates a CDS and then applies a pruning mechanism using connectivity as quantified by $EclPCI$ to establish a total order among nodes in the CDS. The set of edges whose endpoints are CDS nodes comprise the initial EDS. Finally, it adds into the EDS any edges that are attached to edges not incident to DS nodes. The details of the algorithm are shown in Algorithm 6.1.

Proposition 6.2. *The computation complexity of CCEDS is $O(\Delta^3)$, where Δ is the maximum node degree in the network.*

Proof. The worst case regarding the construction phase of the CDS results when a host u has Δ neighbors and each one of them contributes Δ nodes to the coverage of the 2-hop neighborhood of u. In this case, host u needs to run once over its neighbor set of size $O(\Delta)$ and "erase" those nodes of the 2-hop neighborhood of u (which has maximum size $O(\Delta^2)$) covered by the specific neighbor. Further, the computation complexity of the respective pruning phase is also $O(\Delta^3)$ because a relay node u in order to decide if it will act as a relay node or not it needs to calculate the coverage capability of a connected graph composed of both 1-hop and 2-hop neighbors. Thus, each relay node u compares its neighbor set with Δ^2 neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. Finally, the computation complexity of complementing the EDS with some "obsolete" edges; i.e., that are not attached to a DS node, is $O(\Delta^2)$ because a relay node u needs to run once over its neighbor set of size $O(\Delta)$ and check for those neighbors that are not DS nodes if they have a non ds neighbor, and the neighbor set comparison has a $O(\Delta)$ complexity. ⊣

The second algorithm, $EPEDS$, first calculates an $IEDS$ and then connects it. To elaborate, during the $IEDS$ creation process each node selects the highest-weight undominated edge incident to it and adds it to the EDS (if any exist). In order for the $IEDS$ to be converted into a $CEDS$, each node that belongs to a DS edge adds enough incident edges (prioritized by the $EclPCI$ of the one-hop neighbor at the other end of the edge) to collectively dominate its two-hop neighborhood. Finally, this algorithm uses a generic pruning policy which recognizes and then removes redundant edges with small weights from the $CEDS$. The details of the algorithm are shown in Algorithm 6.2.

Algorithm 6.2: *EPEDS*

precondition : Known $EclPCI$ index values of nodes in $(N(u)) \wedge (N^2(u))$

postcondition : Completed $MCEDS$ election process

1 Build edge adjacency matrix $E^{mat}_{(u)}$ with $N(u)$ & $N^2(u)$;

/* $\exists\ e_{(i,j)} \in \mathrm{E} \iff i \in N(j) \wedge j \in N(i)$ */

2 Add weights $w^{edge}_{i,j} = EclPCI(i) * EclPCI(j)$ to $E^{mat}_{(u)}$;

3 Build $S^{edge}_{(u)} = w^{edge}_{u,l_1}, \ldots, w^{edge}_{u,l_m} \mid w^{edge}_{u,l_k\ (1\leq k\leq m)} \in \mathrm{E}, l_k \in \mathrm{N}(u)$;

4 **if** $\exists\ w^{edge}_{u,l_k\ (1\leq k\leq m)} \in S^{edge}_{(u)}$ *not attached to DS edge* **then**

5 Select the edge with the largest weight and set $\mathrm{M}(w^{edge}_{u,l_k\ (1\leq k\leq m)}) = T$; /* EDS election */

6 Announce status change;

7 **end**

8 **repeat**

9 Select a node $l \in N(u)$ with the largest $EclPCI$ index value that covers at least one new node in $N^2(u)$;

10 $\mathrm{M}(l) = \mathrm{T}$; $\mathrm{M}(w^{edge}_{u,l}) = \mathrm{T}$; /* $CEDS$ process */

11 **until** *each node in* $N^2(u)$ *is dominated by at least one DS node in* $N(u)$

12 Announce status of nodes in $N(u)$;

13 Build $S^{edge}_{(u)} = w^{edge}_{u,l_1}, \ldots\ w^{edge}_{u,l_m} \mid w^{edge}_{u,l_k\ (1\leq k\leq m)} \in \mathrm{E}, l_k \in \mathrm{N}(u), M(l_k) = \mathrm{T}$;

14 **repeat**

15 **if** $w^{edge}_{u,l_k\ (1\leq k\leq m)}$ *is dominated by connected* $w^{edge} \in E^{mat}_{(u)}$ *with larger weight* **then**

16 $M(w^{edge}_{u,l_k\ (1\leq k\leq m)}) = F$; /* EDS Pruning */

17 Announce status change;

18 **end**

19 **until** *each* $w^{edge}_{u,l_k\ (1\leq k\leq m)} \in S^{edge}_{(u)}$ *has been considered*

Proposition 6.3. *The computation complexity of EPEDS is* $O(\Delta^3)$*, where* Δ *is the maximum node degree in the network.*

Proof. The worst case regarding the construction phase of the EDS results when a host u has Δ neighbors and each one of them has Δ neighbors too. The adjacency matrix build up and its subsequent population with the weight value $w^{edge}_{i,j}$ of each respective edge requires a node u to compare its 1-hop and 2-hop neighbor set with $O(\Delta^2)$ neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. Further, the computation complexity of electing an edge as a DS edge is $O(\Delta^2)$, as a node u needs to compare its 1-hop neighbor set with Δ neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. Connecting the EDS requires a host u to run once over its neighbor set of size $O(\Delta)$ and "erase" those nodes of the 2-hop neighborhood of u (which has maximum size $O(\Delta^2)$) covered by the specific neighbor. Finally, the computation complexity of the pruning phase is $O(\Delta^3)$, because a node u needs to calculate the coverage capability of a connected graph composed of both 1-hop and 2-hop neighbors in order to decide if it will act as a DS node or not. Thus, each node u compares its

neighbor set with Δ^2 neighbors in the worst case, and the neighbor set comparison has a $O(\Delta)$ complexity. ⊣

The third algorithm, $NPEDS$, uses the mechanics of $EPEDS$ to calculate the $CEDS$, and then uses the pruning mechanism of $CCEDS$.

The computational complexity of $NPEDS$ is upper bounded by $O(\Delta^3)$ where Δ is the maximum degree in the network. We will not provide the detailed proof here because it is a mix of the previous two algorithms.

6.4 Performance evaluation

Next we present the competing algorithms and the datasets we used in the experiments and provide the analytical results.

6.4.1 Competing algorithms

We compare the performance of the three proposed algorithms, $CCEDS$, $EPEDS$ and $NPEDS$, across the various aspects of the $EA-MCMCEDS$ problem. Moreover, since degree-based node dominating set construction could be a viable technique, we developed $WCEDS$, which uses a straightforward energy-aware generalization of degree centrality [90] for multilayer networks, as a benchmark. $WCEDS$ uses the same mechanics as $CCEDS$ to calculate the $CEDS$, with the exception that the weighted degree centrality is used in place of $EclPCI$.

6.4.2 Datasets

Due to the lack of publicly available, real-world military multilayer networks, we developed a generator for multilayer weighted networks which is described in detail in [12]. The construction of a multilayer network is controlled by the average degree of each node, by the number of nodes per layer (i.e., size of the layer), the diameter of each layer, and the number of layers. We apply four distinct *Zipfian* distributions, one per parameter of interest, controlled by the skewness parameter s of the respective Zipfian distribution:

- $s_{degree} \in (0,1)$: to generate the frequency of appearance of highly interconnected nodes; therefore the degree distribution is Zipfian.
- $s_{layer} \in (0,1)$: to choose how frequently a specific layer is selected; therefore the layer IDs collectively as edge anchors follows a Zipfian distribution.
- $s_{node} \in (0,1)$: to choose how frequently a specific node is selected in a specific layer; therefore the distribution of node IDs as edge endpoints is Zipfian distributed.
- $s_{weight} \in (0,1)$: to choose how energy is distributed to nodes; therefore the energy distribution is Zipfian.

We selected a default setting for each of the parameters. Collectively, we call these parameters the *skewness*, and we represent them as a sequence of four floats, e.g., 0.5 – 0.5 – 0.5 – 0.5, which means that $s_{degree} = 0.5$, $s_{layer} = 0.5$, $s_{node} = 0.5$ and $s_{weight} = 0.5$ (the default settings we used to create the datasets). Table 6.1 records all the independent parameters of our topology generator, their range of values, and their default values.

TABLE 6.1. Experimentation parameters values.

parameter	**range**	**default**
avg. node degree (D)	3, 6, 10, 15, 20	6
network diameter (H)	3, 5, 8, 12, 17	8
#network layers (L)	2, 3, 4, 5, 7	4

6.4.3 Results

We performed a simulation-based performance evaluation of the competing algorithms in MATLAB. We include $IEDS$ in the plots as a benchmark, but do not comment on its performance because it does not create a $CEDS$.

In Figs. 6.2-6.4, we plot the performance of the competitors in terms of the the goals of the $EA-MCMCEDS$ problem as the mean degree, diameter, and number of layers of the synthetic multi-layer network is varied. In these figures, the first row of histograms show the size of the $CEDS$ that each algorithm creates, which is measured as a percentage of the total edges in the network, while the second row plots show the percentage of all the inter-layer links that are included in the $IEDS$. An ideal algorithm will minimize the former, while maximizing the latter. The third row of plots show the energy distribution of the vertices selected for the overlay, plotting their mean energy with associate error bars, while the fourth row of plots show the number of overlay nodes. An ideal algorithm would cover the network using relatively few, high-energy nodes, so it would maximize the third row of plots while minimizing the fourth.

6.4.3.1 Impact of topology density

In Fig. 6.2, we consider the impact of topology density on each competitor's performance. In the top row, we evaluate the size of the EDS that each competitor creates. We first observe that the size of the EDS is almost a decreasing function of node density, as in higher network densities, each node will participate in more edges, and each edge can thus dominate more edges. It is interesting that both $CCEDS$ and $WCEDS$ manage to create the smallest $MCEDS$ (approximately 2.0 up to 2.1 times the size of $IEDS$) regardless of how sparse or dense the network is. On the other hand, $EPEDS$ and $NPEDS$ create the largest $EDSs$ for mean degrees 3 and 6 (with $> 60\%$ and $> 50\%$ more edges, respectively), while they perform close to the other heuristics for larger mean degrees. The best performing algorithms in sparse and medium density networks start with an $MCDS$ in creating an $MCEDS$, which means that they can start from much sparser

overlay sets compared to methods that require the creation of a $CEDS$ from the start, as without coordination, many edges are needed for domination in sparse networks.

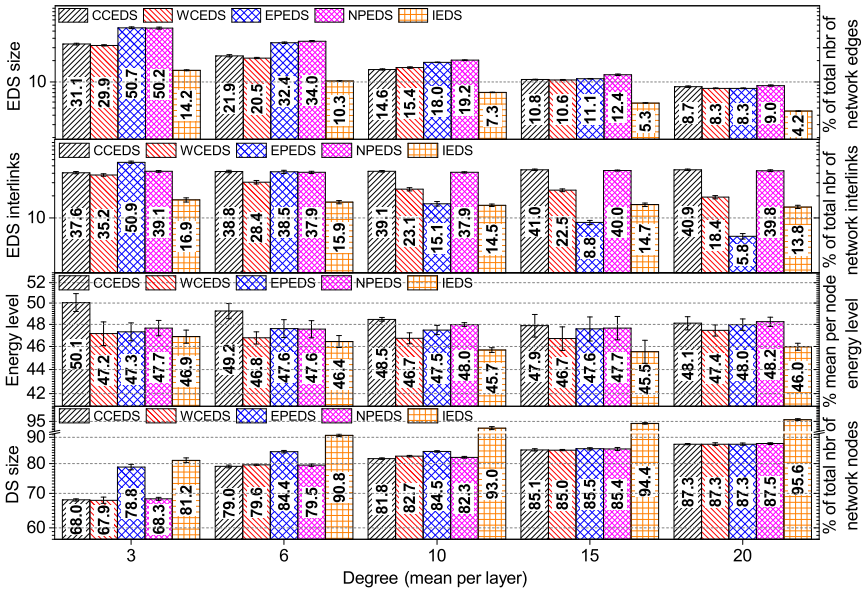

FIGURE 6.2. Impact of network density on the performance of each competitor.

In the second row, we see that $CCEDS$ and $NPEDS$ include more inter-layer links than competitors, with performance levels that do not change much with network density. On the other hand, while $EPEDS$ is the best performing algorithm when mean degree = 3, its performance drops drastically for larger mean degrees. $WCEDS$ also has a similar, yet less drastic, performance drop. This is because $EclPCI$, as used by $CCEDS$ and $NPEDS$ enables these algorithms to privilege inter-layer links for inclusion in the $CEDS$, while $WCEDS$ is layer-agnostic. The surprising drop in performance for $EPEDS$ is due to its edge-pruning mechanism (as opposed to the node-pruning mechanism of $CCEDS$/$NPEDS$), which increases the likelihood of pruning inter-layer links: as the probability of the existence of a dominating edge in either of the linked layers increases with the mean degree, and the probability of self-deselection by the inter-layer edge in the pruning phase also increases.

In the third row, we see that $CCEDS$ on average, leads to a $3-5\%$ increase in overlay node energies in sparse networks (mean degrees 3 and 6) compared to the other competitors. This difference levels out at higher densities, except with the worst-performing $WCEDS$. This is tightly coupled with the arguments around the size of the selected EDS presented for the first row - $CCEDS$ chooses fewer, yet more central and higher energy edges in sparse networks precisely because it does not impose creating a $CEDS$ from the first step.

In the last row, we plot the relative size of the overlays. Interestingly, and counter-intuitively, we see that the overlay size grows with the mean degree. We may expect that a denser network could be dominated by fewer overlay nodes. However, the number of edges in a network grows with the edge density, and as we are seeking to build an *edge*-dominating set, the number of

overlay nodes also grows. In this aspect as well, $CCEDS$, $NPEDS$, and $WCEDS$ perform best, with $EPEDS's$ edge-based pruning leading to a $> 14\%$ and $> 6\%$ handicap for mean degrees 3 and 6.

6.4.3.2 Impact of network diameter

In Fig. 6.3, we consider the impact of network diameter on each competitor's performance. In the top row, we observe that the size of the constructed EDS increases with the network diameter for all algorithms. This is the result of sparser neighborhoods, i.e., fewer links between network nodes. In other words, there are fewer, longer (in hops), and less distinct paths in the multilayer network, which leads to the election of a large number of edges for the EDS. Except for the worst-performing $EPEDS$ (whose size goes from 29% larger than the best competitor when diameter = 3 up to 82% larger when diameter = 17) all the algorithms lead to similar EDS sizes. The pruning mechanism of $EPEDS$ is responsible for its bad performance.

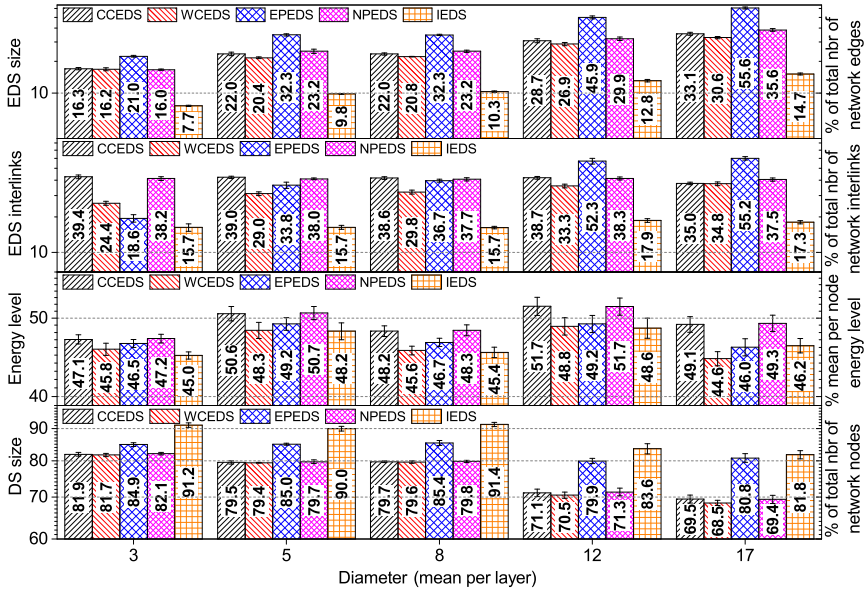

FIGURE 6.3. Impact of network diameter on the performance of each competitor.

From the second row of plots, we see that $CCEDS$ and $NPEDS$ perform best overall with regards to the number of the EDS interlinks, with the relative number of EDS interlinks staying stable as the diameter is varied. This means that they both work well in bushy (small number of hops) or skinny (large number of hops) networks. $EPEDS$, on the other hand, performs remarkably well for larger network diameters, while performing poorly when diameter = 3 (where it includes less than the half number of interlinks in the EDS compared to $CCEDS$). For example, $EPEDS$ creates an overlay with $> 45\%$ more interlinks compared to $CCEDS$ and $NPEDS$ when diameter = 12 or 17, yet this comes at the cost of a much larger EDS size, which is unacceptable. $WCEDS$, the layer-agnostic baseline, only reaches the respective performance of the other algorithms in terms of interlinks when diameter = 17, while having much fewer

interlinks for more bushy networks. So, while $WCEDS$ performs well in terms of EDS size for bushy networks, this is traded off against the lower resiliency of the resulting overlay. Note that the number of the interlinks that are included in the EDS by each algorithm is a direct consequence of the pruning mechanism they employ, and more precisely, how well each of them can distinguish between a simple edge and an interlink.

From the third row of plots, we see that all the algorithms lead to overlays with similar average energy levels and there is no clear winner. However, for medium-to-large network diameters (when it equals 8, 12 and 17) both $CCEDS$ and $NPEDS$ select nodes with slightly more energy (on average) into the overlay (showing a 3.5–10% improvement).

Interestingly, in the last row of plots, we see that the DS size decreases significantly for all the competitors when diameter = 12 and 17. This arises from the fact that the multilayer network is created from the interconnection of sparse and skinny networks in these settings. In such a case, the DS nodes are shared between many EDS edges. Most of the algorithms lead to similar DS sizes, except for $EPEDS$ which leads to 3.5% (when diameter= 3) to 18.0% more nodes in the DS. This is due to its inefficient pruning mechanism.

6.4.3.3 Impact of number of layers

In Fig. 6.4, we consider the impact of the number of network layers on each competitor's performance. From the top row of plots, we see that for the majority of the algorithms (all except $EPEDS$), the number of edges in the EDS decreases with an increase in the number of the multilayer network layers, leading to EDS selections approximately 2.0–2.4 times the size of the $IEDS$, because of the richer connectivity among layers' hub nodes imposed for the specific value(s) of zipfian distribution(s). We also observe that all competitors lead to similar size EDS selections, except for $EPEDS$ whose EDS is 30% larger than the rest.

From the second row of plots, we observe that the number of EDS interlinks decreases as the number of layers increases. This is because it is increasingly difficult for all the algorithms to distinguish between inter-layer and intra-layer edges when the total number of edges increases (a by-product of the increase in the number of layers). However, both $CCEDS$ and $NPEDS$ manage to confront this problem more efficiently than the competition when we have 2-layer and 3-layer networks, with both having at least 25% and 38% more interlinks in the EDS than the competition, respectively. This is because $EclPCI$ can improve the ability of an algorithm to distinguish between interlinks and intra-layer edges. It is interesting to note that the relative differences in the number of interlinks chosen by the competitors does not change with the number of network layers, except in the case of the 7-layer network, in which EPEDS chooses a relatively large number of interlinks in its relatively large EDS. Again, $WCEDS$ has the worst performance among the competitors in this aspect due to its layer-agnostic nature, except for the 2-layer network where it performs 12% better than $EPEDS$.

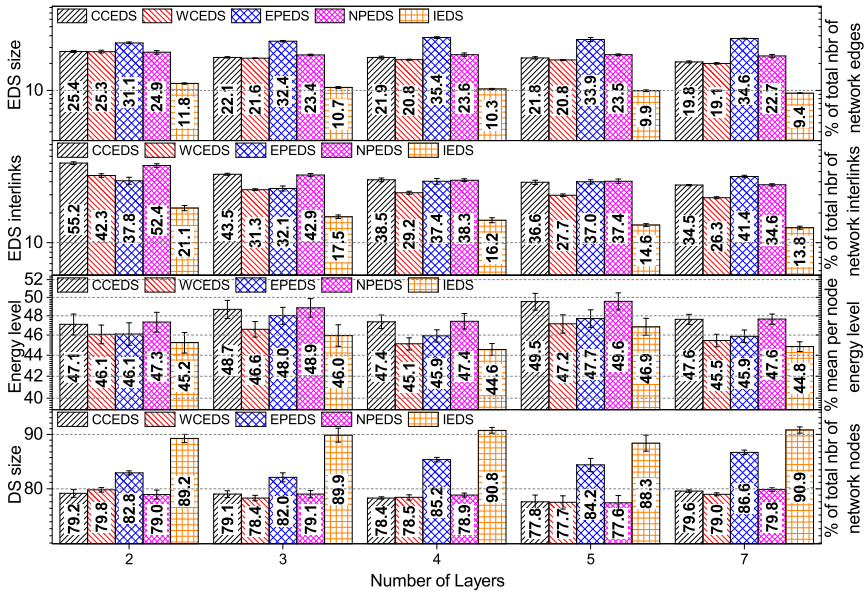

FIGURE 6.4. Impact of number of layers on the performance of each competitor.

In the third row of plots, we see that both $CCEDS$ and $NPEDS$ create overlays with the most average energy per node irrespective of the number of network layers. Interestingly, the number of network layers has no effect on the relative differences in the performance of the competitors.

Finally, in the bottom row of plots we observe that the percentage of nodes included in the DS does not change with the number of network layers for the majority of the competitors. Once again, the exception is $EPEDS$, for which the number of DS nodes increases with the number of layers.

6.4.3.4 Energy analysis of the overlay

In Fig. 6.5, we analyze the (average) energy levels of the core, periphery, and non-members of the overlay along with the size of the EDS for the competing algorithms, in order to illustrate the tradeoff between picking only high-energy nodes and providing sufficient coverage. Each bar has three colored segments: a pink segment corresponding to core overlay nodes, a light orange segment corresponding to peripheral overlay nodes, and a light green segment corresponding to non-members of the overlay. The height of each colored segment represents the percentage of nodes belonging to that node class; therefore, the sum of the heights of the segments is 100%. The number superimposed within each colored segment depicts the average energy of the nodes belonging to that class.

An algorithm will be efficient in terms of overlay size if the total height of the associated pink and the light orange segment is small relative to its competitor(s). We observe from the plots (which vary the number of layers, the diameter, and the mean degree) that $CCEDS$ produces the smallest overlay in (almost) all cases, as observed in earlier figures as well.

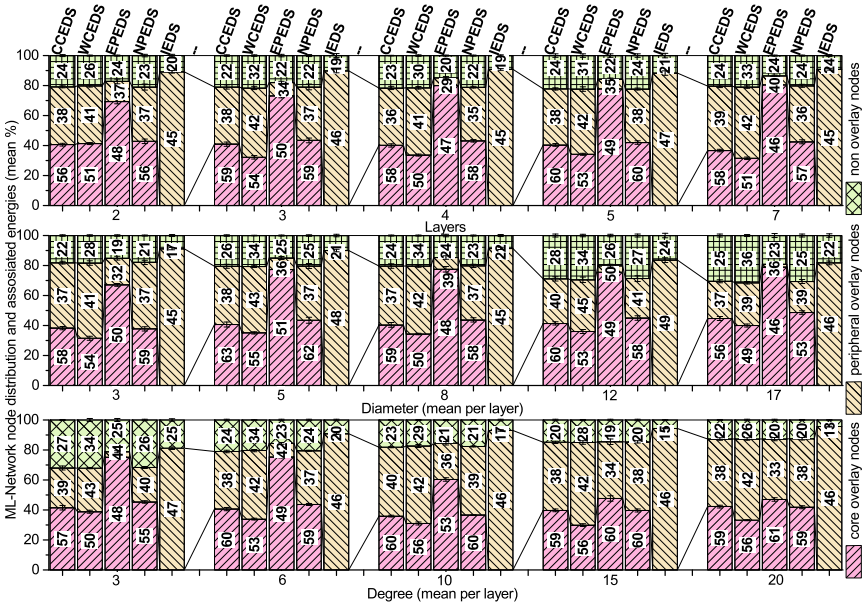

FIGURE 6.5. Energy levels (average) of the core, periphery, and non-members of the overlay along with the size of the EDS.

An algorithm is efficient in terms of energy, *primarily* if the average energy of the its core overlay nodes (superimposed number of the pink segment) is high and the average energy of the overlay's non-members (super-imposed number over the light green segment) is low. We observe that on average, $CCEDS's$ core overlay nodes have higher energy than those of their competing algorithms and outperform the second best algorithm, $NPEDS$, by around 4% in some cases. However, $NPEDS$ has quite similar performance to $CCEDS$ in general, and is slightly superior to $CCEDS$ for networks with very small diameter (1.5% improvement).

6.5 Conclusions

We considered distributed methods of creating a minimum-size overlay network of monitoring devices over a wireless multilayer ad hoc network. We emphasized the overlay network's resilience to correlated layer failures and to energy depletion in devices, paying special attention to inter-layer links, and formalized the problem in terms of edge dominating sets (EDS) in multilayer networks ($EA-MCMCEDS$).

We proposed three distributed algorithms for solving $EA-MCMCEDS$, namely $CCEDS$, $EPEDS$ and $NPEDS$. $CCEDS$ creates a $CEDS$ by first starting from a node dominating set, whereas the other two start from an independent EDS. All employ smart pruning heuristics to reduce the size of the resulting connected EDS. We compared their performance, along with that of a baseline competitor, using extensive simulations varying network topological characteristics and energy distribution patterns in synthetic multi-layer networks. $CCEDS$ was, by a wide margin, the best performing algorithm in terms of EDS cardinality, layer-based resilience, and energy composition in (almost) all cases.

CHAPTER 7

A Rich-Dictionary Markov Predictor for Vehicular Trajectory Forecasting

7.1 Introduction

Vehicular ad hoc networks (VANETs), are spontaneous, flexible wireless networks that are able to support the associated applications in dynamic, multi-hop topologies. However, the relatively high speed of the moving vehicles degrades link quality, causes fast fading, short connectivity and high frequency hand-offs. In general, such mobility related problems are addressed by appropriate broadcasting techniques, by smart routing, by clustering protocols [23]. Nevertheless, as Kolios et al. explain in [72], if mechanical relaying; i.e., "store-and-carry" is allowed for the vehicles, then "*. . . a plethora of different, novel resource-utilization schemes can be explored to increase network performance*". Evidently, one of the key components for achieving relaying is the ability to predict vehicles trajectories accurately.

It has been observed [112] that in practice, weekdays and weekends usually exhibit significantly different traffic conditions, whilst at the same time having similar congested and congestion free traffic patterns. Based on that observation it is straightforward to conclude that every vehicle does not follow the same path every time they leave their base, e.g., house. These different moving patterns relate to time of day such as driving to work in the morning and hobbies in the evening and also the entry point in the road network, which probably means a different final location. In such a realistic situation, where the actual path of each vehicle is not known in

Related publication **[C2]:** Dimitrios Papakostas, Dimitrios Katsaros. "*A Rich Dictionary Markov Predictor for Vehicular Trajectory Forecasting*", **Proceedings of the 30th International Conference on Tools with Artificial Intelligence (ICTAI)**, IEEE Press, Volos, Greece, November 5-7, 2018.

advance and most of the vehicles enter some areas of the city, e.g., city centre, on the same time period, the existence of a fast and accurate prediction mechanism is beneficial to:

- Routing protocols; i.e., the selection of the next hop is a necessary ingredient of store-carry-forward algorithms [70], and also for geocasting protocols [64].
- Traffic management: traffic management applications focus on improving the vehicle traffic flow and traffic assistance. Possible converging vehicle paths might provide drivers useful information so that they can make the best decisions in terms of their route, such as avoiding congested areas.
- Connectivity robustness: user applications which provide value added services like the Internet, and p2p applications, would exploit proposed (predicted) paths that would guarantee an acceptable Quality of Service (QoS) for these applications.
- Safety: drivers are proactively informed about possible conflicting paths between neighboring /approaching cars.

In this chapter, we propose the RDM predictor, a new next-location prediction scheme. The chapter makes the following contributions:

- It exploits the resource-rich environment (battery, computing power and storage) of a vehicle to build a rich summary of its roaming history that subsequently is used to provide more accurate predictions.
- It uses data structures that are constructed in a purely distributed fashion.
- It develops a new forecasting model that is a combination of two prediction mechanisms.
- The proposed algorithm is fully parameterized, presenting different trade-offs in efficiency vs. prediction accuracy.
- It provides a comparison of the proposed method against several, model independent and highly accurate prediction algorithms, and the results show that RDM achieves on the average:
  - More that 35% better prediction accuracy than the second best-performing algorithm.
  - Competitive to faster prediction times than its competitors.

The rest of this chapter is organized as follows: First, we provide the technical insight of the prediction algorithms and explain how the route prediction problem can be modelled as a discrete symbol prediction problem (§ 7.2). Then, we briefly survey related works (§ 7.3), we unveil the unique characteristics of RDM (§ 7.4), we present its prediction mechanism (§ 7.5) and evaluate its performance (§ 7.6). Finally, we conclude the chapter (§ 7.7).

7.2 Technical insight of prediction algorithms

Consider a sequence of position updates $\{x_1, x_2, \ldots x_i\}$ being generated by a vehicle, represented by the stochastic process $X = \{x_i\}$. A predictor will have to predict what the next position x_{i+1} is going to be on the basis of the observed history while minimizing the prediction errors over the course of an entire sequence. Authors in [44] proved the existence of universal predictors that could optimally predict the next item in any deterministic sequence and argued that an optimal predictor must belong to the set of all possible finite state machines. They also showed that universal FS predictors achieve the best possible sequential prediction that any FSM can make and that Markov predictors, a subclass of FS predictors, perform as well as any finite state machine. Moreover, they highlighted that a Markov predictor whose order grows with the number of symbols in the input sequence attains optimal predictability faster than a predictor with a fixed Markov order. Finally, they proved that Markov predictors based on the $LZ78$ incremental parsing algorithm attain optimal predictability because they achieve to changing the Markov order rapidly enough to reach a high order of Markov predictability and slowly enough to gather sufficient information at each order of the model to reflect the model's true nature.

7.2.1 Markov predictors

To provide the formal definition of the prediction model we use in our work we follow the work in [66]; a trajectory a_i of a vehicle i is a finite sequence of symbols a_i^j drawn from an alphabet Σ (where $a_i^j \in \Sigma$, $\forall\ i,j$), with each symbol a_i^j standing for a road-segmID. A predictor accumulates sequences of the type $a_i = a_i^1, a_i^2, \ldots a_i^{n_i}$, where n_i denotes the number of symbols constituting a_i. Without loss of generality, we can assume that all the knowledge of the predictor consists of a single sequence $a = a^1, a^2, \ldots a^{n_i}$. Based on a, the predictor's goal is to construct a model that assigns probabilities for any future outcome given "some" past. As it is stated in [66], this formulation implies a stochastic process $(X_t)_{t \in N}$ where at any given time instance t (meaning that t symbols $x_t, x_{t--1}, \ldots x_1$ have appeared, in reverse order), we need to calculate the conditional probability :

$$\bar{P}[X_{t+1} = x_{t+1} | X_t = x_t, X_{t-1} = x_{t-1}, \ldots], \tag{7.1}$$

where $x_i \in \Sigma, \forall\ x_{t+1} \in\ \Sigma$.

Predictors that use this kind of prediction model are termed higher-order Markov predictors and the "history" $x_t, x_{t-1}, \ldots$ used in the above definition is called the context of the predictor. These predictors maintain a set of relative frequency counts for the symbols seen at different contexts in the sequence, thereby extracting the sequence's inherent pattern. They then use these counts to generate a posterior probability distribution for predicting the next symbol to come.

7.2.2 The vehicle route prediction problem as a discrete symbol prediction problem

In our work we model the road network as a directed graph $G = (V, E)$, where the set V represents the road segments and the set E represents the directed connectivity links between pairs of road segments. Each segment has its own identification number, namely, road-segmID (Figure 7.1).

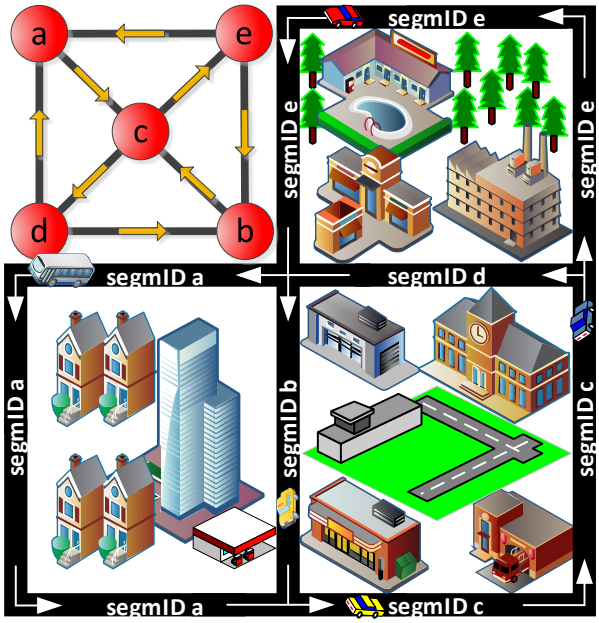

FIGURE 7.1. A sample road network and its corresponding graph; the arrows in the road network and graph depict permitted direction of movement. SegmIDs are represented by nodes.

The segmentation of a road network into segments is a process similar in principle to that of designing location areas for a cellular network [19] [108]. Vehicles are assumed to perform random walks over this network and is also taken that each vehicle is aware of the road-segmID that is moving on, by the use of technologies such as GPS. These position updates, not only are they used to build the moving history of each vehicle but in the form of a discrete symbol sequence they are shared between communicating vehicles in order to support their prediction process.

7.3 Related work

The backbone of the LZ family of prediction algorithms is LZ78 [146]. LZ78 performs incremental parsing of an input string "$x_1, x_2 \ldots x_i$" into *c(i)* substrings (i.e., phrases) "$w_1, w_2 \ldots w_{c(i)}$", such that $\forall j > 0$, the prefix of the substring w_j (i.e., all but the last character of w_j) is equal to some w_i for $1 < i < j$. Because of this prefix property, parsed substrings and their relative frequency counts can be maintained efficiently in a multiway tree structure, namely, a digital tree or prefix

tree (commonly known as trie). An LZ78 parsing of the string “aaababbbbbaabccddcbaaaa” yields the phrases: “a”, “aa”, “b”, “ab”, “bb”, “bba”, “abc”, “c”, “d”, “dc”, “ba”, “aaa”. $LZ78$ maintains statistics for all contexts seen, for example, the context “a” occurs five times (at the beginning of the phrases “a”, “aa”, “ab”, “abc”, “aaa”), the context “bb” is seen two times (“bb”, “bba”), etc.

$LZ78$ has three main drawbacks:

- In any $LZ78$ parsing of an input string, all the information crossing phrase boundaries is lost. In our example string, the fourth symbol “b” and fifth and sixth symbols “ab” form separate phrases; had they not been split, LZ78 would have found the phrase “bab”, thereby creating a larger context for prediction.
- Phrases contained within substrings are also lost;
- The prediction performance of LZ78, is not good for short sequences [44] [66] [115].

LeZi Update (LZU) [15] makes the same parsing of $LZ78$ algorithm, but instead of adding just the substrings resulting from this parsing, it adds also all the suffixes of each substring to the LZU trie. Therefore, phrases within substrings are taken into account. In our example string the phrase “bc” is added in the LZU dictionary, which is a suffix of the phrase “abc”.

The algorithm proposed by Gopalratnam [51], namely $ActiveLeZi(ALZ)$ is intended to consider the phrases among consecutive parsed substrings, thus solving the remaining problem of $LZ78$ algorithm and converging faster to optimal predictability. In order to achieve this, ALZ incorporates a window of variable length, which is determined on the fly, without any extra computational overhead, by the longest phrase parsed by $LZ78$ algorithm at each step. Once the length of the window is updated and a new symbol is added to it, all the suffixes of the window are added to the trie. The detailed performance evaluation of the major Markov predictors in [66] highlighted their shortcomings and suggested routes for their improvement.

Authors in [98] employ the ALZ algorithm to construct frequency trees of a fixed depth k^{max} and perform a Modified Prediction by Partial Match (MPPM) technique in order to obtain a prediction performance that outperforms the best single model predictor. Since the prediction process is online, $MPPM$ utilizes the true data to adaptively weigh the prediction performance of each different order Markov model. The weights are updated at every step and the best performing Markov model is given the highest weight. Thus the problem of finding the best model order for a given sequence length is also implicitly solved by applying the above technique.

Trajectory prediction in $VANETs$ has attracted a significant amount of research recently [110], in order to cope with increased safety issues that arise from the development of autonomous driving technologies [9]. Some recent works deal with lane change prediction [58] [60]; others perform whole trajectory matching with the aim of predicting far-in-the-future positions of a mobile [126, 137]. On the other hand, RDM is a fast, online, next-site prediction model based on analyzing local movement patterns from the recent past.

7.4 The Rich Dictionary Markov (RDM) Predictor

The RDM predictor is designed for the $VANET$ environment, thus assuming significant energy resources, strong computing power and large storage capacity, by following the suggestions in [66]. Therefore, we enriched the $RDM's$ dictionary, we expanded its trie and intentionally developed a more computation-hungry prediction method.

Algorithm 7.1: RDM

precondition : An input sequence
postcondition : An updated dictionary of parsed phrases
remarks : $dictionary$ = stores the parsed phrases, $window$ = a variable length window of previously seen symbols, Max_LZ_Length : the length of the longest parsed phrase, w = a continuously updated phrase that drives the dictionary construction.
initialize : dictionary = null; window = null; Max_LZ_Length = 0;

```
1  Loop
2      Wait for next symbol v;
3      if w.v in dictionary then
4          w = w.v;
5      else
6          add w.v in dictionary;
7          update Max_LZ_Length if neccessary;
8          w = null;
9      end
10     add v to window;
11     if length(window) > Max_LZ_Length then
12         delete window[0];
13     end
14     Update dictionary with all possible prefixes within window that include v;
15 Forever
```

7.4.1 Rich Dictionary Construction

What differentiates RDM from both ALZ and $MPPM$ is that while they all parse both the input sequence and the phrase that resides in the sliding window, only RDM attaches all these phrases to the dictionary as part of the protocol. The net effect of this procedure is that the algorithms develop completely different dictionaries, tries, sliding window sizes, which altogether affect the prediction accuracy. Algorithm 7.1 presents the pseudocode for parsing and processing the input sequence in RDM.

Proposition 7.1. *When the length of the longest phrase parsed by ALZ, RDM and MPPM is less than k^{max} MPPM constructs frequency trees that grow faster.*

Proof. While ALZ and RDM incorporate a window of variable length, $MPPM$ uses a window of fixed length k^{max}. This modification on the one hand enables $MPPM$ to control the growth of its associated trie (max trie depth = k^{max}) on the other hand allows the phrase that resides in the sliding window (and drives the associated trie construction) to continuously increase its length until it is k^{max}. Therefore, $MPPM$ continuously adds larger phrases in its trie while RDM and ALZ only when a new phrase is entered in each algorithms' dictionary. ⊣

Figure 7.2 illustrates the tries that the competing algorithms build (k^{max} = 5 for $MPPM$) by superimposing them into a single trie for the input sequence "aaababbbbbaabccddcbaaaa". We see that $MPPM$ builds the largest trie, however, this is temporal and will stop from happening when the length of the longest phrase parsed by ALZ and RDM becomes greater than k^{max}. In practice $RDM's$ resultant dictionary is richer and the associated trie larger both in span and depth.

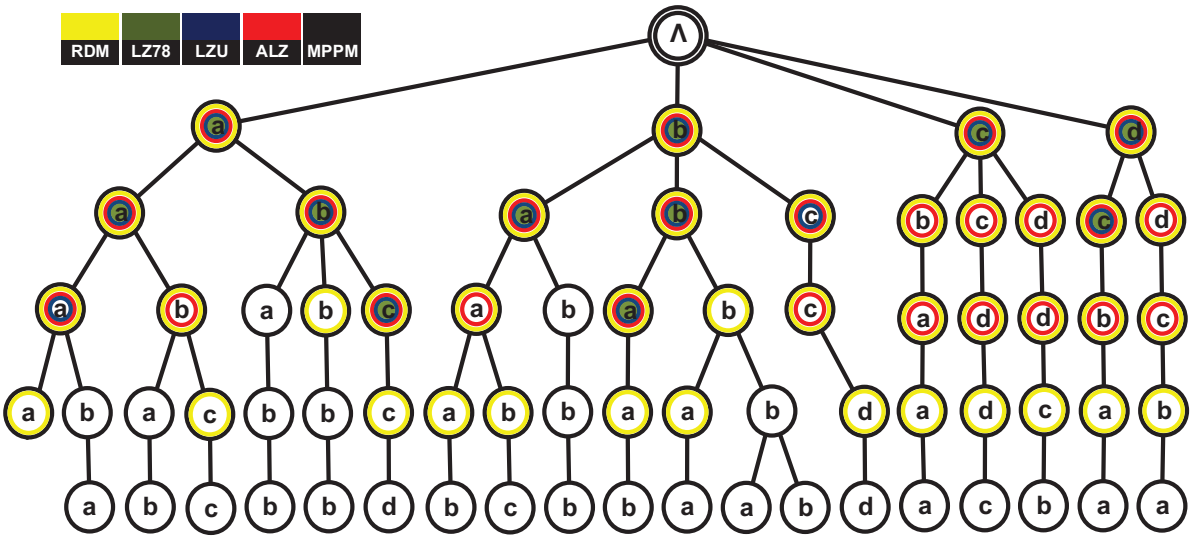

FIGURE 7.2. The trie formed by RDM (Yellow) / LZ78 (Green) / LZU (Blue) / ALZ(Red) / MPPM (Black) after parsing the symbol sequence "aaababbbbbaabccddcbaaaa".

Proposition 7.2. *The trie developed by ALZ is strictly contained within that created by RDM.*

Proof. RDM adds into the dictionary the parsed phrases of the sliding window and thus, increases the size of the sliding window earlier than ALZ (more precisely it updates earlier the Max_LZ_Length parameter). This larger window, whilst incorporates at any time $ALZ's$ equivalent window, produces more (trie span expansion) and longer phrases (trie depth expansion). ⊣

7.4.2 RDM Complexity Analysis

RDM consumes space and time for data processing. The following analysis discusses the worst-case conditions for the prediction process.

Proposition 7.3. *The space complexity of RDM is $O(n^{\frac{3}{2}})$.*

Proof. At each step, RDM parses the symbol string within the sliding window and either updates the cardinality of an existing node or adds a new node to the trie. The worst possible case arises when RDM increases the maximum LZ phrase length and the parsed substrings add new nodes to the trie. We represent the sequence of the parsed substrings as $\hat{g} = x_1, x_1x_2, \ldots, x_1x_2\ldots x_k, where\ |\hat{g}| = n = \frac{k(k+1)}{2}$. A sequence of this form can be represented by an order-k Markov model which stays of order-k through the next k symbols. In the worst case, each parsed substring adds a new node to the trie, so at order k, the trie gains k^2 nodes before the model transitions to order k + 1. Therefore, the number of nodes generated in the trie by the time the model attains order k is $O(k^3) = O(n^{\frac{3}{2}})$, because k = $O(\sqrt{n})$. ⊣

Proposition 7.4. *The time complexity of RDM is $O(n^{\frac{3}{2}})$.*

Proof. The worst case in terms of runtime arises when RDM parses the worst sequence $\hat{g}$ because it will prompt the most updates. Creating or updating a node in the trie requires finding the appropriate child of a given node along the path that phrase traced in the trie. RDM can access a given node's child in constant time. Therefore, it can find a node in time linear in the depth of the trie. When the worst-case sequence $\hat{g}$ is the case where $|\hat{g}| = n$ and order k = $O(\sqrt{n})$ there must be an update for every order up to k before the model transitions to order k + 1. Consequently, by the time the model attains order k and the number of nodes generated in RDM's trie is $O(n^{\frac{3}{2}})$, the runtime is also $O(n^{\frac{3}{2}})$. ⊣

7.5 RDM prediction mechanism

RDM employs two prediction mechanisms that are used simultaneously and are complementary to each other. The first is purely probabilistic and is similar to that in [15], while the second is deterministic, being based on trie traverse in order to make predictions. The first mechanism considers different order Markov models and employs a blending strategy known as exclusion [15] to build a probability distribution for each state (symbol) occurring in the input string. When used it predicts the state with the highest probability value as the most likely action. We explain how the first mechanism works by referring to Figure 7.3, which represents the trie constructed by RDM for the sequence "aaababbbbbaabccddcbaaaa".

The window maintained by the predictor represents the set of contexts RDM uses to compute the probability of the next symbol. In our example, the last phrase "aaaa" (which is also the current RDM window) is used. Within this phrase, the contexts that can be used are all suffixes

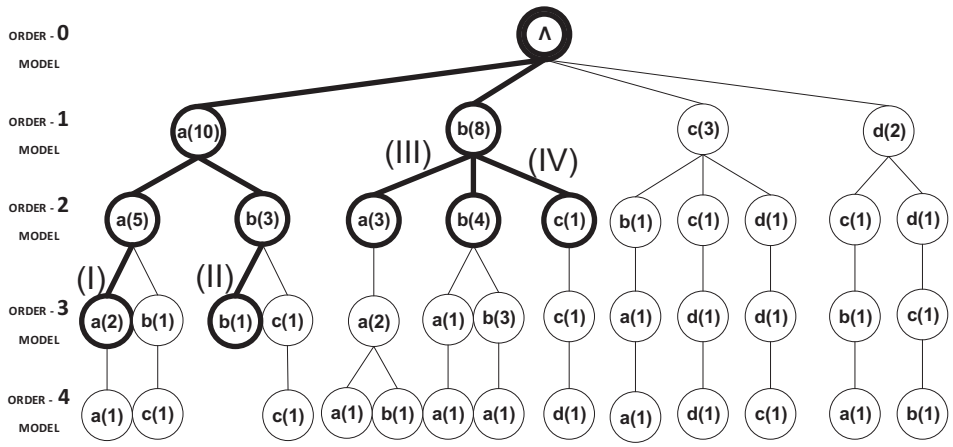

FIGURE 7.3. Different cases (I, II, III, IV) w.r.t the symbol prediction process. The numbers adjacent to each node represent its cardinality in the trie

within the phrase, except for the window itself (i.e., "aaa", "aa", "a" and the null context). Suppose we want to calculate the probability that the next symbol is an "a". We see that an "a" occurs one out of two times that the context "aaa" appears. Therefore, $P(a|aaa) = \frac{1}{2}$. Then we fall back to the order-2 context (i.e., the next lower order model) with probability $P_{esc(2)} = \frac{1}{2}$; the so-called escape factor of the order-2 model, which corresponds to the probability that the outcome is null. At this level, we see that it occurs one "a", one "aa" and one "bc" out of five times that the context "aa" appears, therefore $P(a|aa) = \frac{1}{5}$. Then the algorithm falls back (escapes) to the order-1 model with probability $P_{esc(1)} = \frac{(5-(2+1))}{5} = \frac{2}{5}$. By using the same technique at the order-1 context, we see that an "a" occurs two out of ten times that the "a" appears (the rest belonging to bigger phrases, e.g., "aa", "ab", "aba" etc.) and therefore we have $P(a|a) = \frac{2}{10}$. Then, for the last time, the algorithm falls back (escapes) to the order-0 model with probability $P_{esc(0)} = \frac{2}{10}$ or $\frac{1}{5}$. At that level, we see an "a" two times out of the 23 symbols seen so far, and therefore we predict "a" with probability $P(a|\bigwedge) = \frac{2}{23}$ in the null context. Therefore, in our example, the blended probability of seeing an "a" as the next symbol is:

$$\frac{1}{2} + \frac{1}{2}\{\frac{1}{5} + \frac{2}{5}[\frac{2}{10} + \frac{2}{10}(\frac{2}{23})]\}. \tag{7.2}$$

The second prediction mechanism employs a purely deterministic approach to make predictions. It uses the continuously updated LZ phrase "w" that drives the RDM dictionary construction (see the RDM' pseudo-code) to traverse the trie and pinpoint to the current state of the predictor. Note that at each state we record the prediction hits of the predictor regarding that state in the past. RDM exploits the information (history) given from the higher order model and predicts according to the following rules (see Figure 7.3):

- Case (I): Single branch at the higher order model → Predict the context given by the higher order model. For example, when w = "aaa", the predictor exploits the information given from the 4th order model and predicts "a" as the next symbol.
- Case (II): No higher order model → Use the probabilistic prediction mechanism and predict the context with the largest probability value, e.g., when w = "abb".

When more than one branches exist under the current state of the predictor, RDM calculates the Kendall Tau Rank Distance (KTRD) between the ranking lists of the cardinalities and the prediction hits between the states of the higher order model and uses it as a yardstick for the prediction of the next state. In order to present how RDM works in such cases we will refer in Figure 7.3 to the case where w = "b", and we will assume that the prediction hits regarding the states "a", "b" and "c" of the 2nd order model are 2, 2 and 1, respectively. Thus, when w = "b", RDM ranks the states of the higher order model with regards to their cardinality and prediction hits as follows:

State	a	b	c
Rank by Cardinality	2	1	3
Rank by Prediction Hit	1	1	3

Then, RDM pairs each state with every other state and counts the number of times the values in the two lists are not in the same order:

Pair	Cardinality	Prediction Hit	Count
(a,b)	2 > 1	1 = 1	X
(a,c)	2 < 3	1 < 3	
(b,c)	1 < 3	1 < 3	

The calculation of $KTRD$ is a simple process which is based on a merge sort algorithm and requires time $O(n \log n)$. If n is the list size, the normalized $KTRD$ is :

$$KTRD = \frac{Discordant\ pairs}{n * (n-1)/2} = \frac{1}{3 * (3-1)/2} = 0.33. \tag{7.3}$$

Then RDM uses the first pair of branches it finds (from left to right) at the higher order model and calculates the difference between their cardinalities. Then, the absolute value of the result is divided by the cardinality of the current status of the predictor (we term it here $RESULT$) and is compared with the $KTRD$. Note that when the higher order model consists of more than two states then Case III and/or Case IV scenarios continue to be executed until all states have been examined:

- Case (III): $RESULT \leq KTRD \rightarrow$ Use the probabilistic prediction mechanism and predict the context with the largest probability value. For example, when w = "b", the first pair of states we examine in the order-2 model is "a", "b". Thus, we have $RESULT = \frac{|a(3)-b(4)|}{b(8)} = \frac{|3-4|}{8} = 0.125$. Since $RESULT < KTRD$ the state with the largest probability value between the two is predicted.

- Case (IV): $RESULT > KTRD \rightarrow$ Predict the context with the larger cardinality. For example, when w = "b", if the pair of states under consideration is "b" and "c", then we have $RESULT = \frac{|b(4)-c(1)|}{b(8)} = \frac{|4-1|}{8} = 0.375$. Since $RESULT > KTRD$ it is predicted the state "b" because it has the largest cardinality.

Practically, the use of $KTRD$ enables RDM to speed up the prediction process; it predicts the state with the larger relative cardinality among the states of the higher order model as long as the respective number of discordant pairs between the ranking lists of the cardinalities and the prediction hits remains small.

7.6 RDM Performance Evaluation

We perform a simulation-based evaluation of the performance of RDM. To this end, we designed experiments with vehicle itineraries that are segmented and are represented by datasets with discrete symbol sequences that vary with respect to the number and repeatability of patterns within them.

7.6.1 Simulation setting

The next paragraphs describe the competing algorithms, the datasets, and the performance measures used.

7.6.1.1 Methods compared

We use the $LZ78$ [146] compression algorithm as the baseline algorithm for our comparisons due to its simplicity. We implement and evaluate also LZU [15], ALZ [51] and $MPPM$ [44] since the superiority of Markovian predictors over other techniques has been explained in [66].

7.6.1.2 Datasets used

Since there are no publicly available datasets containing real vehicle movements over segmented city roads, we created five datasets in order to evaluate the performance of RDM. Four simulate vehicle itineraries over a road graph with and without noise, and the fifth is a realistic dataset produced using SUMO an open traffic simulation suite.

The first dataset, named s4n0, consists of an alphabet of four different symbols (s4) and contains no noise (n0). It has perfect regularity in terms of the vehicle patterns and a small alphabet (few road-segmIDs) that creates conditions for all the algorithms to have good prediction performance. In the second dataset, named s4n20, we deliberately disrupted the patterns to a percentage of 20%. The third dataset was created by using 20 symbols and it is polluted with 10% noise (s20n10), whereas the fourth is similar to the third, but with 30% noise (s20n30). The realistic dataset , named VolosItineraries, includes the mobility of 210 vehicles travelling in the road network of the city of Volos(Greece) with different mobility patterns. The traffic simulations are conducted with SUMO and the trace files are injected into our custom simulator in order to perform prediction. Vehicles follow one of three different predefined routes, having a random velocity with a mean value of 11 m/s and a variation of 5 m/s. By using big variation in vehicle

velocities and by recording the position of each vehicle every T seconds (by default 5 seconds), we reassured that the final recorded trace for each vehicle is different from any other even if they follow the same path. Thus, each vehicle trace may contain repeating road segments representing along with the transition from one road segment to another, the staying on a road segment due to traffic congestion, road length and maximum velocity limit. The resulting alphabet created from the realistic dataset consists of several dozen different symbols. The total simulation time is one hour.

7.6.1.3 Performance measures

We use three measures to quantify performance. The first, is the prediction accuracy, which represents the percentage of correct predictions per 1,000 symbols of the input sequence. The second, is the processing time (milliseconds), in the form of the maximum time needed (worst case scenario) for a single prediction in a 1,000 symbol subsequence of the input sequence; it portrays the applicability of each competing algorithm in a real world Intelligent Transportation System (ITS). The third is the number of trie nodes entered into the trie per 1000 symbols of the input sequence; it is an abstract measure of the memory footprint independent of any implementation.

7.6.1.4 Location update techniques

We update the movement history of a vehicle whenever it crosses the boundaries of a new road segment (with the use of the GPS technology) and every T seconds (by default 5 seconds). With our method:

- We ensure that all distinct road segmIDs will be recorded; missed road segmIDs effect on movement history is like noise in the symbol sequence, it disrupts the continuity of the symbols and affects negatively the repetition of the symbol strings.
- We differentiate each vehicle's movement patterns based on its habitual duration of stay in a road segmID.
- We control the amount of information entered in the system.

7.6.2 Evaluation of the results

The simulations were run on a PC with Intel core 2 duo 1.7 MHz CPU, 2GB main memory, 80GB hard disk 7200 rpm hard disk and MSWindows 7 64bit. The codes of the competing algorithms were compiled in Matlab R2015a. On the other hand, a typical communications box supporting Dedicated Short-Range Communications (DSRC), such as those commercialized by DELPHI runs on an x86 architecture Intel core 2 duo 2 GHz CPU, with 2GB onboard DDR2 RAM, and onboard 8 GB Solid State Disk. Therefore, our algorithms can run on an industrial onboard unit.

7.6.2.1 Tuning the RDM

We investigated the impact of blending the models of all orders and using different inter-record time settings on $RDM's$ performance.

The depth factor parameter It enables RDM to exploit only a limited number of lower-order models during the blending strategy [15]. Practically, when excluding some of the lower-order models we bias the system to predict faster. However, we expect that happening at no cost to the $RDM's$ performance, because the excluded models' impact on the final probability assignment is suppressed due to the escape factor. In Figure 7.4 we observe that RDM exhibits best performance in terms of prediction accuracy when we set the depthFactor equal to 1. A setting equal to 1 means that during the probability calculations we use only the order model "in which" the predictor currently lies and the immediate previous one (larger values for this parameter imply exploitation of smaller order models). In general, when depthFactor> 1 the performance remains (almost) the same, as observed also in [115], however when a performance degradation exists it is due to past vehicle mobility patterns (not concerning the present vehicle movement) that are taken into account in the probability calculations.

	Depth Factor										
Dataset	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
s4n0	0.9834	0.9835	0.9832	0.9831	0.9831	0.9831	0.9831	0.9831	0.9831	0.9831	0.9831
s4n20	0.9433	0.9433	0.9431	0.9431	0.9431	0.9431	0.9431	0.9431	0.9431	0.9431	0.9431
s20n10	0.7181	0.7184	0.7184	0.7184	0.7184	0.7184	0.7184	0.7184	0.7184	0.7184	0.7184
s20n30	0.3650	0.3650	0.3645	0.3613	0.3613	0.3613	0.3613	0.3613	0.3613	0.3613	0.3613
VolosItin	0.8466	0.8466	0.8466	0.8466	0.8466	0.8466	0.8466	0.8466	0.8466	0.8466	0.8466

FIGURE 7.4. Impact of depthFactor parameter on RDM's prediction accuracy.

The inter-record time - T It influences the frequency of the prediction process. We conducted several experiments w.r.t the VolosItineraries dataset to explore the impact of various inter-record time settings on the performance of RDM and the results are presented in Figure 7.5.

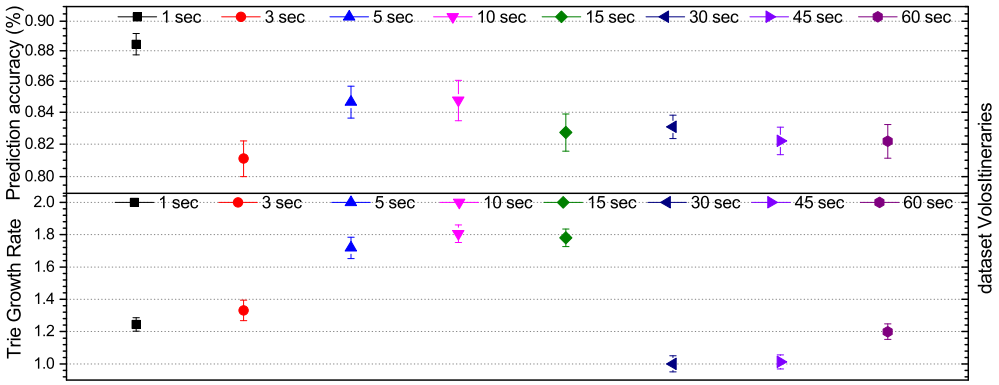

FIGURE 7.5. Impact of inter-record Time on $RDM's$ performance.

We set T = 5 seconds as a tradeoff between $RDM's$ prediction accuracy performance (Figure 7.5 top plot) and the growth rate of its trie (Figure 7.5 bottom plot). We avoid using settings

of T < 5 seconds and T > 10 seconds because they miss to provide the predictions in a timely manner with each road segment change (which is a requirement for an online predictor). Note that the respective performance of RDM when T = 1 second is deceptive as it is a consequence of the plethora of symbols that enter in the system in short time. A setting of T = 10 seconds is also avoided as it leads to a larger trie (see Figure 7.5 bottom plot). While a setting of T = 5 sec builds a trie which is almost 1.7 times the respective trie when T = 30 seconds, this is acceptable to be happening given the resource-rich $VANET$ environment that RDM is planned to work.

7.6.2.2 Comparison of the competing algorithms

The top plot in Figure 7.6 depicts the performance of the competitors regarding their prediction accuracy for the first dataset (s4n0); RDM achieves almost 99% accuracy on the average whereas its competitors have inferior performance and they stagnate to a prediction accuracy of 70% at most. It is expected that all algorithms achieve fast convergence and very good performance because the alphabet is small and the trajectories are noiseless (existence of very few and strong patterns). The bottom plot in Figure 7.6 shows the performance of the algorithms regarding their prediction accuracy for the second dataset (s4n20), which is noisier. RDM still exhibits very good performance (around 95% on the average and a maximum of 98.5%), however, now it is able to achieve more than 90% prediction accuracy only after consuming 4,000 symbols (contrast this to a 95% prediction accuracy after consuming only 2,000 symbols for the previous plot) because of the noise. The other algorithms' performance is around 45% (on the average). Therefore, the rich dictionary of RDM and its new prediction mechanism make RDM to be always better than its competitors, and the performance gap widens (from 50% to 120% on the average) when noise is introduced.

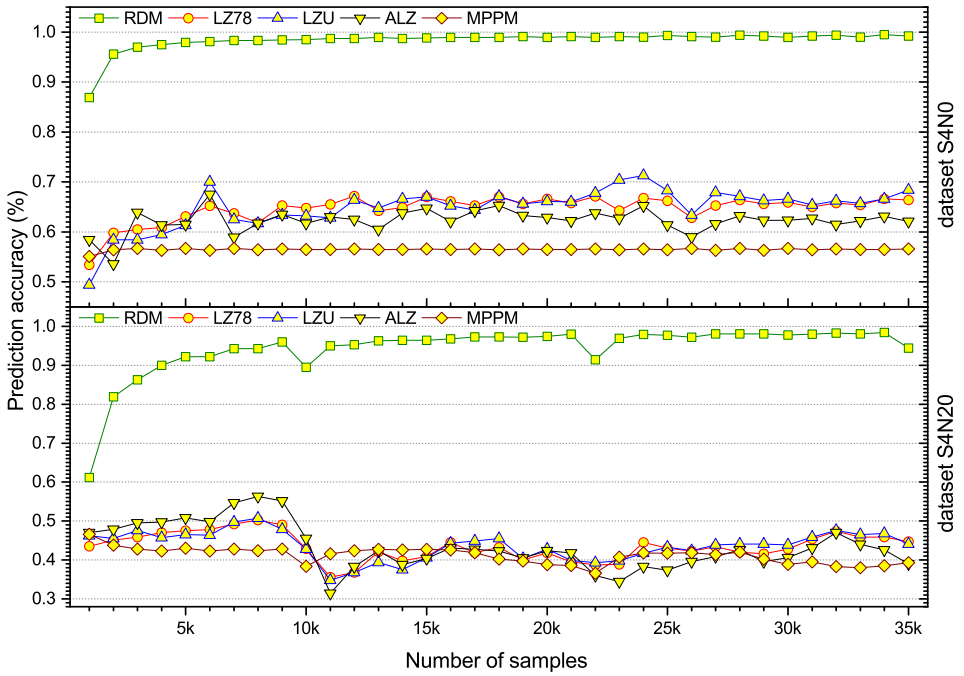

FIGURE 7.6. Prediction accuracy (%) for small alphabets.

In the next experiment we increase the size of the alphabet, and evaluate the competitors under moderate (s20n10) and heavier (s20n30) noise. The results are illustrated in the two plots of Figure 7.7. We expect that all algorithms' performance will degrade significantly because now the alphabet is larger. Indeed, RDM achieves 72% and 37% performance for these datasets (contrast this to 99% and 95% accuracy in the previous two plots). Moreover, the distortion of the patterns due to the introduction of noise decisively affects $RDM's$ performance, i.e., the prediction accuracy drops from 72% to 37% for s20n10 and s20n30, respectively. This drop in performance happens because $RDM's$ deterministic prediction model cannot be widely exploited and thus it consults mainly the probabilistic model. However, RDM maintains the relative performance gap with its competitors (50% - 60% better). As expected, the performance gap between RDM and its competitors can not be as high as before, due a) to the larger alphabet, and b) to higher noise percentage (10% and 30% versus 0% and 20%), which collectively destroys the repetitive movement patterns.

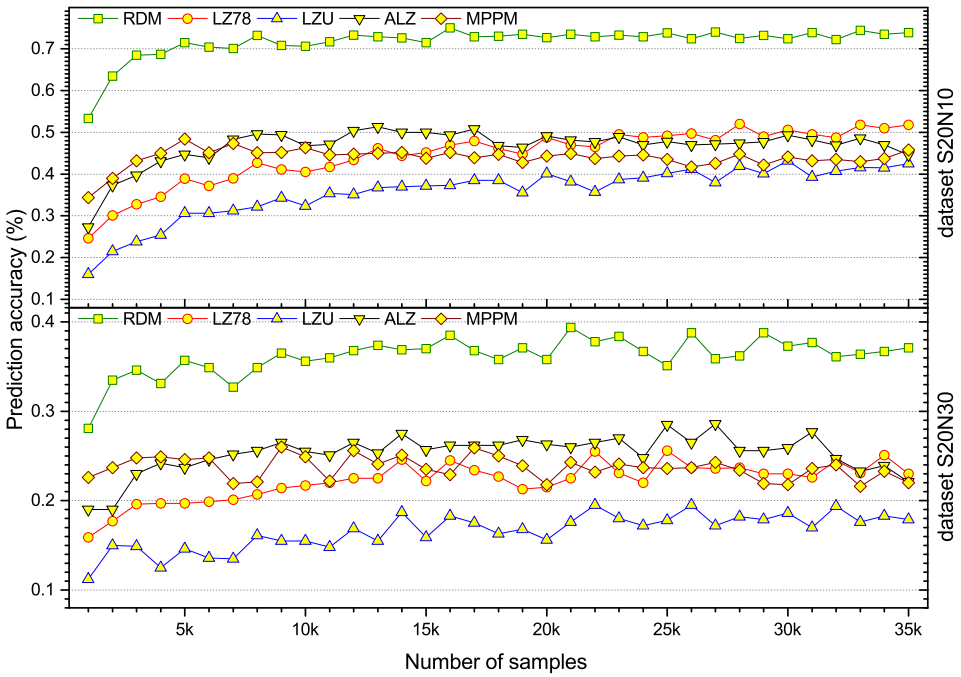

FIGURE 7.7. Prediction accuracy (%) for larger and noisier alphabets.

Figure 7.8 (top plot) depicts the performance of the competitors regarding their prediction accuracy for the VolosItineraries dataset. RDM if the best performing argorithm; it converges to a prediction accuracy of 80% after having consumed only 2.000 symbols and achieves a mean prediction accuracy of 85% and a maximum of 90%. The performance of the other competitors is an increasing function with respect to the number of symbols that enter in the system. However they need to consume at least 5.000 symbols in order to converge to a prediction accuracy of 50%. $MPPM$ is the second best performing algorithm with a mean prediction accuracy of 63% and a maximum of 76%. The rest of the competitors present a prediction accuracy of 57%, on the

average. Therefore this realistic case reaffirms the earlier results, however now the performance gap with the second best performing algorithm ($MPPM$) narrows to 35%.In sum, we have shown in all the plots that RDM can achieve high prediction accuracy.

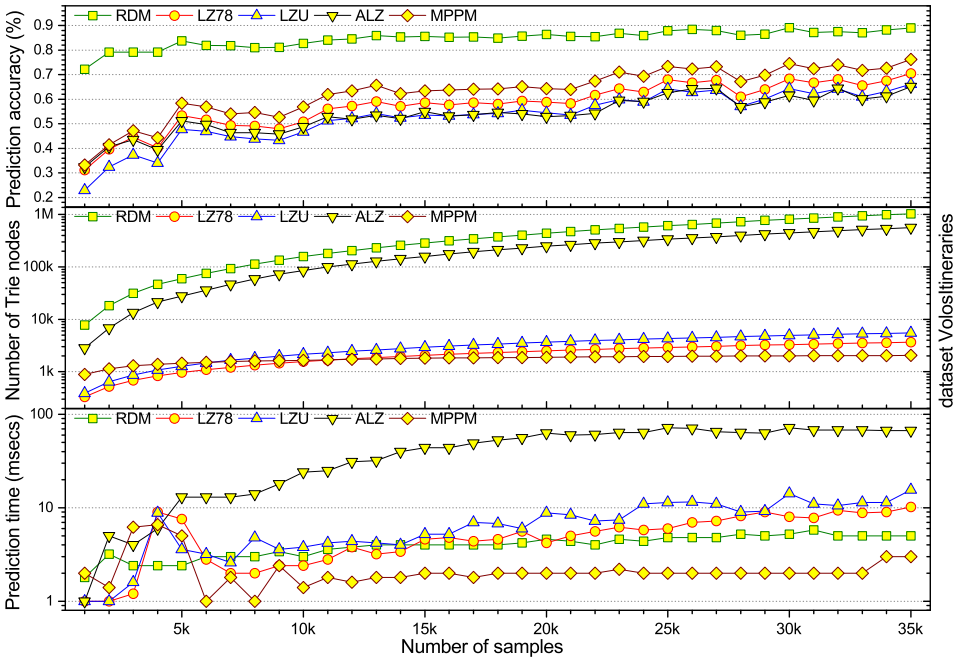

FIGURE 7.8. Competing algorithms efficiency for the VolosItineraries dataset.

The middle plot of Figure 7.8 depicts the memory footprint of the algorithms in the form of node count evolution in each respective trie as each symbol of the VolosItineraries dataset is processed. It can be seen that the pace of trie growth is logarithmic with respect to the sequence size for all the competitors except for $MPPM$ where its trie stops growing after the parsing of 10,000 symbols as a consequence of the fixed depth frequency tree of $k^{max} = 5$, it constructs. As expected, RDM constructs the largest trie which was our initial goal so as to achieve improved prediction accuracy. $MPPM$ constructs the smallest trie followed by $LZ78$ and LZU because of the small dictionary they have. It is noteworthy that the $RDM's$ trie is the double size the respective $RDM's$ trie and two orders of magnitude larger than the $LZU's$ and $LZ78's$ respective tries. Moreover, RDM and ALZ trie grows much faster, whereas LZU trie size also starts growing quickly but stop increasing so fast at a lower level. It is noteworthy that the large trie that RDM produces does not lead to significant communication overhead. In an operational $VANET$ environment a single packet can accommodate a large number of trie nodes, and in any case, if two vehicles need to exchange their mobility profile, they will not exchange the full trie, but rather only the part of it (a few branches) that correspond to the particular area where they are both moving at that particular time.

The bottom plot of Figure 7.8 depicts the per-symbol maximum prediction time needed (worst case in a 1,000 symbol sequence) for the VolosItineraries dataset. Intuitively someone would

expect the largest dictionary be the most time-consuming to process, however RDM succeeds in being faster than the majority of its competitors and competitive to $MPPM$, because of the hybrid mechanism it employs for prediction. It takes 6 milliseconds (worst case) for the RDM to deliver predictions, which is fast enough to support real-time applications, such as vehicular safety applications. On the other hand, $MPPM$ requires the least processing effort which is due to the fixed depth (k^{max} = 5) frequency tree that it constructs.

Overall, RDM is the best performing predictor; it constructs the largest trie, and it is not the slowest algorithm. $MPPM$ in general exhibits similar performance with ALZ except for the realistic dataset (where it is better) thereby confirming the results in [98] and for the s4N0 dataset (where it is worse). ALZ is better than LZ78 in terms of accuracy for large alphabets and input sequences with a small length, thus confirming the results in [51]. It also emerges that ALZ is not significantly better than LZ78 in the other cases, thereby confirming the findings in [101], but contradicting the findings in [51], because we used the exclusion strategy. Moreover, ALZ converges faster than LZ78, which supports the results in [51] and all the predictors' tries are growing at a logarithmic pace (except $MPPM$), which mirrors the behavior reported in [101]. Finally, the creation of RDM addresses a comment that appeared in [115] stating that there is a significant gap that needs be filled by the improvement of online Markov predictors.

7.7 Conclusions

Accurate prediction of vehicular trajectories in a $VANET$ environment is an essential mechanism for $ITSs$. The prediction methods based on Markov predictors are particularly appealing, because of their generality and prediction accuracy. Tailored for the resource-rich $VANET$ environments, the proposed RDM Markov predictor remains highly efficient in terms of prediction accuracy and per-symbol maximum prediction time. RDM is likely to be of immediate interest to $ITSs$ providers. By maintaining global dictionaries along with individual vehicle profile decoded from updates, it will be possible to predict group behavior. In $VANETs$, this can lead to better traffic management, and more efficient bandwidth management and quality of service (QoS) in p2p applications. As future work we plan to bound the growth of the frequency tree that RDM produces and integrate RDM with novel $VANET$ routing protocols that can be used in safety or ecorouting applications for ITS.

CHAPTER 8

A Simulation-based Performance Evaluation of a Randomized MIS-based Clustering Algorithm for Ad Hoc Networks

8.1 Introduction

The present chapter investigates the problem of improving the scalability of a military multilayer ad hoc network. Modern military battlefields consist of an increasing array of entities with wireless communication and sensing capabilities. The coexistence of these wireless entities constitutes the utilization of the wireless medium a very challenging task. We build upon this fact and compare solutions proposed for scaling down ad hoc networks with large numbers of nodes with the goal to highlight their pros and cons.

In the context of ad hoc networks providing a hierarchical organization has been identified as a viable and efficient option for this problem in the literature. A wireless ad hoc network is a type of wireless network which eliminates the complexities of infrastructure setup and administration, by enabling existing nodes to create and join networks *"on the fly"*, anywhere, anytime, for virtually any application. The decentralized nature of wireless ad hoc networks makes them suitable for a variety of applications where central nodes can't be relied on, and may improve the scalability of wireless ad hoc networks, compared to wireless managed networks. The minimal required configuration, the quick deployment, and the presence of dynamic and adaptive routing

Related publication **[J3]:** Dimitrios Papakostas, Dimitrios Katsaros. "*A Simulation-based Performance Evaluation of a Randomized MIS-based Clustering Algorithm for Ad Hoc Networks*", **Simulation Modelling: Practice And Theory (Elsevier)**, vol.48, pp.1-23, November, 2014.

protocols, which allows them to be formed quickly, make ad hoc networks suitable for situations like habitat monitoring, disaster relief, law enforcement operations, battle field communications, target tracking and so on.

The dynamic nature of wireless ad hoc networks though, requires that solutions for multi hop network protocols at all levels must be distributed. Of the solutions proposed for scaling down networks with large numbers of nodes, *network clustering* is among the most investigated for mobile ad hoc networks [6, 7, 10, 11, 26, 49, 74, 118, 125, 135, 144], for sensor ad hoc networks [37, 142], for wireless mesh networks [5] and for vehicular ad hoc networks [82, 88, 127]. Other solutions of network management such as topology control [32] are remotely related to this work.

The basic idea in clustering is that of grouping network nodes that are in physical proximity, thereby providing the *flat* network a *hierarchical* organization, which is smaller in size, and simpler to manage. The subsequent VBN construction uses the induced hierarchy to form a communication infrastructure that is functional in providing desirable properties such as minimizing communication overhead, choosing data aggregation points, increasing the probability of aggregating redundant data, and consequently minimizing the overall power consumption.

In a clumpy approach, clustering algorithms can be divided in two major families. The first with its roots in *graph theory*, exploits the localized network structure for estimating dynamically the "clusterheads" (CHs), while the second provides mechanisms to ameliorate the fact that nodes belonging to the backbone are solely responsible for carrying out all communication, thus running out of energy very soon. Namely, the latter family addresses the energy consumption problem, that is essentially proposes ways to rotate the role of CH among nodes of clusters e.g., the $SPAN$ [27], the $LEACH$ [57] and the $HEED$ [141] protocols. The proposed methods use the residual energy of each node in order to direct its decision about whether it will elect itself as a CH or not. However, this family's methods ignore topological features of the nodes.

The former family of protocols encompasses the most representative and successful solutions in the area of network management. This is because while it exploits the wealth of information extracted from the network topology particularities, it can also take into account application specific requirements, such as Quality of Service (QoS). Moreover, algorithms of this family can easily be combined with a round robin CH rotation method as was described in [37], and thus exploit all the advantages the energy-efficient algorithms provide.

Recently, *RanMIS* [2] [3] – which belongs to the first family – was proposed as a message-optimal algorithm; it employees a probabilistic technique for node clustering and enables every node to independently decide on its role in the clustered network. It ensures rapid convergence, while keeping the message overhead low. Its message complexity on CH selection was studied analytically in [3], but its performance is still in question with respect to the delay it incurs for the clustering protocol termination, for the "stability" of the created clusters in case of node failures, and so on. The focus of this chapter is exactly to fill this gap by conducting a detailed

experimental evaluation of *RanMIS* by simulation.

RanMIS competitors were selected from two distinct protocol categories[1]. The first category is oriented on providing the network with a two-layer hierarchical organization comprised by groups of nodes, i.e., clusters. One "special" node of each cluster (the CH) participates in the so-called VBN; the backbone nodes form a DS over the flat network topology. This means that, each node that is not in the backbone, has at least one backbone node as its neighbor. Backbone nodes are joined via *gateway* nodes. Apparently, if the flat network is connected, it is deterministically guaranteed, that the backbone is connected as well. Distributed Clustering Algorithm (*DCA*) is a high-performance representative of this category.

The second category retains the layered structure of the first category, and it is oriented on facilitating routing without the need of gateway nodes. The concept behind this category is that of building a CDS directly, without firstly selecting CHs and then joining them. It was initially introduced by Das et al. in [34] when they proposed the concept of creating a *communication spine* inside a flat network topology in order to support unicast, multicast, and fault-tolerant routing within an ad hoc network. *WuLi* is a practical and robust representative of this category.

Contrary to *RanMIS*'s randomized nature, its competitors use iterative clustering techniques, meaning that a node waits for a specific event to occur or certain nodes to decide about their role before making a decision themselves.

8.1.1 Motivation and Contributions

The problem of providing a hierarchical organization for ad hoc networks is of crucial importance due to the widespread deployment of such networks (e.g., mobile ad hoc networks, sensor networks, vehicular ad hoc networks) and their scalability problems. Despite the fact that a really large number of algorithms have been proposed during the last decade for performing clustering, the proposal and study of protocols which exhibit low message complexity, fast convergence, incremental backbone maintenance, resilience to hub node failures, connectivity preservation, and backbone stability is still in quest. The considered clustering algorithms seem to encompass all the aforementioned features and therefore their detailed, joint investigation is a significant task for the area of ad hoc network management.

In this context, the present chapter makes the following contributions:

- We investigate for the first time by detailed simulation the performance of *RanMIS*. In particular:
  - We confirm its message optimality for backbone construction across all examined network topologies.

[1]More detailed arguments about the selection of competitors can be found at subsection 8.3.4

  - We reveal the impact of parameter D on the number of rounds before *RanMIS* terminates; moreover, we prove its independence on each node's neighborhood size, contrary to *RanMIS*' original article suggestion.
  - We question the value of parameter M as suggested by the authors of *RanMIS*.
  - We illustrate *RanMIS*'s performance lagging with respect to the number of rounds before termination, which means that it will incur significant delay for clustering.
  - We provide a characterization for the distribution of nodes in clusters for the competing protocols, and show *RanMIS*'s drawback in that it produces 'long' backbones, which implies increased delays for message routing.

- We propose an improvement to *WuLi*, namely *WuLi** appropriate for cluster rebuilding processes in case of node failure.

- We develop a rich comparison framework for the ad hoc network clustering protocols, employing three families of performance measures, namely for protocol cost, for backbone description and for robustness.

- We provide a detailed comparison of the algorithms for these families of measures for many network topologies (with different size and density).

The rest of this chapter is organized as follows: in Section 8.2, we briefly describe the operation of *RanMIS*; in Section 8.3 we firstly describe the operation of *RanMIS*'s competitors and then provide the detailed performance evaluation of *RanMIS* against them; finally, in Section 8.4 we conclude the chapter.

8.2 The Beep-based randomized MIS algorithm

RanMIS [3] considers the problem of computing a MIS, a fundamental distributed computing procedure, that seeks to elect a set of local leaders in a network.

It generates the maximal set of nodes in such a manner that, no two of them are neighbors, by using an extremely harsh broadcast model that relies only on carrier sensing. Since the set is maximal, every node in the network is either in the MIS or a neighbor of a node in the MIS.

The model consists of an anonymous broadcast network in which nodes have no knowledge about the topology of the network. It has its roots to the solution of a similar problem that evolves during the development of the fly's nervous system, when Sensory Organ Precursor (SOP) cells are chosen [3].

According to the specified assumptions, nodes receive as input, an upper bound on the number of nodes in the network n, and an upper bound D, on the number of neighbors any node can have (if no such bound is known, then D is set to n). Furthermore, it is assumed that they all wake up

together at the same synchronous round (and start executing the algorithm), also that they can detect collisions, and finally that no failures occur.

The algorithm proceeds in $log\ D$ phases, each consisting of $M\ log\ n$ steps, where M is a constant. Initially, all nodes are active. Each step in each phase i consists of two exchanges.

In the first exchange, each active node broadcasts a message to its neighbors with probability p_i. Such as in the biological model, the probability p_i increases with i. In the second exchange, a node that has broadcasted a message in the first exchange joins the MIS if none of its neighbors had broadcasted at the first exchange. Such node broadcasts again a message to its neighbors, telling them to become inactive, and exits the algorithm.

For the sake of completeness we give in Figure 8.1 the pseudocode of the *RanMIS* algorithm [3] which is synchronously executed by all nodes .

Algorithm 8.1: RanMIS(n, D) at node u

remarks: n: number of participating nodes in network topology,
u: node under consideration,
D: upper bound on the number of neighbors,
B: 1 bit message,
M: constant

```
1  for i ← 0 to log D do
2      for j ← 0 to M log n do
                                                          /*  exchange 1  */
3          v = 0;
4          With probability 1/2^(log D - i),
5          broadcast B to neighbors;
6          set v = 1;
7          if received message from neighbor then
8              v = 0;
                                                          /*  exchange 2 */
9              if v = 1 then
10                 v = 0;
11                 Broadcast B;
12                 join MIS;
13                 exit the algorithm.
14             end
15         else
16             if received message B in this exchange then
17                 mark node u inactive;
18                 exit the algorithm.
19             end
20         end
21     end
22 end
```

8.3 Performance evaluation

The exhaustively investigated in the past competitor algorithms involved in our analysis, apart from the fact that they belong to different graph-theoretic classes, were chosen basically because of their distinct diversification in the degree of localization compared to *RanMIS*.

The first class, which enables *WuLi* [135], exhibit a high degree of localization. In such a class as soon as a node has collected information about its surrounding topology, it is able to decide whether it will be part of the backbone or not. The only information it needs to wait for, is the identity of the node in its (h hop) neighborhood. In the second class, the degree of localization shown by the algorithms is limited with respect to that of the first. Such protocols implement a distributed version of the heuristics for finding an *independent set* of nodes which is maximal, and a *dominating set* which is minimal. As a representative algorithm of this class we consider the *DCA* [10]. Being maximum, the *independent set* produced by *DCA* is also a minimal dominating set.

WuLi is a very simple distributed procedure consisting of a few local rules, the execution of which creates the desired CDS. Every node v exchanges its neighbor list with all its neighbors. A node set itself has a dominating node if it has at least two unconnected neighbors. In order to reduce the size of a CDS, the original protocol presents two pruning rules. According to the first rule, a node deletes itself from the CDS when its close neighbor set, which includes all of its direct neighbors as well as itself, is completely included in the neighbor set of a neighboring dominating node and it has smaller ID than the neighboring dominating node. According to the second rule, a node deletes itself from the CDS when its open neighbor set, which includes all of its direct neighbors, is completely included in the neighbor sets of two connected neighboring dominating nodes and has the smallest Id.

The *DCA* protocol assumes quasi-stationary nodes with real-valued weights, which are initially *UNDECIDED*. Node weights induce a total ordering of the nodes without any ties because node weights reflect real numbers. During the execution of the algorithm each node will decide to be *IN* or to be *OUT*, of the *independent set*. A node decides when all its neighbors with larger weight have decided. When the time comes, a node decides to be *OUT* if one of its neighbors is *IN*. Otherwise it decides to be *IN*. The *IN* nodes form the minimal dominating set required for clustering. The execution time depends on possible chains of dependency between the nodes, a negative consequence of the reduced localization *DCA* presents. The node with the smallest weight has to wait for all other nodes in the chain.

In subsection 8.3.1 we briefly introduce the graph model used in our simulation. Then in subsection 8.3.2 we present the simulation platform that produced all the network topologies used in the performance analysis. In subsection 8.3.3 we give the performance metrics of the protocols comparison and finally in subsection 8.3.4, we display the simulation results.

8.3.1 Simulation Model

As in most studies in multihop wireless networks, we used unit disk random graphs to represent the network topologies used in the performance analysis. A unit disk graph is determined by node positions and a fixed *communication range* R, for all nodes. The produced topologies which more precisely are described as *Constrained - Connected Random Unit Graphs (C-CRUG)*, because of the constrains the participating nodes are obliged to adhere, were constructed in MATLAB by taking in to consideration two crucial independent variables, concerning the final network connectivity, that is the network density, which is perfectly expressed by the degree d of each node, and the node population n.

8.3.2 Simulation platform

The graphs were normally generated by placing nodes randomly and independently from each other with the help of a modified version of *Minimum degree proximity algorithm (MIN-DPA)* [8], a *C-CRUG* generator, which aims to distribute node degrees more uniformly while maintaining connectivity. The idea is to place each new node in the vicinity of the node that has the smallest number of neighbors, while preserving a minimum distance to increase the probability of achieving a connected graph. The modified version of it though, involves the notion that the average degree of the topology and the transmission range of a node, were predefined.

The simulations refer to scenarios in which n static wireless nodes with maximum transmission range R are randomly and uniformly scattered in a geographic area of side L. We make the assumption that two nodes are neighbors if and only if their Euclidean distance is less than R.

We emphasize that nodes start the protocol execution at the same time and run the clustering and backbone formation algorithms to form a hierarchical multi-hop ad hoc network.

In our simulations R has been set to 50m, the number of nodes n has been assigned the values 100, 500, 1000 and 5000 while L has been set accordingly (see Table 8.1) to influence an average node degree of 4, 7, 10 and 15. This allowed us to test the protocols on increasingly dense networks, from (moderately) sparse to dense networks.

TABLE 8.1. L Parameter Values

Nodes	Degree#4	Degree#7	Degree#10	Degree#15
100	500	400	250	200
500	2000	1500	750	500
1000	5000	4000	1000	700
5000	7000	5000	2000	1500

In order to diminish any statistical errors all tests were repeated for 1000 times and the data used in our graphs are all average values of the results. A short description of the simulation parameters is given in Table 8.2.

TABLE 8.2. Interpretation of Used Symbols

Symbol	Usage	Default value
n	Number of Nodes in Network	
D	Intended Number of Neighbors	24
d	Average Node Degree	
M	Constant	32
R	Node Transmission range	
L	Axis Length (in meters) of AOR (Area Of Responsibility)	

8.3.3 Performance metrics

The assessment of the competing protocols, which was also implemented in MATLAB, were done along three dimensions: the first dimension portrays the cost (in delay and energy consumption) incurred by the protocols, the second concerns the characteristics of the resulting backbone, and the third one deals with the resilience of the resulting spanner to node failures.

Metrics for protocol cost. This family of measures, includes the *protocol duration* which is a direct measure of the delay incurred until the network spanner is established, and the *total number of messages exchanged* among nodes which is a measure of the energy consumed by the network, especially crucial for assessing the usefulness of clustering to networks with energy constrained nodes.

Protocol duration. It is the number of rounds required by the protocol to complete the procedure of backbone formation.

Total number of messages transmitted. We deal with two different situations (cases) where message exchange takes place among network nodes. The first situation concerns the backbone construction operation, i.e., the selection of network nodes which will take on the role of a CH, and the declaration of 'attachment' of $non-CH$ nodes to a CH. The second situation occurs when a backbone node fails. We don't examine failures of non-CH nodes[2] because message exchange in such scenarios is unrelated to the clustering protocol running. It is obvious that, if the number of messages exchanged for backbone maintenance is high, then local reclustering may impose a non-negligible burden to the network, and a backbone recalculation should take place. To uncouple our measurements from MAC particularities and focus only to the pure clustering protocol performance, we assume an ideal MAC layer with no provision for retransmissions in case of collisions exist.

Metrics for backbone description. This family of measures assess the backbone's 'layout'.

[2]either gateway nodes which are used by *RanMIS* and *DCA* in order to achieve backbone connectivity or plain nodes

Backbone size. It is the number of the network nodes that comprise the backbone. A smaller backbone is desirable for minimizing the routing overhead. For instance, in sensor networks with nodes that can turn off their radio interface (energy saving sleep mode), the backbone size is a measure of how many nodes need to stay awake for data transport. Usually, the nodes in the backbone stay up or have a higher duty cycle for guaranteeing routes to the sink, and hence the smaller the backbone the more nodes can be in energy conserving mode. On the other hand, a very small backbone could result in situations where CH-to-member communication would require either excessive amount of energy to reach the node and vice versa, or multi-hop communication within the cluster.

Cluster cardinality distribution. It is the distribution of the number of members for each generated cluster.

Route length. If we consider the subgraph G_b of the network induced by the backbone construction (G_b is the graph where there are no links between ordinary nodes), then this metric gives a measure of how well a routing protocol can perform over the backbone.

Inter-clusterhead distance distribution. This distribution records the distances among neighboring CHs. The larger this distance is the better for the performance of the clustering protocol, since, for instance, a three-hop distance among clusterheads would avoid hidden terminal problems and guarantees the 'independence' of clusterhead transmission.

Metrics for robustness. This evaluation is useful in cases where node failures are anticipated. There are various definitions for evaluating the robustness of a network [31, 107]. In general, the "robustness" of a network measures its resilience (in terms of connectivity) to the removal of edges or vertices. Robustness can be formalized in a variety of ways, depending on how connectivity is measured, whether edges or vertices are removed, and how the edges or vertices to be removed are chosen.

In our experimentations, we consider a wireless ad hoc network where nodes are removed randomly until backbone connectivity is lost. Figure 8.1, depicts a section of a larger wireless network where a message is transmitted from $node_{22}$ to $node_{23}$. It is obvious that the scenario under consideration deals with an MIS network topology where gateway nodes are used to forward any transmitted messages. Adjacent CHs differ in their fill in colors and any clustermembers have a border line color which is in consistency with their CH fill in color. A red dotted line is used to represent the shortest path for the current transmission. We emphasize that in our tests whenever a clustermember faces the dilemma of choosing between more than one CH candidates then by default it selects the one with the smaller Id.

We assume that nodes have an inherent capability to identify in their neighborhood either a failed CH as clustermembers or a failed gateway node as CHs. Taken this assumption in to consideration, as soon as a node failure is perceived, each competitor protocol strives to solve locally any network inefficiencies so that the connectivity is maintained and no uncovered nodes

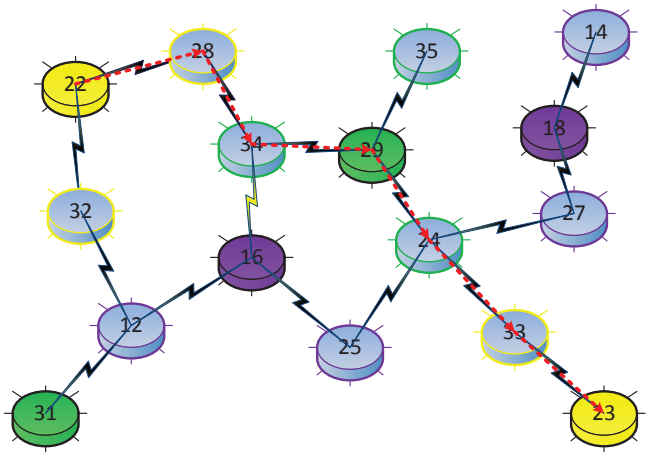

FIGURE 8.1. Shortest Path transmission when all nodes are in working condition.

exist. The metric considers an upper bound of 3-hop message retransmission in order to establish connection between CHs. We examined two different scenarios for robustness depending on the role of the failed node in the network.

Network substratum Robustness is defined as the number of non-CH nodes whose removal (because of failure or energy depletion) causes backbone disconnection. This metric is directly related to *RanMIS* and *DCA* protocols who build an MIS network topology and use as *gateways* non-CH nodes to maintain backbone connectivity. *WuLi* who builds CDS and is presented also in the diagram illustrates the total number of non-CH nodes in the network because failure of non-CH nodes has negligible effects to backbone connectivity. Intuitively network performance is connected to the cardinality of the clusters and it is straightforward that a populated cluster is more desirable than a sparse occupied.

In Figure 8.2, we see the impact of a gateway failure to the shortest path construction. If the failure concerns a non-gateway node then network connectivity won't be affected at all.

We addressed network substratum failures as follows:

- Initially, we run Dijkstra's routing algorithm on the whole network topology to identify if backbone connectivity is maintained after node failure.

- If so, which means that the gateway role on the cluster who suffered the failure was assigned to another node without any further consequences to the backbone connectivity, we repeat the scenario with another node until backbone connectivity is lost.

- If not, which means that node failure had catastrophic consequences to the backbone connectivity, we use the total number of the failed nodes that each protocol could bear prior to loss of backbone connectivity to describe the network resilience to failures of such kind w.r.t. network size and its density.

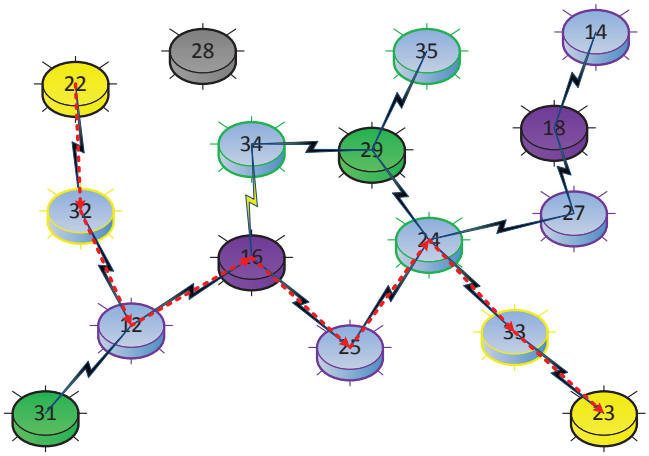

FIGURE 8.2. Shortest Path recalculation as a result to gateway node failure.

Backbone Robustness, is defined as the number of backbone nodes whose removal (because of failure or energy depletion) causes backbone disconnection or the uncovering of ordinary nodes. This metric provides an indirect measure on how long the network will be operational before requiring a backbone recomputation.

We use the network topologies presented in Figures 8.1, 8.3, 8.4 to describe the way we address *CH* node failures with *RanMIS*, *DCA*, *WuLi* protocols, respectively.

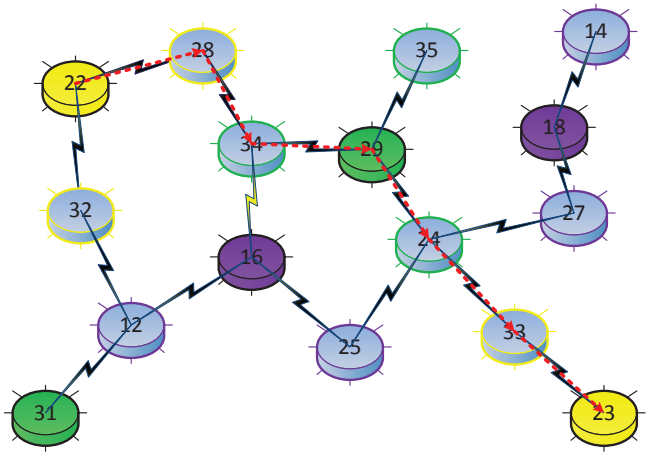

FIGURE 8.3. Shortest Path for a DCA network topology.

In Figure 8.5, we see the impact of an *RanMIS CH* failure to the shortest path construction. We addressed *RanMIS CH* failures as follows:

- We locate any resultant orphan clustermembers due to *CH* failure.
- If any, we attempt to associate them with a neighboring *CH*.

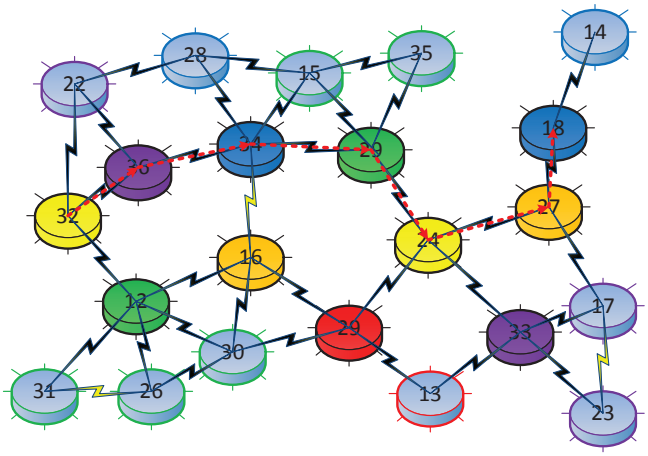

FIGURE 8.4. Shortest Path for a WuLi network topology.

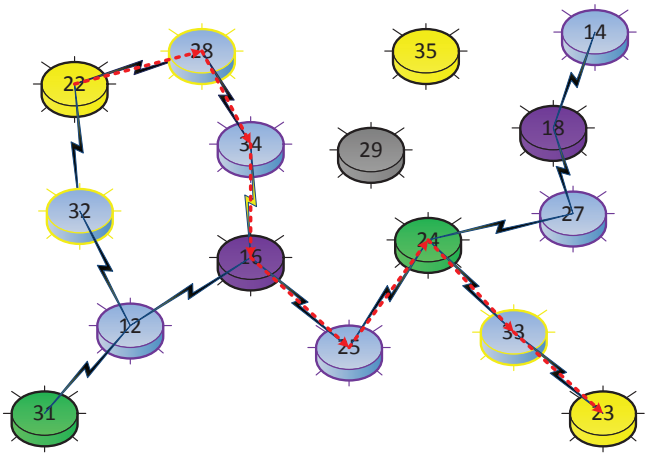

FIGURE 8.5. Shortest Path recalculation as a result to *RanMIS* CH node failure.

- Remaining orphan clustermembers become *CH* candidates, and rerun the protocol.
- Then we run Dijkstra's routing algorithm on the whole network topology to identify if backbone connectivity is maintained.
- If so, we repeat the scenario with another one *CH* until backbone connectivity is lost.
- If not, we use the total number of the removed nodes that each protocol could bear prior to loss of backbone connectivity to describe the network resilience to failures of such kind w.r.t. network size and its density.

In Figure 8.6, we see the impact of a *DCA CH* failure to the shortest path construction. We addressed *DCA CH* failures as follows:

- We locate any resultant orphan clustermembers due to *CH* failure.

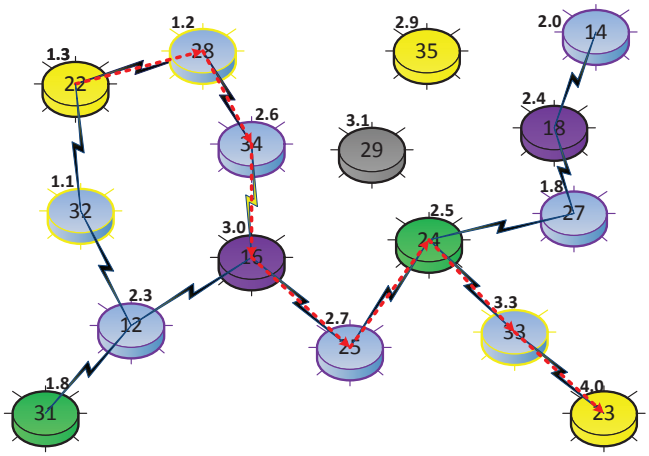

FIGURE 8.6. Shortest Path recalculation as a result to *DCA* CH node failure.

- If any, their status will be determined in sequence behind any undecided neighboring node with larger weight.
- When appropriate we associate orphan clustermembers with a CH whose weight is the larger in the neighborhood.
- Remaining orphan clustermembers become CH candidates, rerun the protocol and finally are announced CHs.
- Then we run Dijkstra's routing algorithm on the whole network topology to identify if backbone connectivity is maintained.
- If so, we repeat the scenario with another one CH until backbone connectivity is lost.
- If not, we use the total number of the removed nodes that each protocol could bear prior to loss of backbone connectivity to describe the network resilience to failures of such kind w.r.t. network size and its density.

In Figure 8.7, we see the impact of a *WuLi* CH failure to the shortest path construction. We addressed *WuLi* CH failures as follows:

- Initially, we locate any CH whose role was determined by the presence of the failed node, when it was in working condition, and we mark it as an orphan clustermember.
- Any clustermembers that the former CH had, are marked as orphan clustermembers also.
- Afterwards, we locate any resultant orphan clustermembers due to CH failure.
- Subsequently, we attempt to associate orphan clustermembers with a neighboring CH.

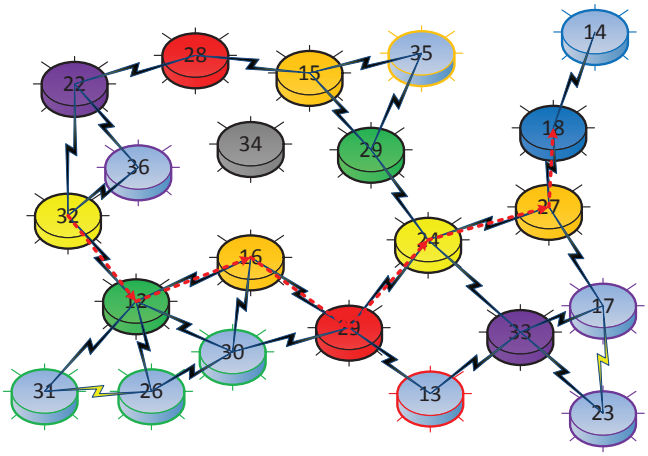

FIGURE 8.7. Shortest Path recalculation as a result to *WuLi* CH node failure.

- Then we run Dijkstra's routing algorithm on the whole network topology to identify if backbone connectivity is maintained.
- If so, we repeat the scenario with another one *CH* until backbone connectivity is lost.
- If not, we use the total number of the removed nodes that each protocol could bear, prior to loss of backbone connectivity, to describe the network resilience to failures of such kind w.r.t. network size and its density.
- It is worth to mention that, uncovered clustermembers are taken into account and cause the termination of protocol execution when discovered.

The way *WuLi* addresses *CH* failures which was proposed by its authors in [135] has been proven inefficient concerning the overall energy consumption to backbone maintenance. It is our belief that once a network structure has been established the effort should be to preserve it as long as the connectivity is maintained. We were motivated by this assumption to present a modified version of *WuLi* concerning the way it deals with CH node failures. In Figure 8.8, we see the impact of a *WuLi CH* failure to the shortest path construction when using the Proposed *WuLi* protocol.

In our proposed version of *WuLi** we address *CH* failures as follows:

- Initially, we deal with substratum clustermembers to identify whether they can be assigned a new role due to *CH* failure or not.
- Subsequently, we attempt to associate orphan clustermembers with a neighboring *CH*.
- Then we run Dijkstra's routing algorithm on the whole network topology to identify if backbone connectivity is maintained.

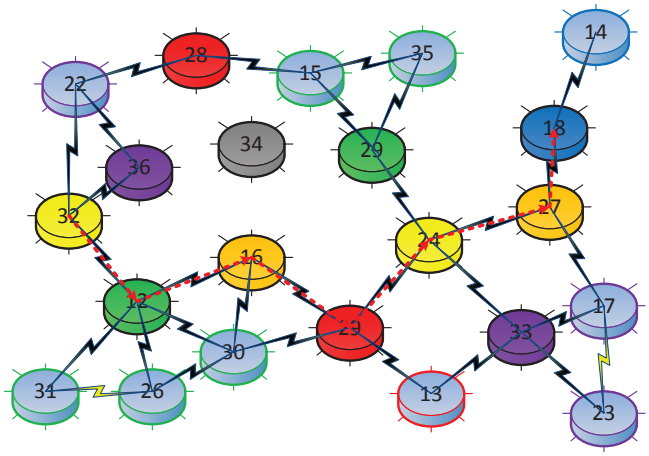

FIGURE 8.8. Shortest Path recalculation as a result to *WuLi** CH node failure.

- If so, we repeat the scenario with another one *CH* until backbone connectivity is lost.
- If not, we use the total number of the removed nodes that each protocol could bear, prior to loss of backbone connectivity, to describe the network resilience to failures of such kind w.r.t. network size and its density.
- Uncovered clustermembers are taken into account also and cause the termination of protocol execution when discovered.

8.3.4 Simulation results

In order to evaluate the performance of *RanMIS*, we selected two graph-theoretic ad hoc networks clustering algorithms, namely the *WuLi* and *DCA*, both of which create dominating sets, the former produces a connected dominating set and the latter produces a maximum independent set. These algorithms are simple, practical, extremely popular in the clustering community [3] and comprise the base for the development of many similar clustering algorithms [42]. We performed a series of experiments to compare the performance of *RanMIS* against these algorithms. The first set of experiments concerns the tuning of *RanMIS* with respect to its parameters, and then it follows its comparison against its competitors.

8.3.4.1 Tuning of *RanMIS*

The original article which introduced *RanMIS* [3], described two parameters that control its operation and affect its overall performance. The first of these parameters, D, concerns the estimated size of neighborhood of each node, while the second, M, is a constant related to the

[3]They both have an excessive number of citations. See http://scholar.google.com/scholar?cites=7672109277762563412&as_sdt=2005&sciodt=0,5&hl=en and http://scholar.google.com/scholar?cites=13992908027776879246&as_sdt=2005&sciodt=0,5&hl=en.

number of rounds required for the algorithm to complete the backbone construction. These parameters are independent and thus we have to make an initial choice for one of them (e.g., for D) and investigate the other (e.g., M), and vice versa.

Impact of parameter D. To evaluate the impact of parameter D on *RanMIS*'s performance, we conducted a series of experiments for various values of D, 6, 12, 24 36, and 48 and for various performance measures. D controls the maximum number of rounds *RanMIS* has in order to complete the backbone construction and it is consistent with the size of each node's neighborhood. That is because in a large neighborhood the intuition is that the likelihood to have a simultaneous transmission and a resultant collision between two adjacent transmitting nodes is increased, and consequently more protocol rounds will be required by the affected nodes in order to decide about their status.

In Figure 8.9, we see the the number of rounds required by *RanMIS* to complete backbone construction for various values of D w.r.t. network size and its density.

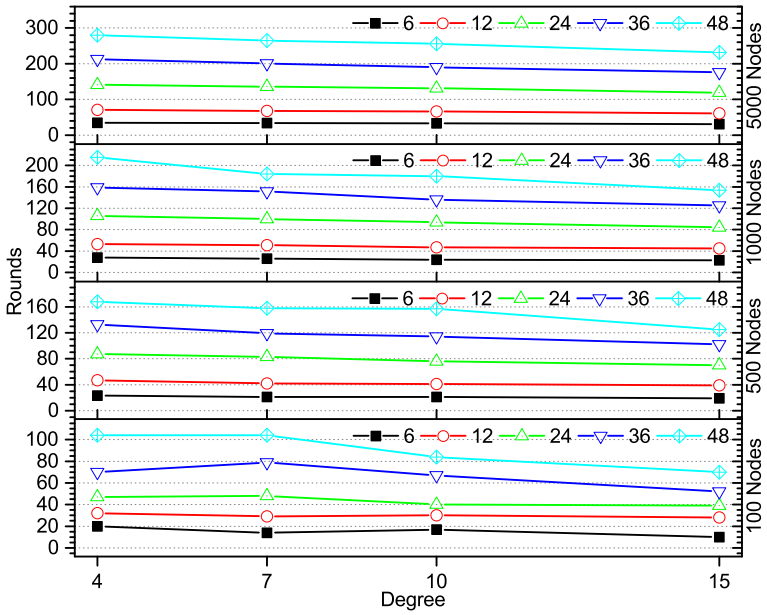

FIGURE 8.9. Impact of D on the number of Rounds required by *RanMIS* to complete backbone construction w.r.t. network size and its density.

The first observation is that, in accordance with our intuition, when the value of D increases, the number of rounds required by *RanMIS* to complete backbone construction increases for every network size. This consequence derives from the fact that there is a direct relation between the probability a node has in order to be chosen as a CH candidate and the setting of parameter D (cf. Algorithm 8.1). The higher the setting of D the lesser the probability for a node to become a CH and if so, the probability for a neighboring node to become a CH is also small (thus, we avoid any unwanted message retransmissions due to collisions). The second observation is that

while the number of rounds is almost irrelevant to network density for a given network size, the number of rounds increases linearly to the network size. This is due to the fact that more clusters will be developed and therefore the competition for becoming a CH increases. It is obvious that the less the required rounds to complete the backbone is, the best for the protocol is (namely in Figure 8.9 this observation favors D setting of 6 and 12 to the other). Though, to conclude which is the best choice for parameter D, we must examine also the number of transmitted messages and the number of clusters created for the various choices of D (cf. Figure 8.10 and Figure 8.11). Thus, we do not conclude at this point that the best value for D is 6.

In Figure 8.10, we see the number of messages exchanged by *RanMIS* for backbone construction for various values of D w.r.t. network size and its density.

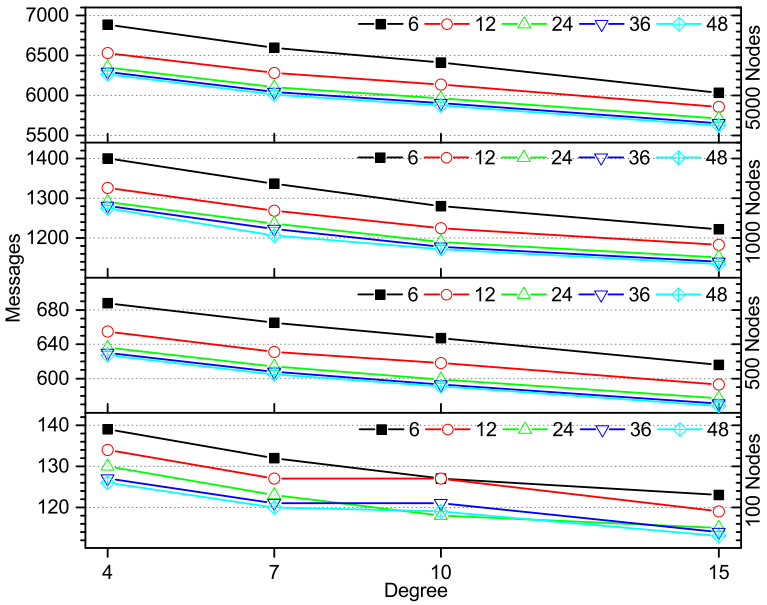

FIGURE 8.10. Impact of D on the number of messages exchanged by *RanMIS* for backbone construction w.r.t. network size and its density.

The situation here is reversed with respect to what we saw in Figure 8.9. In general, the number of messages required by *RanMIS* to complete the backbone construction process decreases for increasing values of D. Therefore, so far the best choice for D is the one that achieves a balance between the number of rounds and the messages transmitted, i.e., $D = 24$ (we will revisit this issue when commenting the next figure). Additionally, examining the figure, we observe a decreasing number of transmitted messages for increasing network density (for every network size). This is because when a node declares itself as a CH, then all of its neighbors (and there are too many nodes in dense networks) immediately attach themselves to this node. The final observation is that the relative ordering of the performance curves for various values of D is maintained across different densities (with some statistically insignificant variation for very small networks).

In Figure 8.11, we see clusters produced by *RanMIS* for backbone construction w.r.t. network

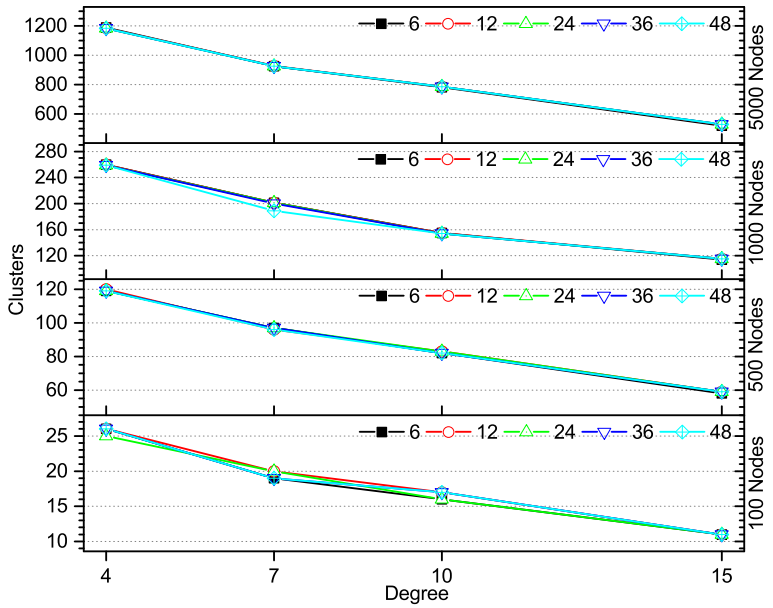

FIGURE 8.11. Impact of D on the number of clusters produced by *RanMIS* for backbone construction w.r.t. network size and its density.

size and its density for various values of D.

The main observation is that the number of clusters produced is not affected by parameter D. However, protocol execution drives to decreasing number of produced clusters w.r.t. increasing network density for every network size, and the explanation for that is similar to the one for the previous figure.

Overall, *a value of* D = 24 *is the most appropriate choice* since it achieves a good trade-off between network backbone construction delay (due to many rounds) and the transmitted messages.

Impact of parameter M**.** To evaluate the impact of constant M on *RanMIS*'s performance, we conducted a series of experiments for various values of $M = 2^x$ (x = 0 - 7), and for various performance measures. For these sets of experiments, the value of D was set equal to 24 (cf. subsection *Impact of parameter* D.)

Initially, we examined the impact of M on the number of rounds required by *RanMIS* to complete the backbone construction. In Figure 8.12, it is clear that an M = 8 setting supersedes any other for practically all network instances (except perhaps for 100 nodes), basically because less work in the inner loop of *RanMIS* (see Algorithm 8.1), signifies less residency in a low probability state for a node to become a CH. Practically, with small settings of M, we bias the system to converge faster. The same observation holds in Figure 8.13, for even smaller values of M (namely 1–8).

The disadvantage though, of a small M setting is that it might create severe competition

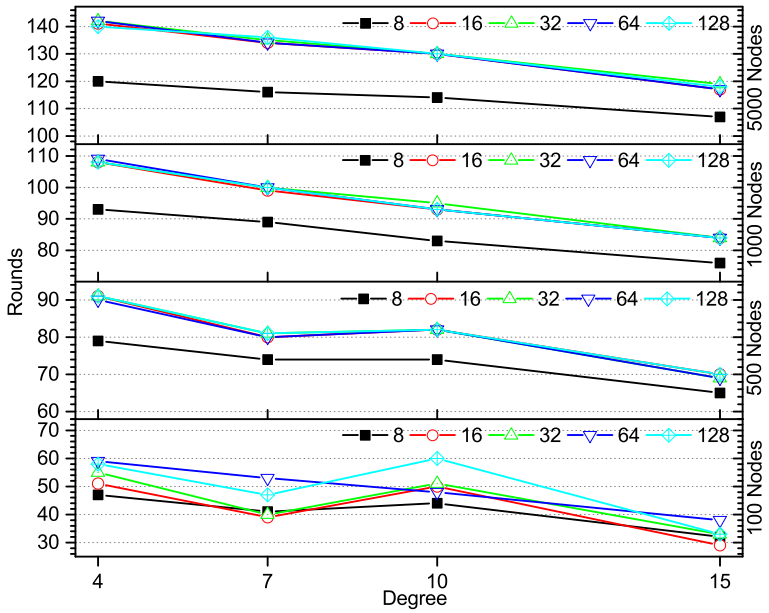

FIGURE 8.12. Impact of M on the number of rounds required by *RanMIS* to complete backbone construction w.r.t. network size and its density.

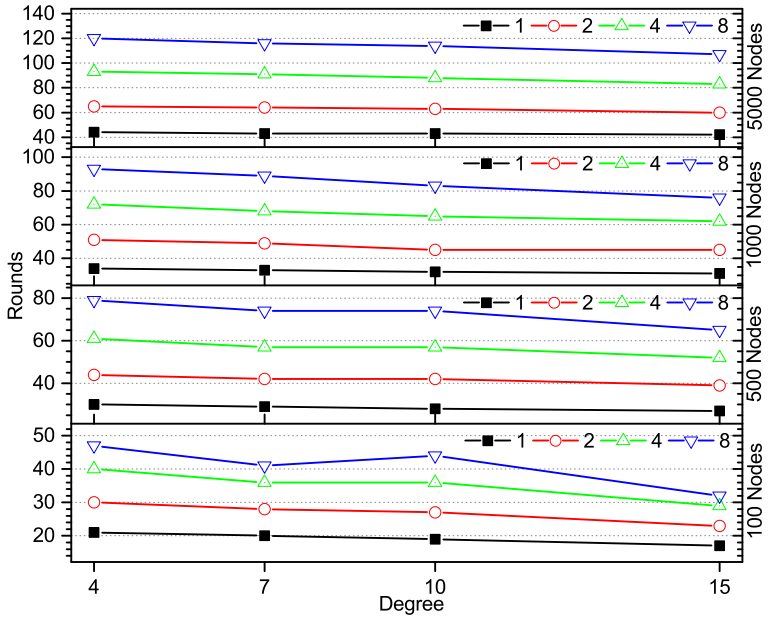

FIGURE 8.13. Impact of M on the number of rounds required by *RanMIS* to complete backbone construction w.r.t. network size and its density.

for the role of CH among neighboring nodes, and since M affects also the maximum number of rounds that are available to the protocol for backbone construction, it is possible to drive it not to converge. Figure 8.14 reflects our thoughts, where for small settings of M (e.g 1) more than 90% of the available rounds are used by *RanMIS* in order to converge.

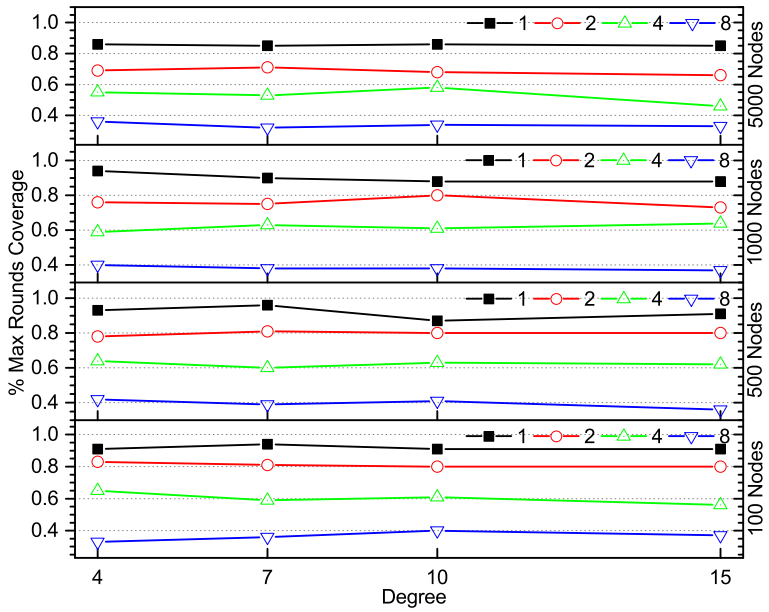

FIGURE 8.14. % percentage coverage of Max number of rounds available to *RanMIS* in order to converge w.r.t. M parameter setting.

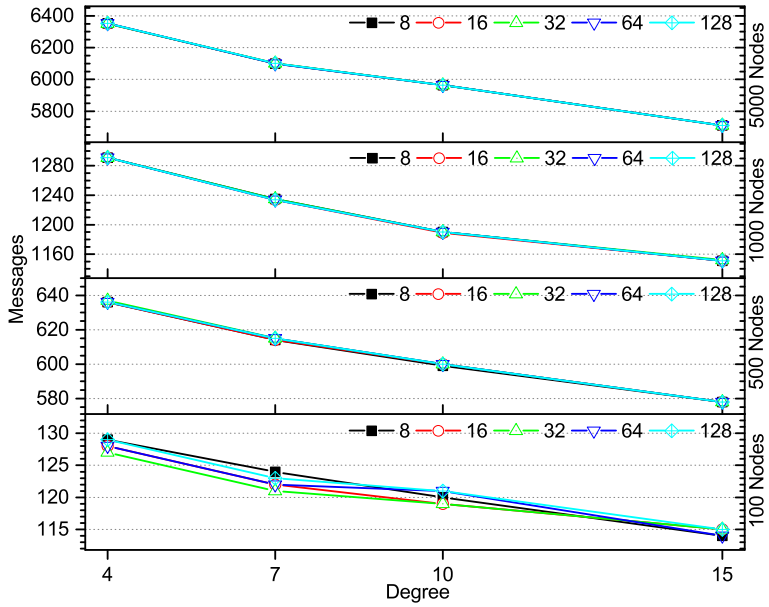

FIGURE 8.15. Impact of M on the number of messages exchanged by *RanMIS* for backbone construction w.r.t. network size and its density.

In Figures 8.15, 8.16, we see the impact of M on the number of messages exchanged by *RanMIS* to complete the backbone. The observation is that for practically all network instances the number of transmitted messages is unrelated to M.

In Figure 8.17, 8.18, we see the impact of M on the number of clusters produced by *RanMIS*.

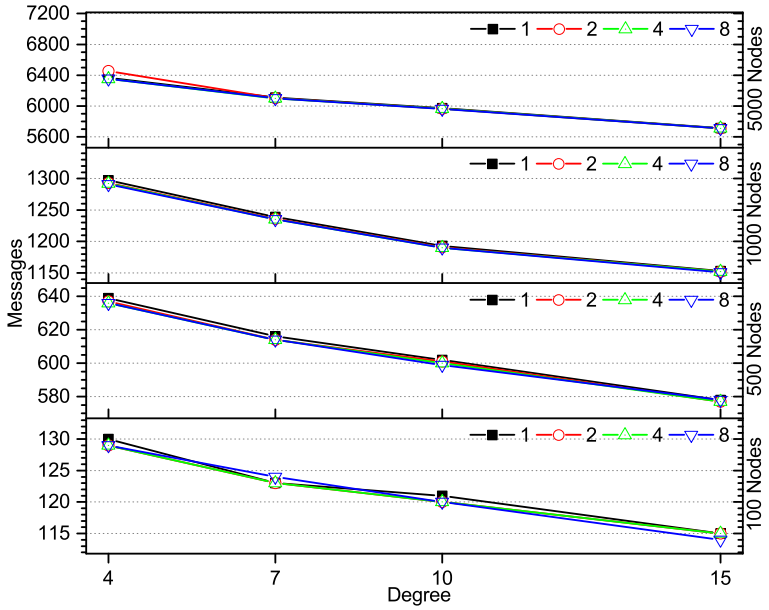

FIGURE 8.16. Impact of M on the number of messages exchanged by *RanMIS* for backbone construction w.r.t. network size and its density.

Similarly, the number of clusters produced is not affected by M (see Figure 8.17).

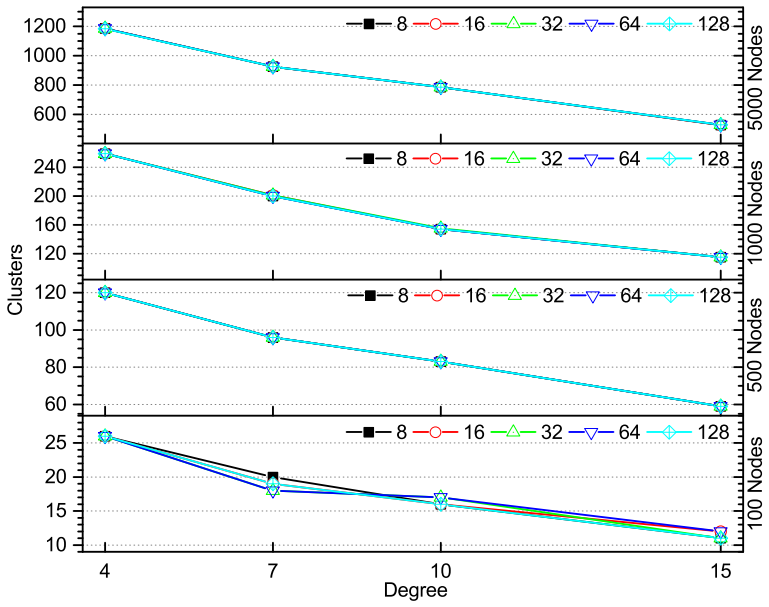

FIGURE 8.17. Impact of M on the number of clusters produced by *RanMIS* w.r.t. network size and its density.

Overall, it seems that *the authors' suggestion of* $M = 34$ *is not the best choice*; it is only marginally good for small networks (100 nodes) when we want to minimize the number of transmitted messages – the difference though is not really significant. We concluded though, that

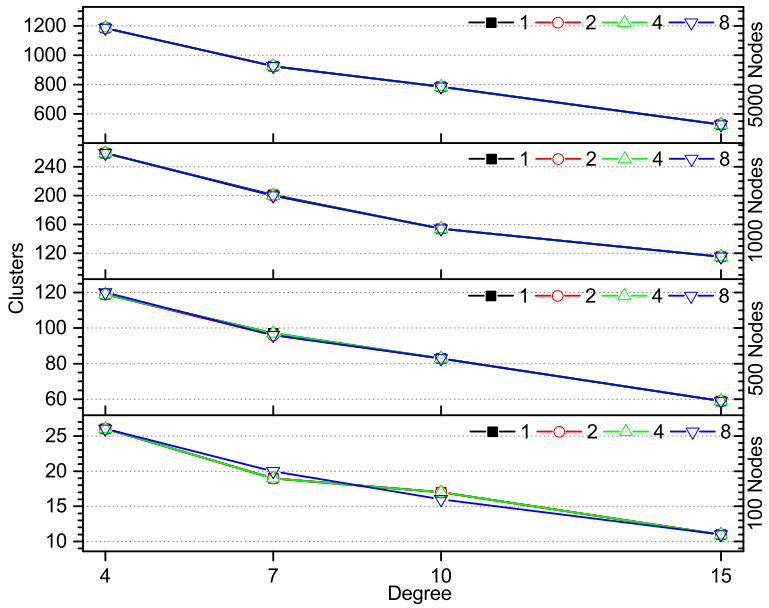

FIGURE 8.18. Impact of M on the number of clusters produced by *RanMIS* w.r.t. network size and its density.

we should be conservative with the setting of M, especially when we have to deal with real life network topologies. Nevertheless, in the rest of this chapter, we follow the *RanMIS*'s creators suggestion and set the value of M equal to 32.

8.3.4.2 Comparison of competing clustering protocols

The next paragraphs unveil the nature of the competing protocols concerning their operational cost, their network topology uniqueness and their backbone robustness.

Results concerning the protocol cost. In this series of experiments, we measured the number of rounds required by each protocol to complete and also the total number of messages transmitted. In Figures 8.19 and 8.20, we see the relative comparison of the algorithms.

As far as the number of protocol rounds is concerned, we observe that *DCA* is the clear winner for all network instances. The performance of the algorithms is consistent with the what the theory predicts for them; thus we dot not comment further on this issue.

As far as the number of transmitted messages for backbone construction is concerned, we observe that *RanMIS* is now the clear winner confirming its theoretical behavior. Its performance gap from the other competitors is stable across all network topologies. *DCA* sends around 40% more messages than *RanMIS* does, and *WuLi* sends around 60% more messages than *RanMIS*. Therefore, even though it takes more time to *RanMIS* to construct the backbone due to its probabilistic nature, it does this with far less messages than its competitors.

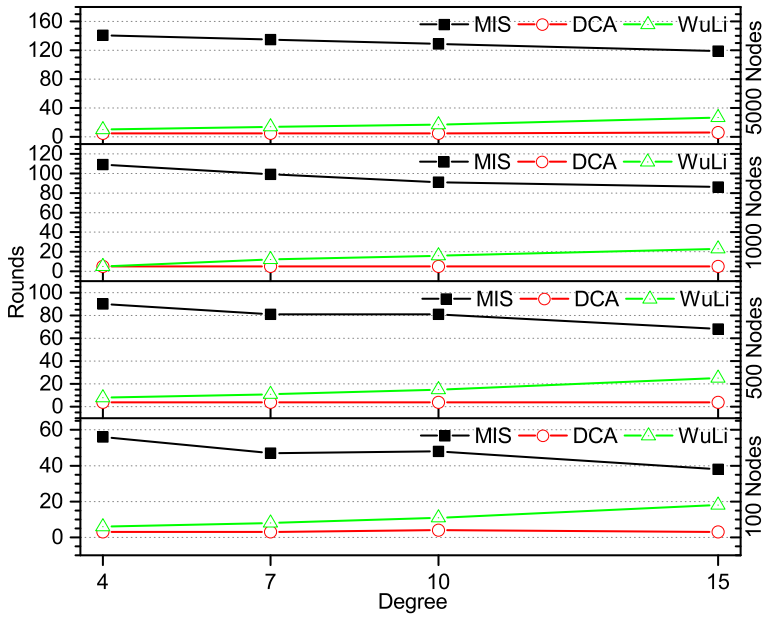

FIGURE 8.19. Rounds required by each competitor algorithm to complete backbone construction w.r.t. network size and its density.

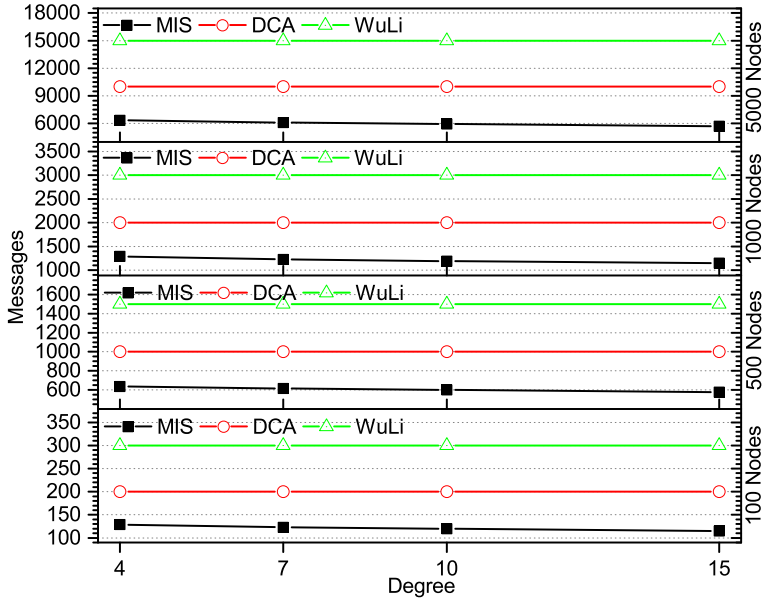

FIGURE 8.20. Messages required by each competitor algorithm to complete backbone construction w.r.t. network size and its density.

As far as the number of transmitted messages for backbone reconstruction when a node dies, are presented in Figure 8.21. In this figure, apart from the three competitors we present also the proposed improvement to *WuLi* which was described in page 132. The first observation is that this proposed improvement is the best performing algorithm which is expected, since the

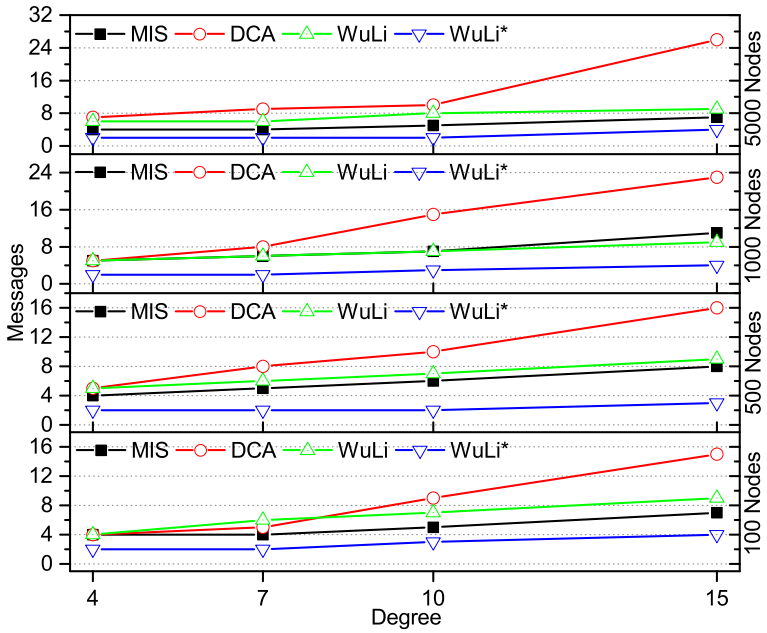

FIGURE 8.21. Impact of network size and its density on messages required to rebuild locally the broken backbone in case of a CH failure.

central idea behind its development was how to least modify the existing network backbone, and therefore transmit as few messages as possible. The second best performing algorithm is *RanMIS*, which is also expected since it is message optimal. The performance of all algorithms deteriorates for denser networks, because a single node's failure affects a lot of neighboring nodes, but their performance is not significantly affected by the network size.

Results concerning the backbone description. In Figures 8.22–8.29 we present the performance of the protocols for the metrics belonging to the category of "backbone description".

The number of generated clusters by each protocol is presented in Figure 8.22. The first observation is that *RanMIS* and *DCA* produce almost the same number of clusters; even though their difference is indistinguishable in the figure, their difference is less than 1% which can be attributed to statistical variation. The reason for this resemblance is that both algorithms are building maximum independent sets using either a weight for each node (assigned dynamically or statically) in the case of *DCA* or using randomization in choice in the case of *RanMIS*. On the other hand, *WuLi* produces too many clusters, since it is mandatory to create *connected* dominating sets, whereas the former two algorithms are building plain dominating sets. Finally, we observe that the number of generated clusters reduces with increasing density, since in dense networks more nodes are able to find a CH (i.e., dominator) in their 1-hop neighborhood.

Next, we evaluated the cluster cardinality distribution for various densities of the network. The results are illustrated in Figures 8.23–8.26. At this point we should emphasize that this

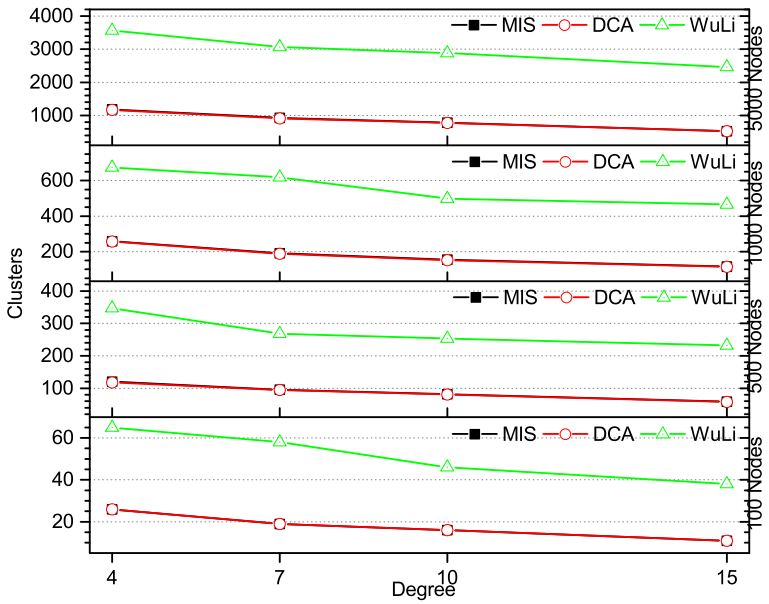

FIGURE 8.22. Clusters produced by each competitor algorithm w.r.t. network size and its density.

distribution should in general be 'bell-shaped' or look like any distribution with probability mass concentrated around the average node degree of the topology. In Figures 8.23 and 8.24, we see that this property is achieved by *RanMIS* and *DCA* for sparse networks, but it is not achieved for dense networks (in Figures 8.25 and 8.26). In these last two figures, it seems that a significant percentage of clusters – around 22% of them – contain very few nodes, i.e., 1 or 2, including the clusterhead. On the other hand, *WuLi* exhibits an undesirable pattern across all network topologies, since more than 70% of its clusters contain 1 or 2 nodes, which is also a consequence of the large numbers of clusters produced (see Figure 8.22).

The diameter[4] length of the produced backbone by each protocol is presented in Figure 8.27 (in meters) and in Figure 8.28 (in hops). The common observation is that *WuLi* is the best performing algorithm which is due to the fact that it produces $CDSs$ and thus there are no gateway nodes intervening among CHs so as to increase the diameter. The second observation is that the diameter in general decreases with increasing network density, since in dense networks it easier to find short routes for any pair of nodes. In some cases where the topology is peculiar, the diameter increases in denser networks (e.g., network with 500 nodes with density equal to 7).

In Figure 8.29 we illustrate the percentage of adjacent CHs that reside at a distance of 3-hops from each other. Apparently, *WuLi* produces CHs at 1-hop distance from each other, and therefore we do not include it in the plot. For the other two algorithms, which produce maximum independent sets, it holds by the definition of the MIS, that any two adjacent CHs will be either at a distance of 2 hops or at a distance of 3 hops, with the latter being the preferred one (see

[4]The largest of the shortest paths for any pair of nodes.

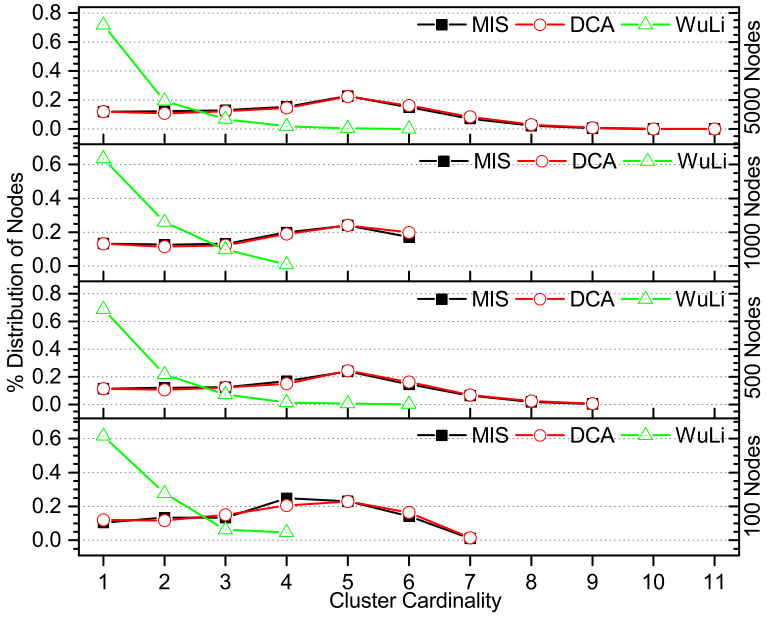

FIGURE 8.23. Distribution of nodes in clusters after backbone construction w.r.t. network size and a density of $D = 4$.

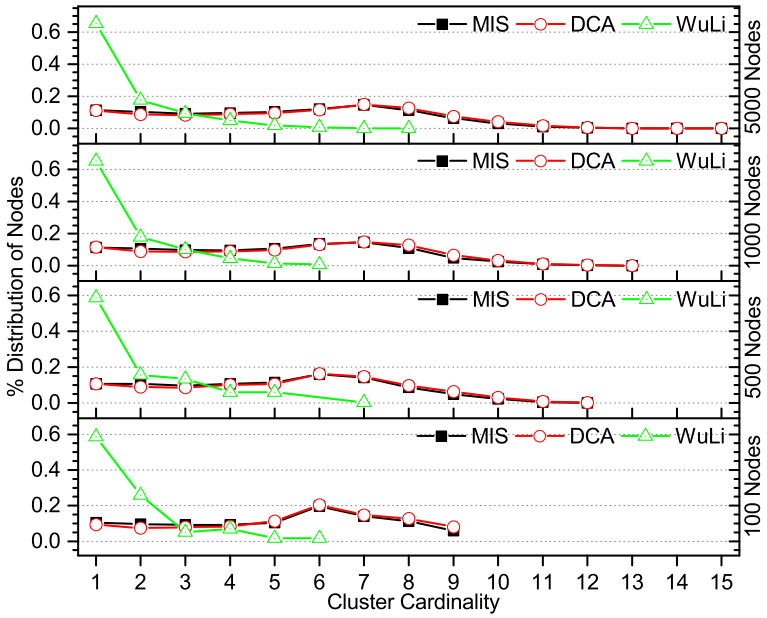

FIGURE 8.24. Distribution of nodes in clusters after backbone construction w.r.t. network size and a density of $D = 7$.

paragraph on "Inter-clusterhead distance distribution" definition, Section 8.3.3). We see that both algorithms achieve approximately the same performance – the gap among them is statistically insignificant. In particular, we see that the majority (more than 50%) of adjacent CHs in almost

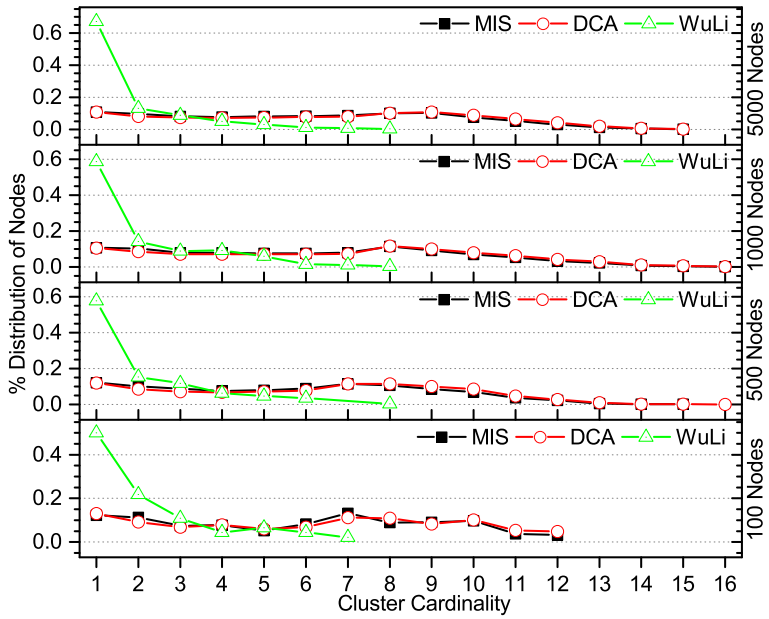

FIGURE 8.25. Distribution of nodes in clusters after backbone construction w.r.t. network size and a density of $D = 10$.

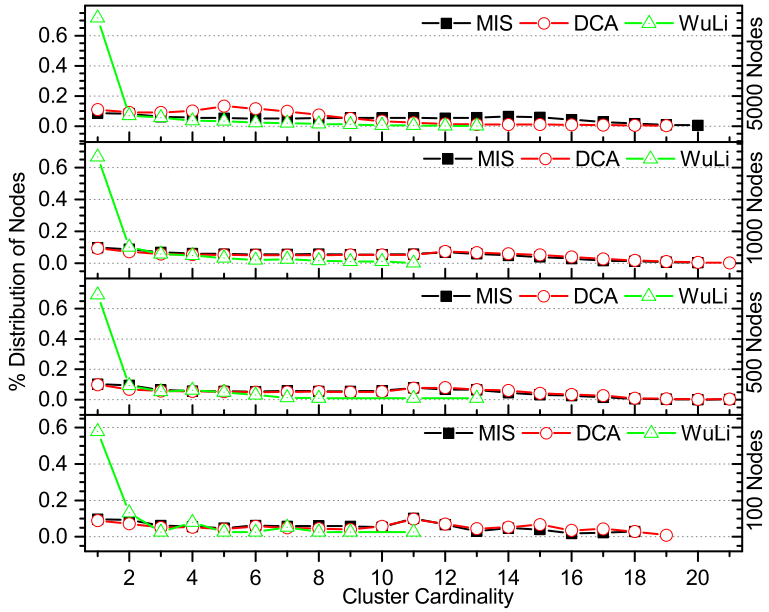

FIGURE 8.26. Distribution of nodes in clusters after backbone construction w.r.t. network size and a density of $D = 15$.

all network topologies, are 3-hops away. The only exception appears for very small and very sparse networks (i.e., 100 nodes with degree 4), where this percentage drops to 48%; this is due to the fact that there are not too many links among the nodes to establish 3-hop distance among neighboring CHs.

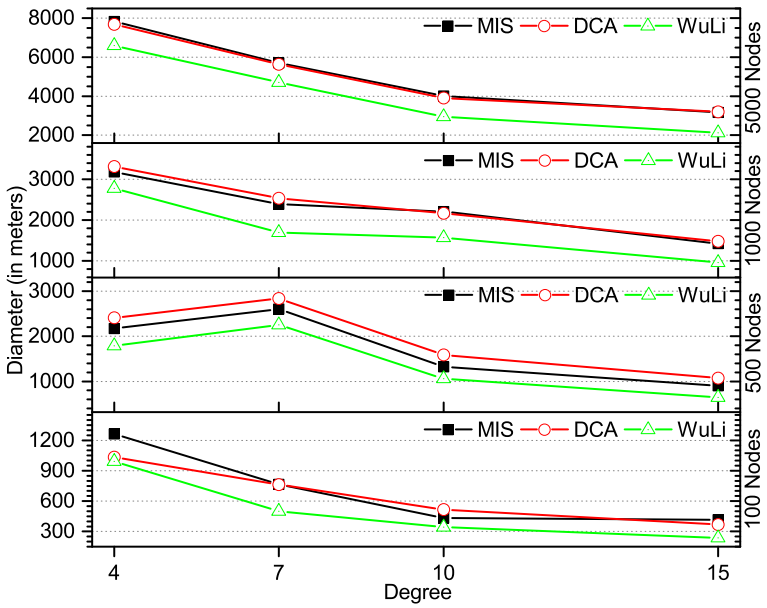

FIGURE 8.27. Network Diameter diversification (in meters) after backbone construction w.r.t. network size and its density.

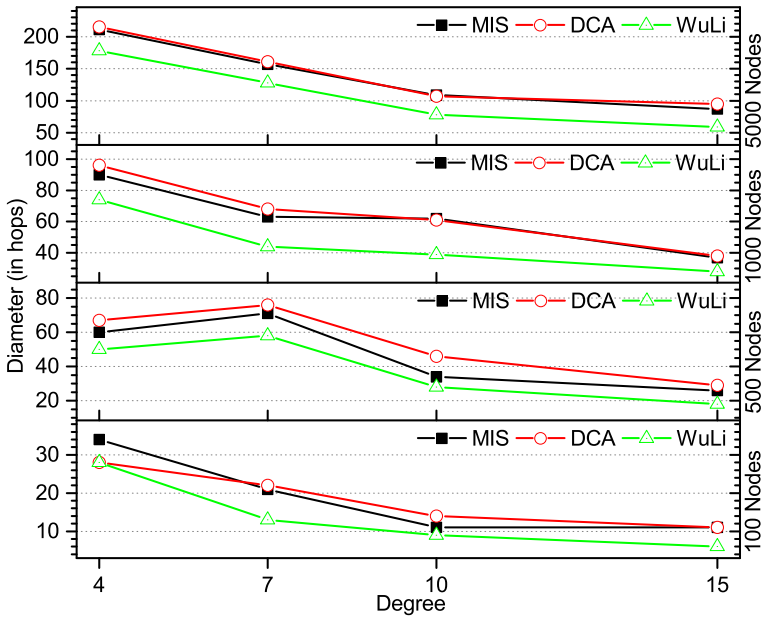

FIGURE 8.28. Network Diameter diversification (in Hops) after backbone construction w.r.t. network size and its density.

Robustness evaluation. Next, we evaluated the robustness of the resulting backbone for various densities of the network. The results are illustrated in Figures 8.30–8.31. In Figure 8.30 we illustrate the number of non-CHs nodes that must be removed to lose backbone connectivity.

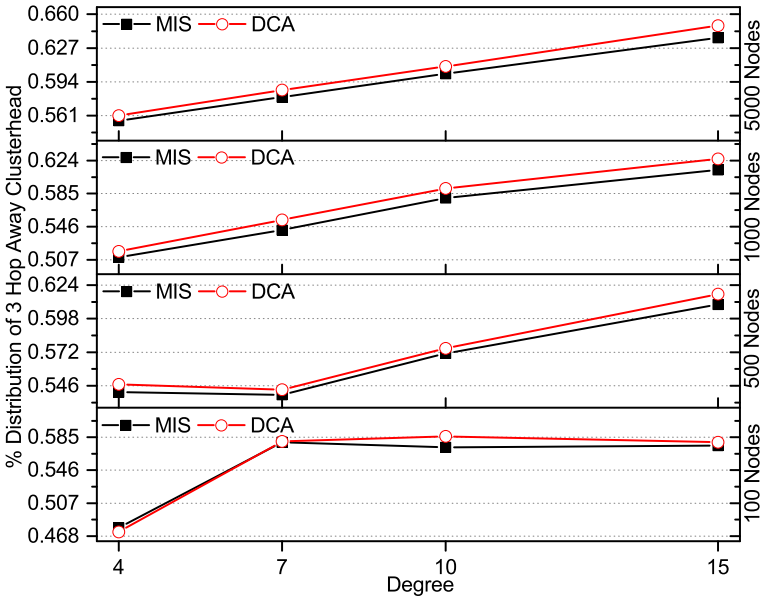

FIGURE 8.29. CH Distance Distribution after backbone construction w.r.t. network size and its density.

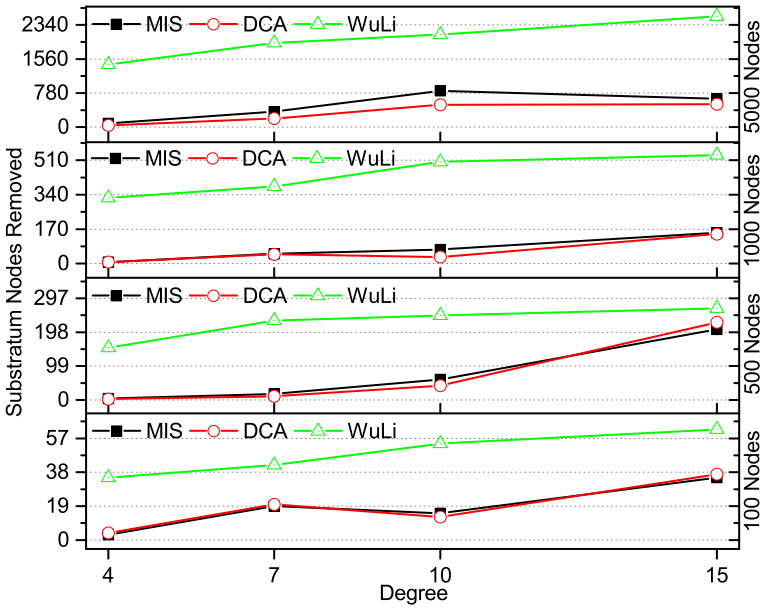

FIGURE 8.30. Non-CH nodes (gateways) removed before backbone recalculation w.r.t. network size and its density.

For *WuLi* which has no gateway nodes, the plot simply shows the total number of clusters members. For the other two algorithms, their performance is similar for all topologies, but for the large networks, where *RanMIS* exhibit a more robust behavior than *DCA*, with their performance gap widening for denser networks (namely 4–7 degree).

Finally, we examined the backbone robustness of the competing protocols and the results are illustrated in Figure 8.31 and 8.21. For all but the very dense networks, *RanMIS* is the best performing algorithm when the number of backbone nodes removed or the number of transmitted messages is considered. Though, for very dense networks (density equal to 15), *RanMIS*'s performance is inferior to *WuLi* (and its variation) when considering the number of backbone nodes allowed to be removed. When examining the number of transmitted messages, *RanMIS* loses only, as expected, by *WuLi*'s variation whose goal (as mentioned before) is to violate *WuLi* principle of CHs selection in order to keep the message overhead as low as possible.

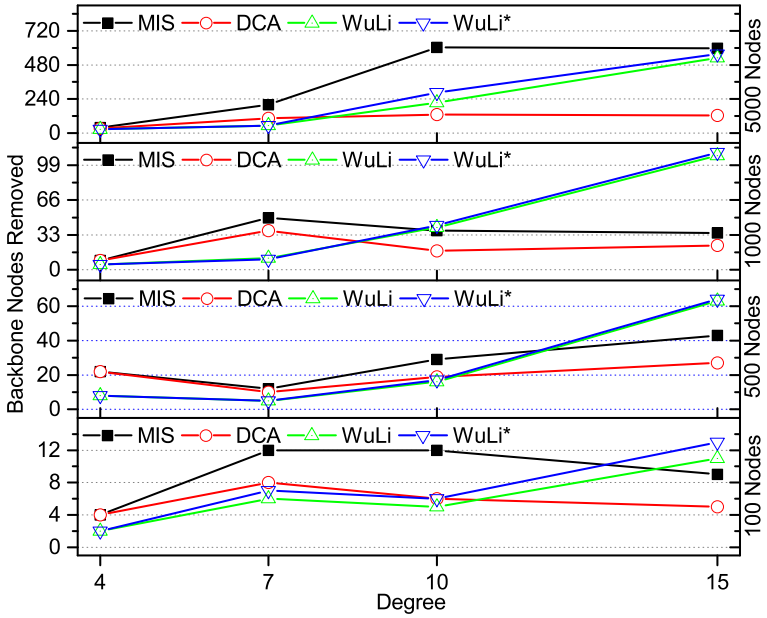

FIGURE 8.31. CH nodes removed before backbone recalculation w.r.t. network size and its density.

Overall comparison of the protocols. Here we briefly summarize the most significant conclusions from the experiments. The value of parameter D controls a tradeoff between the number of rounds needed to complete *RanMIS* and transmitted messages; a value equal to 24 seems to achieve a good balance. The value of parameter M does have an impact on the number of rounds, and it seems that a value equal to 8 should be the preferred choice. None of these parameters has an impact on the number of cluster produced. Concerning *RanMIS*'s comparison with the rest of the protocols, we found that *RanMIS* is the worst when the number of rounds is considered, but the best from the perspective of transmitted messages; its number of rounds and messages increases for larger networks, and drops for denser networks. *RanMIS* has a graceful distribution for the distribution of cluster cardinality, and for the inter-cluster distance, but it produces relatively "long" backbones. Finally, *RanMIS* is quite robust when backbone nodes die being the most resilient protocol for sparse networks, but not the best one for very dense networks, staying behind $WuLi^{\star}$.

RanMIS incorporates some pros that are beneficial to ad hoc networks. A *RanMIS* network topology would encompass considerable availability, because it is efficient in the message complexity that involves backbone construction and maintenance (see Figures 8.20, 8.21). Additionally, it is sustainable, in the sense that it can continue operating for an extended amount of time, compared to its competitors, because of its resilience in node failures (see Figure 8.31). The 'intra-cluster redundancy', as it is described by the network cluster cardinality (Figures 8.23–8.26), is a very useful network attribute, because it allows for adopting a round-robin CH selection method, so as to put some nodes in a sleep mode and thus increase network longevity.

8.4 Conclusion

Ad hoc network node clustering is a very practical and useful topology management approach to reduce the communication overhead and exploit data aggregation in wireless ad hoc networks. Despite the very large literature on the subject, still there is room for the development of protocols that will achieve to run in very few rounds (and thus incur small delay) and at the same time transmit very few messages (thus addressing the broadcast storm problem). In this chapter, we investigated by detailed simulation the performance of *RanMIS* using a new exhaustive evaluation framework. The results confirmed the message-optimality of *RanMIS*, and exhibited its drawback when the number of rounds is concerned based on the original suggestions. Overall, we strongly believe that *RanMIS* could be a viable clustering option for wireless ad hoc networks.

The properties of a network topology constructed by *RanMIS* can be used by applications such as habitat monitoring [96], by disaster relief [20] or law enforcement operations management [30], or even military applications, such as the battle field communications, the early warning, the security surveillance, the identification, the shooter position estimation [128], the battlefield situational awareness [59], the battle damage assessment, the targeting [63], the nuclear-biological-chemical hazard awareness [53], where quasi stationary nodes are involved.

CHAPTER 9

CONCLUSIONS AND FUTURE WORK

In the context of the present dissertation we considered the problem of backbone formation for modern military multilayer ad hoc networks with the purpose to better monitor and / or manage those and improve their performance in terms of their latency, their scalability and their lifetime. We employed tools from graph theory and network science with aim to study network topology and uncover those node characteristics that play crucial role in information flow within the network. We employ a wide range of different multilayer network structures, including vehicular ad hoc networks. Across all the development phases of our work, a common research approach has been followed. Particularly, all the proposed mechanisms where evaluated in widely adopted simulation environments in order to be consistent with the research community, be reproducible and thus provide solid proof of our findings.

In order to improve the performance of military multilayer ad hoc networks we investigated the possibility of forming the backbone in terms of connected node dominating sets, and subsequently we defined – for the first time in the literature – the problem of minimum connected node dominating set for multilayer networks. We proved that decomposition-based and aggregation-based approaches for DS calculation in multilayer networks won't work and highlighted the significance of assessing and exploiting each node's intra- and inter-layer links in order to be considered as candidate members of DS. Thus, we strived for an improved topology management of a military multilayer network by constructing robust $CDSs$ under the concept of influential spreaders.

Our work so far proved that prominence of network nodes cannot be "predicted" by merely measuring the number of connections (degree) incident upon the focal node. Thus, we generalize the well established PCI centrality in the domain of multilayer networks by introducing a number of novel approaches that quantify the importance of a multilayer node. Likewise, the proposed methodology is based on local knowledge of network (and layer) connectivity, that is, at most two hop neighbor related information. We acknowledge that, extending local knowledge of network connectivity beyond two hop neighborhood can provide better awareness to nodes about their "strategic" position in the multilayer ad hoc network, on the other hand, though, it can be prohibited in such a dynamic environment as this can be both compute and communication

intensive. We developed distributed algorithms that based on the proposed centrality metrics outperformed other state-of-the-art competitors regarding various performance measures.

Regarding the computation of a resilient network overlay for communication link monitoring in multilayer ad hoc networks we recognize that it is a hard to address problem, especially using distributed algorithms, because of coordination failures which can lead to loss of communication or over-dependence on a specific layer. We prove that a node to be able to exert sufficient knowledge regarding the multilayer network traffic data, must be well connected to as many layers as possible and interpret this attribute in the proposed schema for identifying influential spreaders in these complex systems.

Motivated by applications in traffic monitoring in diverse communication systems, we considered distributed methods of creating a minimum-size overlay network of monitoring devices over a wireless multilayer ad hoc network. We emphasized the overlay network's resilience to correlated layer failures, paying special attention to inter-layer links, and formalized the problem in terms of Multi-Colored edge dominating sets (EDS) in multilayer networks ($MCMCEDS$). We proved that the problem of finding the MCEDS in a multilayer network is NP-hard. In order to solve it we propose a centrality measure-based technique that provides to nodes awareness about the significance of the edges that are adjacent to them. We propose distributed algorithms that heuristically calculate the $MCMCEDS$ and assess their performance regarding various performance measures.

Focusing on the vehicular ad hoc networking paradigm ($VANETs$) we considered the problem of accurate prediction of vehicular trajectories which is an essential mechanism for ITS. We proved that by maintaining global dictionaries along with individual vehicle profile decoded from updates, it will be possible to predict group behavior. This can lead to better traffic management, and more efficient bandwidth management and quality of service (QoS) in p2p applications. We propose a new trajectory prediction scheme which builds in a purely distributed fashion a rich summary of a vehicle's roaming history that subsequently is used to provide accurate predictions. In the context of military multilayer ad hoc networks trajectory forecasting of mobile nodes that make use of VANETs technology can improve information flow within the network. To elaborate, the respective vehicles present an attractive and viable option so as to be selected as relay nodes from the associated multilayer network routing algorithms.

Emerging advances in network technology are helping to break down the barriers to telecommunications interoperability that have long stymied military planners. Nowadays, nodes are able to integrating multiple waveforms and thus enable interoperability with legacy users and communications with entities using other standards. This evolution predisposes an increased complexity for military multilayer ad hoc networks; growth in colossal sized structures with numerous connections that require advanced handling and analysis; opportunistic connections of mobile nodes that play a fundamental role in dynamical processes, etc. All these considerations confess that establishing, monitoring / maintaining and improving connectivity in future military multilayer ad hoc networks still holds a vast domain of yet undiscovered tasks and thus traditional network theory needs to appropriately adapt and evolve in order to embrace the newly and yet undiscovered needs of these networks.

To this end the analysis of military multilayer ad hoc networks and dynamical processes on these structures will be a core part of our future directions. Understanding the peculiarities of each networked system and further combine those attributes in this multi-structure poses significant challenges. Among the different open problems to be solved, we highlight the following:

(i) The need of setting up novel centrality measures that could possibly better highlight node importance so as to better understand the topology of these unique networks.

(ii) Gathering a solid knowledge and understanding on the interactions between military entities (nodes) that pursue a common operational objective and bind each respective layer separately, as a single network, and how these interactions are adapted and correlated on the whole multilayer structure.

(iii) Establishing approximability results for the $MCMCEDS$ problem.

(iv) The need to control the growth of the respective data structures associated with the trajectory forecasting mechanism in VANETs so as to seamlessly integrate these in the multilayer network environment.

(v) The need to investigate integrating Delay Tolerant Networking (DTN) paradigm as a candidate technology to improve connectivity in the multilayer network environment.

These consideration are only the start line of otherwise endless research directions towards understanding our ever evolving network structures and their applications in our everyday lives.

BIBLIOGRAPHY

[1] *Centrality measures in multilayer networks*, University of Oxford, (2015).

[2] Y. AFEK, N. ALON, Z. BAR-JOSEPH, C. A., B. HAEUPLER, AND F. KUHN, *Beeping a maximal independent set*, in Proceedings of the EATCS International Conference on Distributed Computing (DISC), vol. 6950, 2011, pp. 32–50.

[3] Y. AFEK, N. ALON, O. BARAD, E. HORNSTEIN, N. BARKAI, AND Z. BAR-JOSEPH, *A biological solution to a fundamental distributed computing problem*, Science, 331 (2011), pp. 183–185.

[4] I. F. AKYILDIZ, W. LEE, M. C. VURAN, AND S. MOHANTY, *A survey on spectrum management in cognitive radio networks*, IEEE Communications Magazine, 46 (2008), pp. 40–48.

[5] M. ALMULLAA, K. ABROUGUIB, AND A. A. BOUKERCHE, *LEADMesh: Design and analysis of an efficient leader election protocol for wireless mesh networks*, Simulation Modelling Practice and Theory, 36 (2013), pp. 22–32.

[6] A. AMIS, R. PRAKASH, T. VUONG, AND H. D.T., *Max-min d-cluster formation in wireless ad hoc networks.*, in Proceedings of the IEEE Conference on Computer Communications (INFOCOM), 2000, pp. 32–41.

[7] V. ANITHA AND M. P. SEBASTIAN, *(k,r)-dominating set-based, weighted and adaptive clustering algorithms for mobile ad hoc networks*, IET Communications, 5 (2011), pp. 1836–1853.

[8] F. ATAY, I. STOJMENOVIC, AND H. YANIKOMEROGLU, *Generating random graphs for the simulation of wireless ad hoc, actuator, sensor, and internet networks*, in Proceedings of the IEEE Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2008, pp. 597–615.

[9] S. . BAGLOEE, M. TAVANA, M. ASADI, AND T. OLIVER, *Autonomous vehicles: challenges, opportunities, and future implications for transportation policies*, Journal of Modern Transportation, 24 (2016), pp. 284–303.

[10] S. BASAGNI, *Distributed clustering for ad hoc networks*, in Proceedings of the IEEE International Symposium on Parallel Architectures, Algorithms and Networks (I-SPAN), 1999, pp. 310–315.

[11] S. BASAGNI, M. MASTROGIOVANNI, A. PANCONESI, AND C. PETRIOLI, *Localized protocols for ad-hoc clustering and backbone formation: A performance comparison*, IEEE Transactions on Parallel and Distributed Systems, 17 (2006), pp. 292–306.

[12] P. BASARAS, G. IOSIFIDIS, D. KATSAROS, AND L. TASSIULAS, *Identifying influential spreaders in complex multilayer networks: A centrality perspective*, IEEE Transactions on Network Science and Engineering, (2018).

[13] P. BASARAS, D. KATSAROS, AND T. L., *Detecting influential spreaders in complex, dynamic networks*, IEEE Computer magazine, 46 (2013), pp. 26–31.

[14] E. BERTINO AND N. ISLAM, *Botnets and internet of things security*, Computer, 50 (2017), pp. 76–79.

[15] A. BHATTACHARYA AND S. K. DAS, *Lezi-update: An information-theoretic framework for personal mobility tracking in pcs networks*, Wireless Networks, 8 (2002), pp. 121–135.

[16] S. BOCCALETTI, G. BIANCONI, R. CRIADO, C. I. DEL GENIO, J. GOMEZ GARDENES, M. ROMANCE, I. SENDINA-NADAL, Z. WANG, AND M. ZANIN, *The structure and dynamics of multilayer networks*, Physics Reports, 544 (2014), pp. 1–122.

[17] P. BONACICH AND P. LLOYD, *Eigenvector-like measures of centrality for asymmetric relations*, Social Networks, 23 (2001), pp. 191–201.

[18] R. CARR, T. FUJITO, G. KONJEVOD, AND O. PAREKH, *A 2 1/10-approximation algorithm for a generalization of the weighted edge-dominating set problem*, in Algorithms - ESA 2000, M. S. Paterson, ed., Springer Berlin Heidelberg, 2000, pp. 132–142.

[19] E. CAYIRCI AND I. F. AKYILDIZ, *Optimal location area design to minimize registration signaling traffic in wireless systems*, IEEE Transactions on Mobile Computing, 2 (2003), pp. 76–85.

[20] E. CAYIRCI AND T. COPLU, *SENDROM: Sensor networks for disaster relief operations management*, ACM Wireless Networks, 13 (2007), pp. 409–423.

[21] G.-J. CHANG, C.-W. CHANG, D. KUO, AND S.-H. POON, *Algorithmic aspect of stratified domination in graphs*, 113 (2013), pp. 861—865.

[22] J. H. CHANG AND L. TASSIULAS, *Energy conserving routing in wireless ad-hoc networks*, in Proceedings of the IEEE Conference on Computer Communications (INFOCOM), vol. 1, 2000, pp. 22–31.

[23] M. CHAQFEH, A. LAKAS, AND I. JAWHAR, *A survey on data dissemination in vehicular ad hoc networks*, Vehicular Communications, 1 (2014), pp. 214–225.

[24] G. CHARTRAND, T. W. HAYNES, M. A. HENNING, AND P. ZHANG, *Stratification and domination in graphs*, Discrete Mathematics, 272 (2003), pp. 171–185.

[25] G. CHARTRAND AND M. J. STEWART, *The connectivity of line-graphs*, Mathematische Annalen, 182 (1969), pp. 170–174.

[26] M. CHATTERJEE, S. K. DAS, AND D. TURGUT, *WCA: A weighted clustering algorithm for mobile ad-hoc networks*, Journal of Cluster Computing, 5 (2002), pp. 193–204.

[27] B. CHEN, K. JAMIESON, H. BALAKRISHNAN, AND R. MORRIS, *Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks*, ACM Wireless Networks, 8 (2002), pp. 481–494.

[28] M. CHLEBIK AND J. CHLEBIKOVA, *Approximation hardness of dominating set problems in bounded degree graphs*, Information and Computation, 206 (2008), pp. 1264–1275.

[29] N. CHOUBEY AND S. RAO, *Topology control in wireless sensor networks*, in Proceedings of the IARIA Sensor Technologies and Applications (SENSORCOMM), 2009, pp. 339–345.

[30] C. CHUNG-KUO AND J. HUANG, *Video surveillance for hazardous conditions using sensor networks*, in Proceedings of the IEEE International Conference on Networking, Sensing and Control (ICNSC), 2004, pp. 1008–1013.

[31] P. CRUCITTI, V. LATORA, M. MARCHIORI, AND A. RAPISARDA, *Error and attack tolerance of complex networks*, Physica A: Statistical Mechanics and its Applications, 340 (2004), pp. 388–394.

[32] A. CUZZOCREA, A. PAPADIMITRIOU, D. KATSAROS, AND Y. MANOLOPOULOS, *Edge betweenness centrality: A novel algorithm for QoS-based topology control over wireless sensor networks*, Journal of Network and Computer Applications, 35 (2012), pp. 1210–1217.

[33] F. DAI AND J. WU, *An extended localized algorithm for connected dominating set formation in ad-hoc wireless networks*, IEEE Transactions on Parallel and Distributed Systems, 15 (2004), pp. 908–920.

[34] B. DAS AND V. BHARGHAVAN, *Routing in ad hoc networks using minimum connected dominating sets*, in Proceedings of the IEEE International Conference on Communications (ICC), 1997, pp. 376–380.

[35] M. DE DOMENICO, A. SOLÉ-RIBALTA, E. OMODEI, S. GÓMEZ, AND A. ARENAS, *Ranking in interconnected multilayer networks reveals versatile nodes*, Nature Communications, 6 (2015), p. 6868.

[36] N. DIMOKAS AND D. KATSAROS, *Detecting energy-efficient central nodes for cooperative caching in wireless sensor networks*, in Proceedings of the IEEE Advanced Information Networking and Applications (AINA).

[37] N. DIMOKAS, D. KATSAROS, AND Y. MANOLOPOULOS, *Energy-efficient distributed clustering in wireless sensor networks*, Journal of Parallel and Distributed Computing, 70 (2010), pp. 371–383.

[38] N. DIMOKAS, D. KATSAROS, L. TASSIULAS, AND Y. MANOLOPOULOS, *High performance, low complexity cooperative caching for wireless sensor networks*, ACM / Springer Wireless Networks, 17 (2011), pp. 717–737.

[39] D.-Z. DU AND P.-J. WAN, *Connected Dominating Set: Theory and Applications*, vol. 77, Springer, 2013.

[40] J. DUAN AND H. Z. DEYUN GAO, CHUAN HENG FOH, *TC-BAC: A trust and centrality degree based access control model in wireless sensor networks*, Ad Hoc Networks, 11 (2013), pp. 2675–2692.

[41] M. EIDSAA AND E. ALMAAS, *s–core network decomposition: A generalization of k-core analysis to weighted networks*, Physical Review E, 88 (2013), pp. 1–9.

[42] K. ERCIYES, O. DAGDEVIREN, D. COKUSLU, AND D. OZSOYELLER, *Graph-theoretic clustering algorithms in mobile ad hoc networks and wireless sensor networks - Survey*, Applied and Computational Mathematics, 6 (2007), pp. 162–180.

[43] S. ESHGHI, M. H. R. KHOUZANI, S. SARKAR, N. B. SHROFF, AND S. S. VENKATESH, *Optimal energy-aware epidemic routing in dtns*, IEEE Transactions on Automatic Control, 60 (2015), pp. 1554–1569.

[44] M. FEDER, N. MERHAV, AND M. GUTMAN, *Universal prediction of individual sequences*, IEEE Transactions on Information Theory, 38 (1992), pp. 1258–1270.

[45] C. E. FOSSA AND T. G. MACDONALD, *Internetworking tactical MANETs*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2010.

[46] L. C. FREEMAN, S. P. BORGATTI, AND D. R. WHITE, *Centrality in valued graphs: A measure of betweenness based on network flow*, Social Networks, 13 (1991), pp. 141–154.

[47] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freenan and Company, 1979.

[48] M. GERLA AND J.-C. TSAI, *Multicluster, mobile, multimedia radio network*, ACM/Springer Wireless Networks, 1 (1995), pp. 255–265.

[49] J. GHADERI, L.-L. XIE, AND X. SHEN, *Hierarchical cooperation in ad hoc networks: Optimal clustering and achievable throughput*, IEEE Transactions on Information Theory, 55 (2009), pp. 3425–3436.

[50] A. A. GOHARI, R. PAKBAZ, P. M. MELLIAR-SMITH, L. E. MOSER, AND V. RODOPLU, *RMR: Reliability Map Routing for tactical mobile ad hoc networks*, IEEE Journal on Selected Areas in Communications, 29 (2011), pp. 1935–1947.

[51] D. J. GOPALRATNAM, K.AND COOK, *Online sequential prediction via incremental parsing: The active lezi algorithm*, IEEE Intelligent Systems, 22 (2007), pp. 52–58.

[52] P. GUPTA AND P. R. KUMAR, *The capacity of wireless networks*, IEEE Transactions on Information Theory, 46 (2000), pp. 388–404.

[53] J. K. HART AND K. MARTINEZ, *Environmental sensor networks: A revolution in the earth system science?*, Earth-Science Reviews, 78 (2006), pp. 177–191.

[54] T. W. HAYNES, S. HEDETNIEMI, AND P. SLATER, *Fundamentals of Domination in Graphs*, Chapman & Hall/CRC Pure and Applied Mathematics, CRC Press, 1998.

[55] C. HE, H. MA, S. KANG, AND R. CUI, *An overlapping community detection algorithm based on link clustering in complex networks*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2014, pp. 865–870.

[56] W. HEINZELMAN, A. CHANDRAKASAN, AND H. BALAKRISHNAN, *Energy-efficient communication protocol for wireless microsensor networks*, in Proceedings of the Hawaii International Conference on System Sciences (HICSS), vol. 8, 2000.

[57] W. B. HEINZELMAN, A. P. CHANDRAKASAN, AND H. BALAKRISHNAN, *An application-specific protocol architecture for wireless microsensor networks*, IEEE Transactions on Wireless Communications, 1 (2002), pp. 660–670.

[58] A. HOUENOU, P. BONNIFAIT, V. CHERFAOUI, AND W. YAO, *Vehicle trajectory prediction based on motion model and maneuver recognition*, in Proceedings of the IEEE/RSJ Intelligent Robots and Systems (IROS), 2013, pp. 4363–4369.

[59] G. HUA, Y.-X. LI, AND X.-M. YAN, *Research on the wireless sensor networks applied in the battlefield situation awareness system*, in Advanced Research on Electronic Commerce, Web Application, and Communication, vol. 144, 2011, pp. 443–449.

[60] J. HUANG AND H. S. TAN, *Vehicle future trajectory prediction with a DGPS/INS-based positioning system*, in Proceedings of the IEEE American Control Conference (ACC), 2006.

[61] V. N. IOANNIDIS, P. A. TRAGANITIS, Y. SHEN, AND G. B. GIANNAKIS, *Kernel-based semi-supervised learning over multilayer graphs*, in Proceedings of the IEEE Signal Processing Advances in Wireless Communications (SPAWC), vol. 06, 2018.

[62] L. G. S. JEUB, M. W. MAHONEY, P. J. MUCHA, AND M. A. PORTER, *A local perspective on community structure in multilayer networks*, Network Science, (2015).

[63] X. JI AND H. ZHA, *Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling*, in Proceedings of the IEEE Conference on Computer Communications (INFOCOM), 2004, pp. 2652–2661.

[64] O. KAIWARTYA AND S. KUMAR, *Geocast routing: Recent advances and future challenges in vehicular adhoc networks*, in Proceedings of the IEEE Signal Processing and Integrated Networks (SPIN), 2014, pp. 291–296.

[65] D. KATSAROS, N. DIMOKAS, AND L. TASSIULAS, *Social network analysis concepts in the design of wireless ad hoc network protocols*, IEEE Network magazine, 24 (2010), pp. 23–29.

[66] D. KATSAROS AND Y. MANOLOPOULOS, *Prediction in wireless networks by markov chains*, IEEE Wireless Communications, 16 (2009), pp. 56–64.

[67] L. KATZ, *A new status index derived from sociometric analysis*, Psychometrika, 18 (1953), pp. 39–43.

[68] H. KAZEMI, G. HADJICHRISTOFI, AND L. A. DASILVA, *Mman - a monitor for mobile ad hoc networks: Design, implementation, and experimental evaluation*, in Proceedings of the ACM ACM International Workshop on Wireless Network Testbeds, Experimental evaluation & CH (WiNTECH), 2008, pp. 57–64.

[69] B. KIMURA, C. CARDEN, AND R. NORTH, *Joint tactical radio system - empowering the warfighter for joint vision 2020*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2008.

[70] T. KIMURA, T. MATSUDA, AND T. TAKINE, *Probabilistic store-carry-forward message delivery based on node density estimation*, in Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC), 2014, pp. 432–437.

[71] M. KITSAK, L. K. GALLOS, S. HAVLIN, F. LILJEROS, L. MUCHNIK, H. E. STANLEY, AND H. A. MAKSE, *Identification of influential spreaders in complex networks*, Nature Physics, 6 (2010), pp. 888–893.

[72] P. KOLIOS, V. FRIDERIKOS, AND K. PAPADAKI, *Future wireless mobile networks*, IEEE Vehicular Technology Magazine, 6 (2011), pp. 24–30.

[73] S. LEE, S. H. YOOK, AND Y. KIM, *Centrality measure of complex networks using biased random walks*, The European Physical Journal B, 68 (2009), pp. 277–281.

[74] S. LENG, Y. ZHANG, H.-W. CHEN, L. ZHANG, AND K. LIU, *A novel k-hop compound metric based clustering scheme for ad hoc wireless networks*, IEEE Transactions on Wireless Communications, 8 (2009), pp. 367–375.

[75] L. LI AND J. Y. HALPERN, *Minimum-energy mobile wireless networks revisited*, in Proceedings of the IEEE International Conference on Communications (ICC), vol. 1, 2001, pp. 278–283.

[76] L. MACCARI AND R. L. CIGNO, *Pop-routing: Centrality-based tuning of control messages for faster route convergence*, in Proceedings of the IEEE International Conference on Computer Communications (INFOCOM), 2016.

[77] L. MACCARI, Q. NGUYEN, AND R. L. CIGNO, *On the computation of centrality metrics for network security in mesh networks*, in Proceedings of the IEEE Global Communications Conference (GLOBECOM), 2016.

[78] J. P. MACKER, B. ADAMSON, AND D. J. CLAYPOOL, *Temporal stability for dynamic network relay sets*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2012.

[79] J. P. MACKER AND D. J. CLAYPOOL, *Dynamic communities in evolving network graphs*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2012.

[80] J. P. MACKER AND I. J. TAYLOR, *Prediction and planning of distributed task management using network centrality*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2014, pp. 859–864.

[81] N. MAGAIA, A. P. FRANCISCO, P. PEREIRA, AND M. CORREIA, *Betweenness centrality in Delay Tolerant Networks: A survey*, Ad Hoc Networks, 33 (2015), pp. 284–305.

[82] L. MAGLARAS AND D. KATSAROS, *Distributed clustering in vehicular networks*, in Proceedings of the IEEE Vehicular Communications and Networking (VECON), 2012, pp. 601–607.

[83] ——, *Social clustering of vehicles based on semi-markov processes*, IEEE Transactions on Vehicular Technology, 65 (2016), pp. 318–332.

[84] J. MCCOY AND M. A. HENNING, *Locating and paired-dominating sets in graphs*, Discrete Applied Mathematics, 157 (2009), pp. 3268–3280.

[85] L. MICHAELRAJ, S. AYYASWAMY, AND S. ARUMUGAM, *Chromatic transversal domination in graphs*, 75 (2010), pp. 33–40.

[86] K. MIKKO, A. ARENAS, M. BARTHELEMY, J. P. GLEESON, Y. MORENO, AND M. A. PORTER, *Multilayer networks*, Journal of Complex Networks, 2 (2014), pp. 203–271.

[87] A. MUNARO, *On some classical and new hypergraph invariants*, PhD thesis, 2016.

[88] M. NI, Z. ZHONG, AND D. ZHAO, *MPBC: A mobility prediction-based clustering scheme for ad hoc networks*, IEEE Transactions on Vehicular Technology, 60 (2011), pp. 4549–4559.

[89] F. A. ONAT AND I. STOJMENOVIC, *Generating random graphs for wireless actuator networks*, in Proceedings of IEEE WoWMoM, 2007.

[90] T. OPSAHL, F. AGNEESSENS, AND J. SKVORETZ, *Node centrality in weighted networks: Generalizing degree and shortest paths*, Social Networks, 32 (2010), pp. 245–251.

[91] L. PAGE, S. BRIN, R. MOTWANI, AND T. WINOGRAD, *The pagerank citation ranking: Bringing order to the web*, 1999.

[92] D. PAPAKOSTAS, P. BASARAS, D. KATSAROS, AND L. TASSIULAS, *Backbone formation in military multi-layer ad hoc networks using complex network concepts*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2016, pp. 842–848.

[93] D. PAPAKOSTAS, S. ESHGHI, D. KATSAROS, AND L. TASSIULAS, *Energy-aware backbone formation in military multilayer ad hoc networks*, Ad Hoc Networks, 81 (2018), pp. 17–44.

[94] ——, *Distributed algorithms for multi-layer connected edge dominating sets*, IEEE Control Systems Letters, 3 (2019), pp. 31–36.

[95] S. U. PILLAI, T. SUEL, AND S. CHA, *The perron-frobenius theorem: Some of its applications*, IEEE Signal Processing Magazine, 22 (2005), pp. 62–75.

[96] J. POLASTRE, R. SZEWCZYK, A. MAINWARING, D. CULLER, AND J. ANDERSON, *Analysis of wireless sensor networks for habitat monitoring*, in Wireless Sensor Networks, C. S. Raghavendra, K. M. Sivalingam, and T. Znati, eds., 2004, pp. 399–423.

[97] K. POULARAKIS, G. IOSIFIDIS, AND L. TASSIULAS, *SDN-enabled Tactical Ad Hoc Networks: Extending Programmable Control to the Edge*, CoRR, abs/1801.02909 (2018).

[98] S. K. PULLIYAKODE AND S. KALYANI, *A modified PPM algorithm for online sequence prediction using short data records*, IEEE Communications Letters, 19 (2015), pp. 423–426.

[99] A. QAYYUM, L. VIENNOT, AND A. LAOUITI, *Multipoint relaying: An efficient technique for flooding in mobile wireless networks*, Tech. Rep. 3898, INRIA, March 2000.

[100] ——, *Multipoint relaying for flooding broadcast messages in mobile wireless networks*, in Proceedings of the IEEE Hawaii International Conference on System Sciences (HICSS), 2002, pp. 3866–3875.

[101] A. RODRIGUEZ-CARRION, C. GARCIA-RUBIO, AND C. CAMPO, *Performance evaluation of lz-based location prediction algorithms in cellular networks*, IEEE Communications Letters, 14 (2010), pp. 707–709.

[102] R. ROUSSEAU AND L. ZHANG, *Betweenness centrality and q-measures in directed valued networks*, Scientometrics, 75 (2008), pp. 575–590.

[103] I. RUBIN, A. BAIOCCHI, F. CUOMO, AND P. SALVO, *Vehicular backbone network approach to vehicular military ad hoc networks*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2013, pp. 901–909.

[104] P. RUIZ AND P. BOUVRY, *Survey on broadcast algorithms for mobile ad hoc networks*, ACM Computing Surveys, 48 (2015), pp. 1–35.

[105] Y. E. SAGDUYU, Y. SHI, T. ERPEK, S. SOLTANI, S. J. MACKEY, D. H. CANSEVER, M. P. PATEL, B. F. PANETTIERI, B. K. SZYMANSKI, AND G. CAO, *Multilayer manet routing with social-cognitive learning*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2017, pp. 103–108.

[106] M. SALEHI, R. SHARMA, M. MARZOLLA, M. MAGNANI, P. SIYARI, AND D. MONTESI, *Spreading processes in multilayer networks*, IEEE Transactions on Network Science and Enginering, 2 (2015), pp. 65–83.

[107] R. M. SALLES AND D. A. J. DONATO A. MARINO, *Strategies and metric for resilience in computer networks*, The Computer Journal, 55 (2012), pp. 728–739.

[108] C. U. SARAYDAR, O. E. KELLY, AND C. ROSE, *One-dimensional location area design*, IEEE Transactions on Vehicular Technology, 49 (2000), pp. 1626–1632.

[109] L. SCHIAVONE, N. BROWNE, R. NORTH, L. SCHIAVONE, N. BROWNE, AND R. NORTH, *Joint Tactical Radio System - Connecting the GIG to the Tactical Edge*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2006, pp. 1–6.

[110] M. SCHREIER, V. WILLERT, AND J. ADAMY, *Bayesian, maneuver-based, long-term trajectory prediction and criticality assessment for driver assistance systems*, in Proceedings of the IEEE Intelligent Transportation Systems Conference (ITSC), 2014, pp. 334–341.

[111] E. M. SHAHRIVAR AND S. SUNDARAM, *The strategic formation of multi-layer networks*, IEEE Transactions on Network Science and Enginering, 2 (2015), pp. 164–178.

[112] R. SIMMONS, B. BROWNING, Y. ZHANG, AND V. SADEKAR, *Learning to predict driver route and destination intent*, in Proceedings of the IEEE Intelligent Transportation Systems Conference (ITSC), 2006, pp. 127–132.

[113] P. J. SLATER, *Colored problems in graphs*, AKCE International Journal of Graphs and Combinatorics, 4 (2007), pp. 191–196.

[114] A. SOLÉ-RIBALTA, M. DE DOMENICO, S. GÓMEZ, AND A. ARENAS, *Centrality rankings in multiplex networks*, in Proceedings of the ACM Conference on Web Science, 2014, pp. 149–155.

[115] L. SONG, D. KOTZ, R. JAIN, AND X. HE, *Evaluating next-cell predictors with extensive wi-fi mobility data*, IEEE Transactions on Mobile Computing, 5 (2006), pp. 1633–1649.

[116] R. SONG, J. D. BROWN, P. C. MASON, M. SALMANIAN, AND H. TANG, *HMS: Holistic MPR selection and network connectivity for tactical edge networks*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2015, pp. 726–731.

[117] J. SPENCER AND T. WILLINK, *SDN in Coalition Tactical Networks*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2016, pp. 1053–1058.

[118] I. STOJMENOVIC, M. SEDDIGH, AND J. ZUNIC, *Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks*, IEEE Transactions on Parallel and Distributed Systems, 13 (2002), pp. 14–25.

[119] J. SUM, J. WU, AND K. I.-J. HO, *Analysis on a localized pruning method for connected dominating sets*, Journal of Information Science and Engineering, 23 (2007), pp. 1073–1086.

[120] Q. TANG, K. YANG, P. LI, J. ZHANG, Y. LUO, AND B. XIONG, *An energy efficient MCDS construction algorithm for wireless sensor networks*, EURASIP Journal on Wireless Communications and Networking, (2012), p. 83.

[121] T. TERRIFF, *The past as future: The US army's vision of warfare in the 21st century*, Journal of Military and Strategic Studies, 15 (2014).

[122] P. THULASIRAMAN AND K. A. WHITE, *Topology control of tactical wireless sensor networks using energy efficient zone routing*, Digital Communications and Networks, 2 (2016), pp. 1–14.

[123] F. TOSHIHIRO, *On approximability of the independent/connected edge dominating set problems*, Information Processing Letters, 79 (2001), pp. 261–266.

[124] P. A. TRAGANITIS, Y. SHEN, AND G. B. GIANNAKIS, *Topology inference of multilayer networks*, in Proceedings of the IEEE Computer Communications Workshops (INFOCOM WKSHPS), 2017, pp. 1–14.

[125] C. TSELIKIS, S. MITROPOULOS, N. KOMNINOS, AND C. DOULIGERIS, *Degree-based clustering algorithms for wireless ad hoc networks under attack*, IEEE Communications Letters, 16 (2012), pp. 619–621.

[126] A. VAHEDIAN, X. ZHOU, L. TONG, Y. LI, AND J. LUO, *Forecasting gathering events through continuous destination prediction on big trajectory data*, in Proceedings of the ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL), 2017.

[127] S. VODOPIVEC, J. BESTER, AND A. KOS, *A survey on clustering algorithms for vehicular ad-hoc networks*, in Proceedings of the International Conference on Telecommunications and Signal Processing (TSP), 2012, pp. 52–56.

[128] P. VOLGYESI, G. BALOGH, A. NADAS, C. B. NASH, AND A. LEDECZI, *Shooter localization and weapon classification with soldier-wearable networked sensors*, in Proceedings of the ACM International Conference on Mobile Systems, Applications and Services (MobiSys), 2007, pp. 113–126.

[129] P.-J. WAN, K. M. ALZOUBI, AND O. FRIEDER, *Distributed construction of connected dominating set in wireless ad hoc networks*, ACM/Springer Mobile Networks and Applications, 9 (2004), pp. 141–149.

[130] F. WANG, H. DU, E. CAMACHO, K. XU, W. LEE, Y. SHI, AND S. SHAN, *On positive influence dominating sets in social networks*, Theoretical Computer Science, 412 (2011), pp. 265–269.

[131] N. WANG, D. C. SCHMIDT, H. VAN'T HAG, AND A. CORSARO, *Toward an adaptive data distribution service for dynamic large-scale network-centric operation and warfare (NCOW) systems*, in Proceedings of the IEEE Military Communications Conference (MILCOM), 2008, pp. 1–7.

[132] K. WASSERMAN, S.AND FAUST, *Social Network Analysis (Methods and Applications) || Social Network Analysis in the Social and Behavioral Sciences*, vol. 10.1017/CBO9780511815478, 1994.

[133] J. WU AND F. DAI, *Broadcasting in ad hoc networks based on self-pruning*, in Proceedings of the IEEE Conference on Computer Communications (INFOCOM), vol. 3, 2003, pp. 2240–2250.

[134] J. WU AND H. LI, *On calculating connected dominating set for efficient routing in ad hoc wireless networks*, in Proceedings of the AMC Discrete Algoritms and Methods for MOBILE Computing and Communications (DialM), 1999, pp. 7–14.

[135] ——, *A dominating-set-based routing scheme in ad hoc wireless networks*, Telecommunication Systems, 18 (2001), pp. 13–36.

[136] Y. XU, J. HEIDEMANN, AND D. ESTRIN, *Geography-informed energy conservation for ad hoc routing*, in Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom), 2001, pp. 70–84.

[137] A. Y. XUE, R. ZHANG, Y. ZHENG, X. XIE, J. HUANG, AND Z. XU, *Destination prediction by sub-trajectory synthesis and privacy protection against such prediction*, in Proceedings of IEEE International Conference on Data Engineering (ICDE), 2013.

[138] H. YANG, X. MENG, AND S. LU, *Self-organized network-layer security in mobile ad hoc networks*, in Proceedings of the ACM workshop on Wireless Security (WiSe), 2002, pp. 11–20.

[139] M. YANNAKAKIS AND F. GAVRIL, *Edge dominating sets in graphs*, SIAM Journal on Applied Mathematics, 38 (1980), pp. 364–372.

[140] M. YOUNIS, I. F. SENTURK, K. AKKAYA, S. LEE, AND F. SENEL, *Topology management techniques for tolerating node failures in wireless sensor networks: A survey*, Computer Networks, 58 (2014), pp. 254–283.

[141] O. YOUNIS AND S. FAHMY, *HEED: A hybrid, energy-efficient, distributed clustering approach for ad-hoc sensor networks*, IEEE Transactions on Mobile Computing, 3 (2004), pp. 366–379.

[142] O. YOUNIS, M. KRUNZ, AND S. RAMASUBRAMANIAN, *Node clustering in wireless sensor networks: Recent developments and deployment challenges*, IEEE Network magazine, 20 (2006), pp. 20–25.

[143] J. YU, N. WANG, G. WANG, AND D. YU, *Connected dominating sets in wireless ad hoc and sensor networks: A comprehensive survey*, Computer Communications, 24 (2013), pp. 121–134.

[144] J. Y. YU AND P. H. J. CHONG, *A survey of clustering schemes for mobile ad hoc networks*, IEEE Communications Surveys and Tutorials, 7 (2005), pp. 32–48.

[145] J. Y. YU AND P. J. CHONG, *A survey of clustering schemes for mobile ad hoc networks*, IEEE Communications Surveys Tutorials, 7 (2005), pp. 32–48.

[146] J. ZIV AND A. LEMPEL, *Compression of individual sequences via variable-rate coding*, IEEE Transactions on Information Theory, 24 (1978), pp. 530–536.