STATUS OF THESIS

Title of thesis	Localized Movement Control Connectivity Restoration algorithms for Wireless Sensor and Actor Networks

I MUHAMMAD IMRAN

hereby allow my thesis to be placed at the Information Resource Center (IRC) of Universiti Teknologi PETRONAS (UTP) with the following conditions:

1. The thesis becomes the property of UTP
2. The IRC of UTP may make copies of the thesis for academic purposes only.
3. This thesis is classified as

☐ Confidential

☐ Non-confidential

If this thesis is confidential, please state the reason:

____

____

____

The contents of the thesis will remain confidential for ___________ years.

Remarks on disclosure:

____

____

____

Endorsed by

______________________________ ______________________________

Signature of Author

Permanent address:
H. No. 348, St#10,
Ghauri Town Phase-III, Near
Gangal East, Islamabad, Pakistan.

Date: ________________________

Signature of Supervisor

Name of Supervisor
Assoc. Prof. Dr. Abas Md. Said

Date: ________________________

UNIVERSITI TEKNOLOGI PETRONAS

DISSERTATION TITLE: LOCALIZED MOVEMENT CONTROL CONNECTIVITY RESTORATION ALGORITHMS WIRELESS SENSOR AND ACTOR NETWORKS

By

MUHAMMAD IMRAN

The undersigned certify that they have read, and recommend to the Postgraduate Studies Program for acceptance of this thesis for the fulfilment of the requirements for the degree stated.

Signature:	______________________
Main Supervisor:	Assoc. Prof. Dr. Abas Md. Said
Signature:	______________________
Co-Supervisor:	Dr. Halabi Hasbullah
Signature:	______________________
Field-Supervisor:	Assoc. Prof. Dr. Mohamed Younis
Signature:	______________________
Head of Department:	Dr. Mohd. Fadzil B. Hassan
Date:	______________________

DISSERTATION TITLE: LOCALIZED MOVEMENT CONTROL CONNECTIVITY RESTORATION ALGORITHMS FOR WIRELESS SENSOR AND ACTOR NETWORKS

By

MUHAMMAD IMRAN

A Thesis

Submitted to the Postgraduate Studies Programme

as a Requirement for the Degree of

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCES

UNIVERSITI TEKNOLOGI PETRONAS

BANDAR SERI ISKANDAR,

PERAK

MAY 2011

DECLARATION OF THESIS

Title of thesis	Localized Movement Control Connectivity Restoration Algorithms for Wireless Sensor and Actor Networks

I MUHAMMAD IMRAN

hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTP or other institutions.

Witnessed by

______________________	______________________
Signature of Author	Signature of Supervisor
Permanent address:	Name of Supervisor
H. No. 348, St#10,	Assoc. Prof. Dr. Abas Md. Said
Ghauri Town Phase-III, Near	
Gangal East, Islamabad, Pakistan.	
Date : ______________________	Date : ______________________

DEDICATION

I dedicate this research work to the ***Holy Prophet Muhammad (Peace be upon him),*** to his ***Progeny,*** to his ***Companions,*** to my loving parents, *to my beloved wife and daughter.* I also dedicate this research work to my sister and to my brothers and ***to all my friends and family*** for their love, encouragement and great moral support.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank ***Almighty Allah*** for giving me this wonderful opportunity, patience and the intellectual ability to complete this research work. All praises be to Allah for His majesty, bestowal of favors, knowledge, ability, wisdom and power. I would not have been able to successfully complete this dissertation without the blessings of Allah.

I would like to express my sincere gratitude to my supervisor, ***Associate Professor Dr. Abas Md. Said*** for putting me on this research track and for his enduring professional guidance throughout my research work. I am grateful to my co-supervisor ***Dr. Halabi Hasbullah*** for his help, guidance and wonderful scientific supervision. I owe a special word of gratitude to my field supervisor, ***Associate Professor Dr. Mohamed Younis,*** for his insightful guidance and keen advice.

I especially want to thank Universiti Teknologi PETRONAS (UTP) which supported me financially without which this research work would have been impossible. The support of all the staff from different departments especially the Computer and Information Sciences, Postgraduate office and Student Support Services are also gratefully acknowledged. All the staff have been very cooperative, always welcomed me with smiling faces and co-operated up to their best possible level.

I am thankful to my brothers, sister, all family members, friends, colleagues, relatives and my well-wishers who prayed for my success. My gratitude also goes out to all my colleagues in UTP for their occasional help. Thanks to anybody whom I may have unintentionally missed who deserves a mention!

Finally, and most importantly, my ultimate gratitude goes to my parents, who have always prayed for me, guided me, helped me and supported me throughout my life. I am, furthermore, extremely grateful from the core of my heart to my beloved

wife who has always been patient and supportive of me. I cannot forget my daughter, raising her small hands before Allah and requesting for my success. But just saying thanks to my wife and daughter is not enough. May Allah give them blessings in this world and the hereafter.

May Allah reward all those who helped me and prayed for my success with His bountiful blessings. Amin.

ABSTRACT

Wireless Sensor and Actor Networks (WSANs) are gaining an increased interest because of their suitability for mission-critical applications that require autonomous and intelligent interaction with the environment. Hazardous application environments such as forest fire monitoring, disaster management, search and rescue, homeland security, battlefield reconnaissance, etc. make actors susceptible to physical damage. Failure of a critical (i.e. cut-vertex) actor partitions the inter-actor network into disjointed segments while leaving a coverage hole. Maintaining inter-actor connectivity is extremely important in mission-critical applications of WSANs where actors have to quickly plan an optimal coordinated response to detected events. Some proactive approaches pursued in the literature deploy redundant nodes to provide fault tolerance; however, this necessitates a large actor count that leads to higher cost and becomes impractical. On the other hand, the harsh environment strictly prohibits an external intervention to replace a failed node. Meanwhile, reactive approaches might not be suitable for time-sensitive applications. The autonomous and unattended nature of WSANs necessitates a self-healing and agile recovery process that involves existing actors to mend the severed inter-actor connectivity by reconfiguring the topology. Moreover, though the possibility of simultaneous multiple actor failure is rare, it may be precipitated by a hostile environment and disastrous events. With only localized information, recovery from such failures is extremely challenging. Furthermore, some applications may impose application-level constraints while recovering from a node failure.

In this dissertation, we address the challenging connectivity restoration problem while maintaining minimal network state information. We have exploited the controlled movement of existing (internal) actors to restore the lost connectivity while minimizing the impact on coverage. We have pursued distributed greedy heuristics.

This dissertation presents four novel approaches for recovering from node failure. In the first approach, volunteer actors exploit their partially utilized transmission power and reposition themselves in such a way that the connectivity is restored. The second

approach identifies critical actors in advance, designates them preferably as non-critical backup nodes that replace the failed primary if such contingency arises in the future. In the third approach, we design a distributed algorithm that recovers from a special case of multiple simultaneous failures. The fourth approach factors in application-level constraints on the mobility of actors while recovering from node failure and strives to minimize the impact of critical node failure on coverage and connectivity. The performance of proposed approaches is analyzed and validated through extensive simulations. Simulation results confirm the effectiveness of proposed approaches that outperform the best contemporary schemes found in literature.

ABSTRAK

Sensor Tanpa Wayar dan Rangkaian Tindakan atau "*Wireless Sensor and Actor Networks"* (WSANs) sedang mendapat kepentingan yang semakin meluas kerana kesesuaian mereka untuk aplikasi misi yang kritikal yang memerlukan autonomi dan kebijaksanaan interaksi dengan persekitaran. Persekitaran aplikasi yang berbahaya seperti pemantauan kebakaran hutan, pengurusan bencana, mencari dan menyelamat, keselamatan tanah air, pengintipan medan perang, dll membuatkan pelaku terdedah kepada kecederaan fizikal. Kegagalan sesebuah pelaku yang kritis (iaitu "cut-vertex") membahagikan rangkaian antara pelaku-pelaku kepada segmen-segmen beririsan selain meninggalkan lubang liputan. Mempertahankan kelangsungan antara pelaku sangat penting dalam aplikasi misi yang kritikal dalam WSANs apabila pelaku harus cepat merancang tindakan terkoordinasi yang optimum untuk situasi yang telah dikenal-pasti. Beberapa pendekatan proaktif digunakan dalam kesusasteraan menyebarkan nod berlebihan untuk memberikan toleransi kesalahan, namun ini memerlukan jumlah pelaku yang besar dengan penglibatan kos yang lebih tinggi dan menjadikannya tidak praktikal. Dari sudut pandangan yang berbeza, persekitaran yang sukar melarang keras campur tangan luaran untuk menggantikan nod yang gagal. Sementara itu, pendekatan pengaktifan semula mungkin tidak sesuai untuk aplikasi yang sensitif terhadap masa. Sifat autonomi dan tanpa pengawasan dari WSANs memerlukan penyembuhan-diri dan tangkas dalam proses pemulihan yang melibatkan pelaku sedia ada untuk memperbaiki kesambungan antara pelaku yang terputus melalui penyediaan semula topologi. Selain itu, walaupun kemungkinan kegagalan pelaku secara serentak amat jarang, ia mungkin dipercepat oleh persekitaran yang bermusuhan dan kejadian bencana. Dengan maklumat tempatan, pemulihan dari kegagalan tersebut sangat mencabar. Selain itu, beberapa aplikasi mungkin mempunyai sekatan tahap-aplikasi ketika pulih daripada kegagalan nod.

Dalam disertasi ini, kami mengatasi masalah kesambungan pemulihan yang mencabar ketika mana hanya memelihara keadaan maklumat rangkaian tempatan. Kami telah mengeksploitasi gerakan terkawal yang pelaku sedia ada (dalaman) untuk mengembalikan kelangsungan yang hilang sambil meminimumkan kesan terhadap liputan. Kami telah menggunakan agihan heuristik serakah (tamak).

Disertasi ini menyajikan empat pendekatan baru untuk pulih daripada kegagalan nod. Pada pendekatan pertama, pelaku sukarelawan mengeksploitasi sebahagian penggunaan kuasa penghantaran mereka dan reposisi diri sehingga sambungan dipulihkan. Pendekatan kedua mengenalpasti pelaku kritis dengan lebih awal, melabelkan mereka sebaiknya nod simpanan yan tidak kritikal yang menggantikan kegagalan utama jika kontingensi tersebut muncul di masa depan. Dalam pendekatan ketiga, kami merekabentuk sebuah algoritma teragih yang pulih daripada kegagalan serentak (kes khas). Pendekatan faktor keempat dalam tahap aplikasi menghalang mobiliti pelaku sementara pulih daripada kegagalan nod dan berupaya untuk meminimumkan kesan daripada kegagalan nod kritis dalam liputan dan kelangsungan. Prestasi pendekatan yang dicadangkan dianalisa dan disahkan melalui pelbagai jenis simulasi yang luas. Keputusan simulasi mengesahkan keberkesanan pendekatan yang dicadangkan mengatasi skim kontemporari yang ditemui sebelum ini dalam kesusasteraan.

.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACR	Application-centric Connectivity Restoration
AUV	Autonomous Underwater Vehicle
C^2AM	Connectivity with application level Constraints on Actors Mobility
C^2AP	Coverage-aware Connectivity-constrained Actor Positioning
CDS	Connected Dominating Set
C^3R	Coverage-conscious Connectivity Restoration
COCOLA	COnnected COLA
COLA	COverage and Latency aware Actor placement
CORE	COordinated RElocation
DARA	Distributed Actor Recovery Algorithm
DCR	partitioning Detection and Connectivity Restoration
DFS	Depth First Search
MPADRA	Multiple PADRA
MRN	Mobile Robot Network
MSN	Mobile Sensor Network
PADRA	PArtition Detection and Recovery Algorithm
PCR	Partitioning detection and Connectivity Restoration
RAM	Recovery Algorithm to handle Multiple failures
RIM	Recovery through Inward Motion
SNR	Signal-to-noise ratio
UAV	Unmanned Air Vehicle
UGV	Unmanned Ground Vehicle

USAR	Urban Search and Rescue
VCR	Volunteer-instigated Connectivity Restoration
WSAN	Wireless Sensor and Actor Network
WSN	Wireless Sensor Network

CHAPTER 1

INTRODUCTION

This chapter begins with a brief introduction of wireless sensor and actor networks (WSANs) and their applications. The words “dissertation” and “thesis” will be used interchangeably throughout this thesis as well as the words “actor (s)” and “actor nodes”. The chapter highlights the importance of maintaining inter-actor connectivity and coverage in WSANs and introduces the main problem addressed in the thesis. Then the chapter presents the aim, objectives and scope of this dissertation. The chapter summarizes the research contributions and ends with the thesis organization.

1.1 Introduction of WSANs

Wireless sensor networks (WSNs) (Akyildiz et al. 2002; Yick et al. 2008) have been an immense area of interest in recent years. The work is still going on to transform the vision of a sensor-rich world (Borriello et al. 2007; Chee-Yee and Kumar 2003) into reality based on the advancements achieved in almost all aspects, including algorithms, protocols, architecture, system design, supporting tools, standards, applications, etc (Krishnamachari 2007; Xia 2008).

WSNs are widely used in a variety of applications such as in civilian, medical and military applications, etc (Akyildiz et al. 2002; Xu 2003; Garcia-Hernandez et al. 2007; Imran et al. 2008; Yick et al. 2008). Typically, they involve stringent resource constrained passive sensors that monitor sense and collect data from the environment and transmit it to a base station (Akyildiz et al. 2002). Sensor nodes in such networks are unable to interpret the data or take appropriate action in the environment. Whereas, most of the applications require autonomous and intelligent interaction with the environment such that the network is expected to respond to monitored events by performing appropriate actions. The requirement for the coexistence of sensors and actors has led to the emergence of a new class of networks capable of performing both sensing and acting on the environment, referred to as Wireless Sensor and Actor Networks (WSANs) (Akyildiz and Kasimoglu 2004). Many visionaries believe that WSANs will play a crucial role in building the network infrastructure of future cyber-physical systems (Rajkumar and Lee 2006), which promise to revolutionize the way we interact with the physical world (Xia 2008).

A wireless sensor and actor network (WSAN) is a heterogeneous network composed of numerous miniaturized sensors but fewer actors that are geographically distributed and interconnected via a wireless medium. Figure 1.1 shows a sample autonomous WSAN environment. The sensor nodes probe their surroundings, measure ambient conditions, and transmit the collected data to one or multiple actors through single-hop or multi-hop communications (Ramanathan and Rosales-Hain 2000) that require sensor-actor connectivity (Akyildiz and Kasimoglu 2004; Jie et al. 2008; Melodia et al. 2007). Actors process the received information and interact with each other in order to make decisions on the most appropriate way to perform the required action. This requires that actors should be able to communicate with each other in order to identify the most appropriate set of actors that will participate in the operation. Therefore, actors establish and maintain inter-actor or actor-actor (Akyildiz and Kasimoglu 2004; Melodia et al. 2007) connectivity in order to enable such

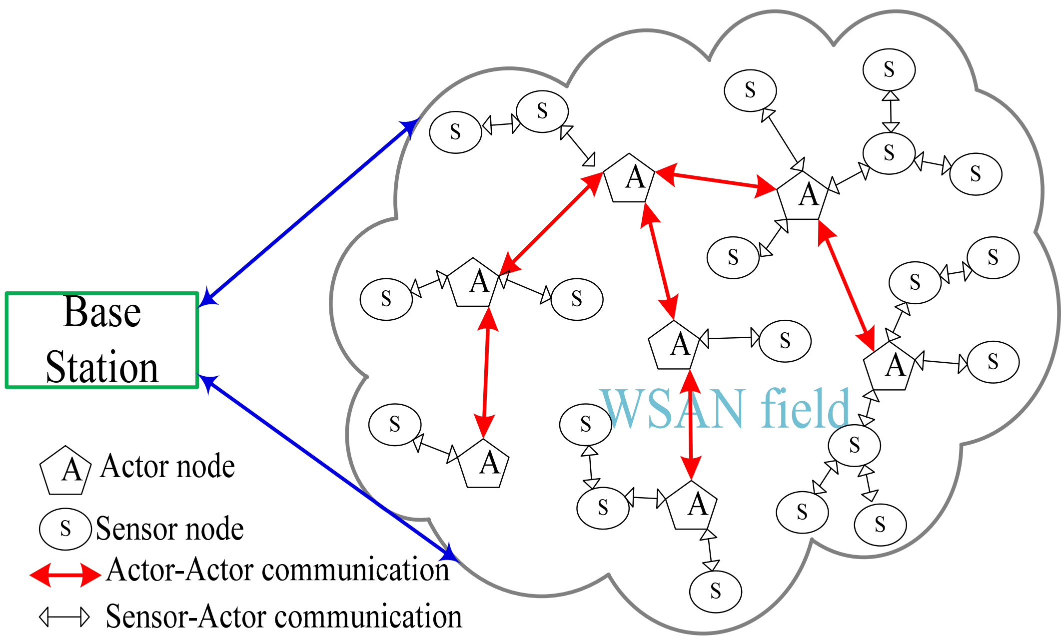


Figure 1.1: An autonomous Wireless Sensor and Actor Network setup.

communication. The base station is principally responsible for monitoring and managing the overall network through communication with sensors and actors. This allows remote monitoring and responding to the environment.

These tiny sensor nodes mainly consist of three components: sensing, computation and communication. Typically, they are equipped with limited battery power, and communication is the major consumer of energy (Akyildiz et al. 2002). Sensor nodes are low cost, low power, multi-functional devices that interact with the environment to observe some physical phenomena such as pressure, temperature, humidity, light, etc; and transmit it to actors in the vicinity. The availability of smart, cheap, lightweight and more powerful sensor nodes enables the deployment of sensor nodes in sheer size.

Most applications of WSANs use robots as actor nodes. Actors in WSANs refer to mobile heterogeneous devices such as robots, rovers, Unmanned Air Vehicles (UAVs), Autonomous Underwater Vehicles (AUVs), Unmanned Ground Vehicles (UGVs), water sprinklers, pan/tilt cameras, robotic arms, etc; which are different from actuators (Melodia et al. 2006). Many robotics research laboratories such as Sandia National Laboratories, NASA's JPL robotics, USC robotics research lab, Space and Naval Warfare Systems Centre (SPAWAR), iRobot®, etc are designing diverse actors. The detailed characteristics of more actors such as robotic mules, SKITs, mini-robots, etc. can be found in (Akyildiz and Kasimoglu 2004).

1.2 Applications of WSANs

WSANs are gaining growing interest because they not only enhance and complement the existing sensor network applications but also introduce an enormous range of new applications that require autonomous and intelligent interaction with the environment. This work is concerned with applications where actors are the essential part of the network and perform actions based on the data reported by sensors. These applications include forest fire monitoring, disaster management, search and rescue (Casper and Murphy 2003), homeland security, battlefield reconnaissance, home automation, microclimate control in buildings, oil and gas pipeline monitoring, space exploration, etc. (Akyildiz and Kasimoglu 2004; Suet-Fei 2006).

Actors are the key element to most WSAN applications, as their effectiveness entirely depends on the actor's response to detected events. Since most applications of WSANs are critical in nature, the role of an actor is extremely crucial for a timely response to events such as fire, earthquakes, disasters, hurricanes, etc. in order to prevent serious consequences. Each application determines the role of an actor (or set of actors) depending upon the requirements while considering the environment and the capabilities of the actors that may vary from one application to another. For example, an actor can extinguish a fire, lift rubble, rescue trapped survivors, deactivate a landmine or carry weapons.

In most WSAN applications, actors need to collaborate and coordinate with each other in order to plan an optimal response and synchronize their operations. For example, in the forest fire detection and containment applications, sensors are deployed to detect fires and report them to actors in the vicinity. Actors such as fire extinguishing robots and flying aircraft need to be engaged as rapidly as possible in order to control the erupted fire and prevent it from spreading. Therefore, actors should collaboratively identify the most appropriate set of actors that will participate in the operation. This necessitates that actors should be able to communicate with each other. Therefore, actors establish and maintain inter-actor topology in order to enable such communication.

1.3 Maintaining Inter-actor Connectivity and Coverage in WSANs

As discussed earlier in section 1.1, WSANs employ fewer actor nodes than sensor nodes that cooperate with each other to plan an optimal coordinated response to an event reported by sensors in their vicinity. After deployment, the actors discover each other and establish a connected inter-actor topology in order to collaborate with each other. This requires that actors remain approachable to each other at all times. Therefore, maintaining inter-actor connectivity is extremely crucial to the successful operation of WSANs.

In addition to inter-actor connectivity, actor coverage is one of the most important design goals in most applications of WSANs. It is often desirable for the actors to provide services at every part of the deployment area. Moreover, the actors' responsiveness is usually desired in order for the network to be effective (Batalin and

Sukhatme 2002b; Batalin and Sukhatme 2005; McLaughlan and Akkaya 2007; Akkaya and Janapala 2008; Akkaya and Younis 2008; Akkaya et al. 2009). For example, in forest monitoring applications, actors such as fire trucks and flying aircraft should be able to detect the presence of an erupted fire in the vicinity in a timely manner and take the appropriate action to counter the fire. This necessitates that there be at least one actor to receive event notifications, coordinate with peer actors and instigate an optimal coordinated response to the detected event. The number of actors is less, therefore, a good coverage should minimize the overlap among the action range of the deployed actors during recovery.

1.4 Partitioned WSANs

Nonetheless, failure of an actor may partition the inter-actor network into disjointed segments besides leaving a coverage hole. Consequently, an inter-actor interaction may cease and the network would become incapable of delivering a timely response to a serious event. A harsh application environment, energy depletion, malicious attack, physical damage, hardware failure, etc. can be some of the reasons for an actor failure.

This work categorizes actor failure in the context of WSANs into simultaneous multiple actor failures and a single actor failure. In simultaneous multiple actor failures, more than one actor node may fail at once. It can further be categorized based on the location of the failed actors. The failed actors might be collocated or far apart from each other. In either case, the network may be divided into disjointed chunks depending on the significance of the failed actors in the topology. Recently, some approaches have been pursued that place additional (external) relay nodes in WSNs to recover from such multiple sensor node failures and federate the disjointed segments (Lee 2010; Lee and Younis 2010a; Lee and Younis 2010b; Lee and Younis 2010c). Since, WSNs deploy sensor nodes in abundance, simultaneous failure of multiple nodes is not surprising. However, the failure of multiple actors is rare because actors in WSANs are fewer as compared to sensors in WSNs. Moreover, actors are more powerful nodes than sensors; therefore, there is less probability of multiple actors failing. However, failure of multiple actors may happen due to explosions in a battlefield, etc.

On the other hand, failure of an actor is more likely in WSANs due to some of the reasons mentioned earlier in this section. The impact of an actor failure in WSANs is much more significant as compared to sensors in WSNs. This is because sensor networks often deploy redundant nodes due to their lower cost that may also help them to tolerate some node failures. Whereas, the optimization objective of WSANs is to deploy the minimum number of actors due to their cost. The impact of an actor failure can be so significant, that the network may become dysfunctional. The failure of an actor may disrupt inter-actor connectivity that leads to a situation where actors cannot collaborate and coordinate their actions and hence cannot react to an event leaving a coverage hole. Figure 1.2 shows an articulation, where an actor failure partitions the inter-actor network into disjointed segments besides leaving a coverage hole. The shaded polygon indicates the failed actor *F* and the solid circles represent the sensors that detect the event of a fire. Inter-actor connectivity restoration is extremely crucial for the WSANs to become functional again. In this dissertation, we consider a single actor failure and a special case of simultaneous multi-actor failure which we believe is more likely to happen. In scenarios in which the failure is caused by external factors such as explosions, multiple nodes may get damaged.

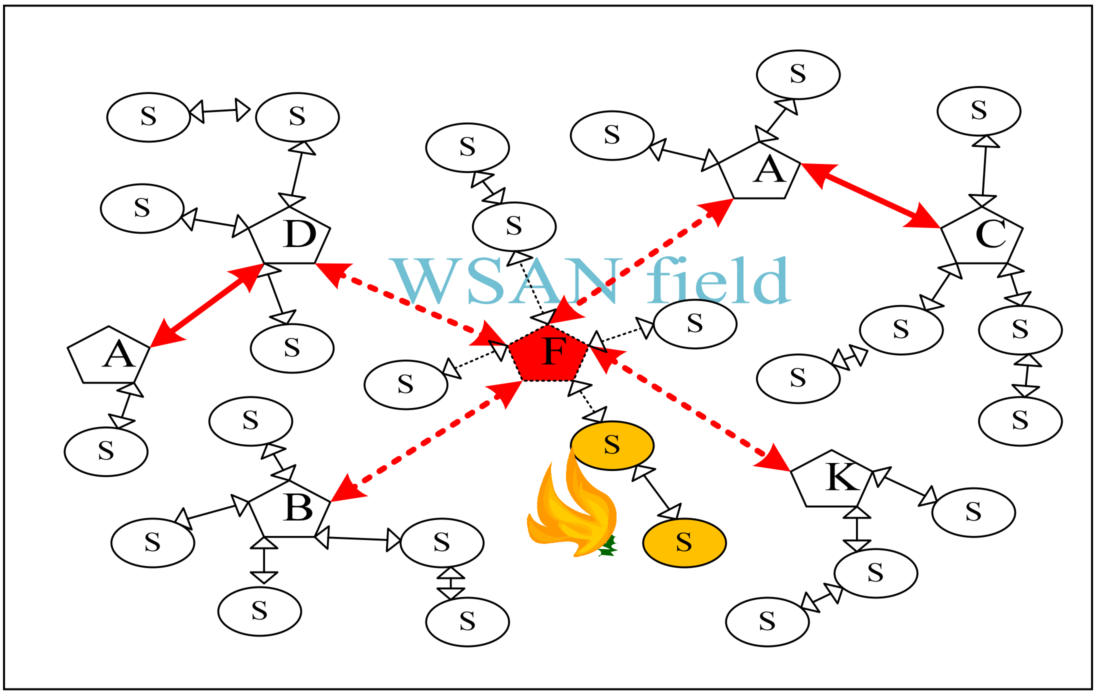


Figure 1.2: Impact of an actor failure on inter-actor connectivity and coverage.

Connectivity restoration is an emerging area of research especially in context of WSANs. Most of the existing techniques in literature, such as (Lee 2010; Lee and Younis 2010a; Lee and Younis 2010b; Lee and Younis 2010c), have been designed for static WSNs to recover from large scale network damage that involves multiple sensor nodes. Most of these schemes are centralized and rely on deploying additional (external) relay nodes to restore the connectivity. As stated earlier, simultaneous failure of multiple actors is rare due to distinguished characteristics of WSANs. Moreover, centralized approaches may not be suitable for the distributed and dynamic nature of WSANs. Furthermore, replacement of failed actors in autonomous, unattended, mission-critical and time-sensitive applications operating in a hostile environment may not be feasible.

Most of the existing movement based connectivity restoration approaches are either proactive or reactive. Proactive approaches establish and maintain bi-connected topology (Basu and Redi 2004; Das et al. 2007; Butterfield et al. 2008). The idea is to have node independent paths in order to tolerate a single node failure. Such approaches necessitate a large actor count that increases the cost and becomes impractical. In reactive approaches, neighbours of the failed node triggers a recovery process once the failure is detected (Abbasi et al. 2009b; Akkaya and Senel 2009; Tamboli and Younis 2010; Younis et al. 2010). Reactive approaches may not be suitable for mission-critical time sensitive applications due to high messaging overhead and an increased recovery delay. Hybrid approaches plan recovery ahead of time and executes after the failure. Moreover, movement based approaches do not consider application-level constraints on mobility of actors while recovering from a node failure. Unconstrained movement of actors may cause major failure at an application-level. For instance, moving an incompetent actor node (with little or unsuitable capabilities) or an actor node executing a critical task may not only disrupt the ongoing mission but may also cause unnecessary movement overhead. In this dissertation, we consider actor capabilities and the current task in execution as application-level interests while relocating the actor node.

1.5 Motivation

As discussed in the previous section, WSANs not only enhance and complement the existing WSNs applications but may also introduce an enormous range of new applications. For example, in Urban Search and Rescue (USAR) applications, sensors and actors are employed to search for disaster survivors in situations too risky and dangerous for humans due to events such as fires, earthquakes, disasters, etc. For instance, various robots were used during 9/11 to search for disaster survivors in situations too difficult or dangerous for humans (Casper and Murphy 2003). Moreover, mobile robots may also assist humans in their daily life. Our motivation is primarily driven by the fact that WSANs "will play a crucial role for building the network infrastructure of future cyber-physical systems (Xia 2008) and the world will be covered by networks of networks of smart sensors and actors (Stankovic 2008)".

However, "despite some existing research in WSANs, coordination and communication problems that arise in WSANs due to coexistence of sensors and actors are yet to be investigated" (Akyildiz and Kasimoglu 2004). Maintaining inter-actor or actor-actor connectivity is of paramount concern in order to plan an optimal coordinated response to a detected event. Failure of an actor may partition the inter-actor network into disjointed segments besides leaving an area where there is no actor to provide desired services. This may not only hinder inter-actor interaction but may also cause catastrophic events such as risking the life of some survivors. Since WSANs operate autonomously in unattended setups, the recovery should be a self-healing and agile process. Moreover, the criticality of the applications and the resource constrained nature of networks necessitate a low restoration time and reduced overhead.

The urge for inter-actor connectivity restoration and the actor relocation (Younis and Akkaya 2008) capability motivate us to explore the localized movement control algorithms for connectivity restoration. Although maintaining inter-actor connectivity is crucial, the literature review (presented in the next chapter) revealed that a considerable amount of effort is required to conduct research in movement control algorithms for connectivity restoration in WSANs and the same is presented in this dissertation.

1.6 Aim and Objectives

The ultimate aim of this research is to design connectivity restoration algorithms for wireless sensor and actor networks while minimizing the impact on actor coverage. The objectives of the research are:

1. To design localized connectivity restoration algorithms that impose a minimum recovery overhead and minimize impact of recovery on actor coverage.

2. To implement and evaluate the performance of the proposed connectivity restoration algorithms in contrast to contemporary schemes published in literature.

1.7 Scope and limitations of Research

The scope of this research is limited to maintaining inter-actor communication and coordination. Although, recovery from an actor failure helps to re-establish sensor-actor interaction, it is out of the scope of this research. This work mainly deals with single actor failure at a time and no other node fails during the execution of the recovery process until and unless otherwise stated. In other words, this work handles sequential actor failures. This work focuses on maintaining inter-actor connectivity during actor relocation and does not handle issues such as modifying the cluster membership, etc. In this dissertation, we mainly concentrate on the algorithmic aspect of the internetworking problem without considering the diversity of physical, data link and network layer issues.

1.8 Thesis Contributions

This dissertation tackles the problem of restoring inter-actor connectivity lost due to one or a special case of multiple actor failure in a WSAN. This work exploits the internal (existing) actor’s mobility to rejuvenate the actor-actor connectivity. The prime objective of this research work is to restore inter-actor connectivity while minimizing the recovery time, overhead and impact of the recovery on coverage. In addition, this work factors in application-level interests on mobility of actors while recovering from a node failure. The following highlights the major contributions of the thesis:

1.8.1 Handling single actor failure: This dissertation presents a novel reactive algorithm for connectivity restoration that exploits the partially utilized transmission range of neighboring actors and moves them towards the failed node discussed in chapter 3. The performance of the proposed algorithm is validated through extensive simulations. Simulations results confirm the effectiveness of the proposed algorithm against contemporary recovery schemes. The main advantage of the approach is simplicity and effectiveness because it does not execute sophisticated procedures to assess the impact of an actor failure on inter-actor connectivity. Moreover, we have proposed a novel hybrid approach to connectivity restoration in chapter 4. The proposed approach identifies critical (cut-vertex) actors as part of pre-failure planning and designates backup nodes for them. The pre-assigned backup nodes detect the failure and execute a recovery procedure that may involve successive relocations. We analyze the convergence of the proposed algorithm, proving its correctness and confirming its efficiency. The proposed algorithm outperforms the best published recovery schemes that address the same problem. Furthermore, a variant of the proposed algorithm is also presented and validated through simulations in the same chapter.

1.8.2 Handling a special case of multiple actor failure: This work presents a localized approach in section 4.2 that can handle a special case of multi-actor failure which we believe is more likely to happen. The proposed approach identifies critical actors (i.e. cut-vertices) in advance and designates distinct backups for them. The designated backups execute recovery concurrently once the failure of the primary actors is detected. The performance of the proposed algorithm is analyzed and validated through simulation.

1.8.3 Satisfying application-level interests during connectivity restoration: Chapter 5 presents a novel hybrid connectivity restoration algorithm that factors in application-level interests on mobility of actors while recovering from an actor failure. In order to avoid application-level failures, the proposed approach designates a backup node that best matches the actor capabilities of the primary and is executing the least critical task besides minimizing the recovery overhead and impact on coverage and connectivity. To the best of our knowledge, this is the first hybrid approach that considers application-level constraints on mobility of actors while recovering from an actor failure. The simulation validation confirms the effectiveness of the proposed algorithm in terms of meeting the connectivity restoration and/or application-level goals while minimizing the recovery overhead and impact of the recovery on coverage and connectivity.

1.8.4 Maintaining inter-actor connectivity and coverage: Repairing inter-actor connectivity strives to maintain most of the existing topology intact and preserve actor coverage. The average number of neighbors indicates the level of connectivity. On the other hand, coverage is computed as measuring the total area covered by the deployed actors. The analysis and simulation results confirm that the proposed approaches keep most of the topology intact and minimize the coverage loss compared to contemporary recovery schemes.

1.8.5 Localized Self-diagnosis and self-healing: The main advantage of this work is the ability to self-diagnose and self-heal from actor failure while only maintaining minimal network state information. All the proposed approaches in chapter 3-5 require each actor to maintain only a list of direct neighbors. This not only improves the scalability of the network but significantly reduces the communication overhead during recovery. The neighbors of the failed actor diagnosis the failure and initiates a recovery process that does not require external intervention.

1.9 Thesis Organization

Figure 1.3 shows the overall flow and organization of this dissertation. The remainder of the thesis is organized as follows:

Chapter 2 details the published node recovery schemes pursued in different contexts that exploit node mobility. We classify and categorize the existing approaches and present works which are closely related to ours.

In Chapter 3, a novel Volunteer-instigated Connectivity Restoration (VCR) algorithm for WSANs is presented. VCR is a reactive approach that opts to repair severed connectivity while imposing a minimal overhead on the neighbor actors. The performance of the VCR is validated through extensive simulations. Simulation results confirm the effectiveness of the VCR and validate the supremacy of its performance against contemporary schemes available in the literature.

Reactive approaches to connectivity restoration may not be suitable for mission-critical time-sensitive applications. Chapter 4 presents a novel hybrid Partitioning detection and Connectivity Restoration (PCR) algorithm. The performance of the PCR algorithm is analyzed and validated through extensive simulations. A variant of the PCR is the partitioning Detection and Connectivity Restoration (DCR) and is also presented in Chapter 4. The VCR, PCR and DCR can handle one failure at a time and no other node fails during recovery. Moreover, a hybrid Recovery Algorithm to handle a special case of Multiple (RAM) simultaneous failures is also presented in Chapter 4.

None of the hybrid approaches consider application-level interests on mobility of actors during connectivity restoration. Chapter 5 presents an Application-centric Connectivity Restoration (ACR) algorithm that factor in application-level constraints while recovering from an actor failure. The performance comparison of the ACR with other contemporary schemes is also presented. Finally, the chapter summarizes the proposed algorithms presented in Chapter 3, 4 and 5.

Chapter 6 concludes the dissertation, summarizes the research contributions and highlights the future work.

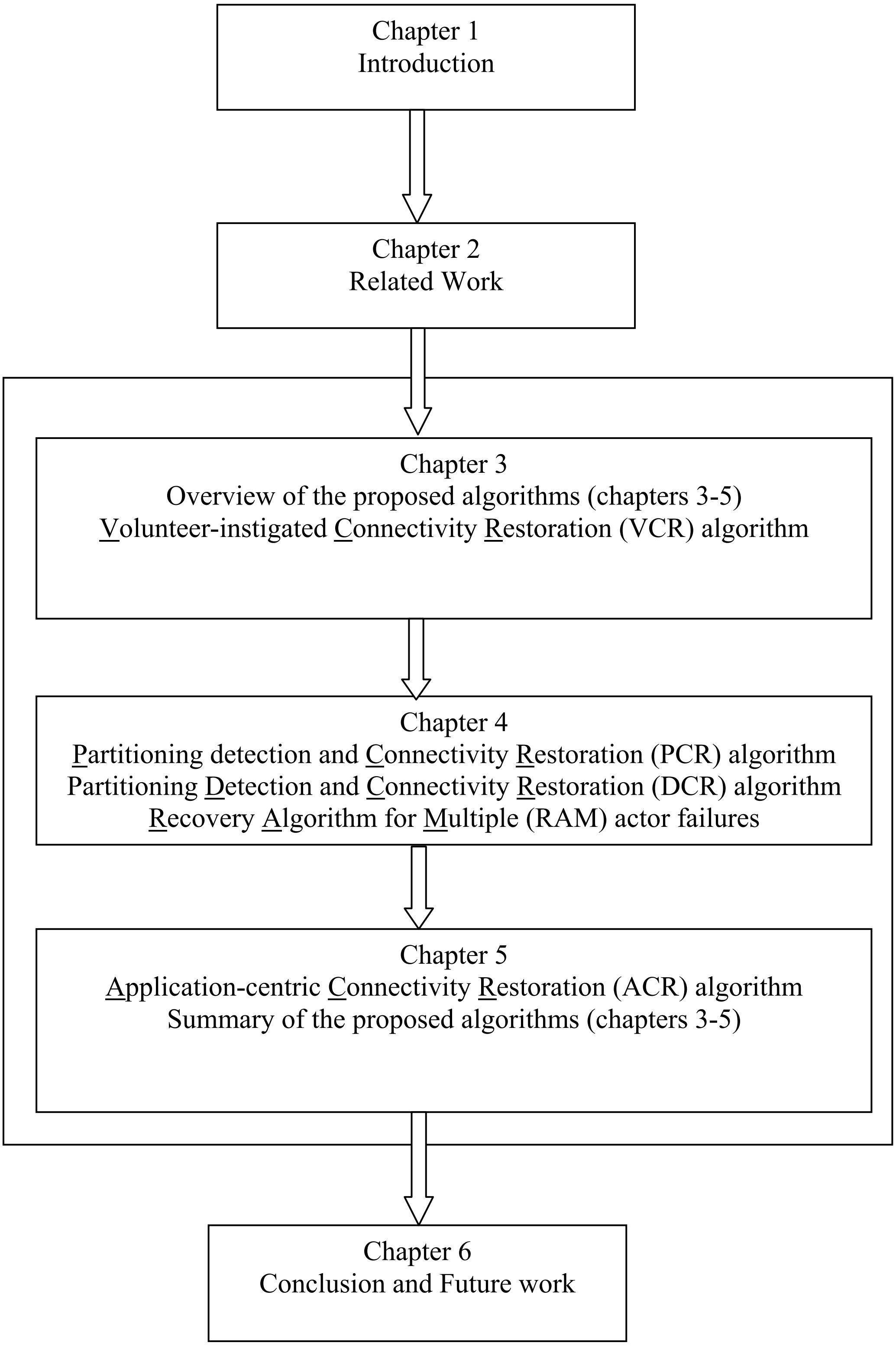


Figure 1.3: Overall flow and organization of the thesis

CHAPTER 2

RELATED WORK

In this chapter, we discuss the published node recovery schemes pursued in different contexts. We categorized the localized failure detection and recovery schemes found in the literature. Several classifications of existing node mobility approaches will be discussed. We categorized the failure recovery schemes into proactive, reactive and hybrid depending on when the recovery is triggered. Furthermore, we classified node relocation based recovery schemes depending on their primary objective. Finally, the recently proposed localized recovery schemes in different contexts will be discussed in detail.

2.1 Fault tolerance and Connectivity restoration

As discussed in Chapter 1, maintaining inter-actor connectivity and actor coverage is extremely crucial in almost all applications of WSANs, since actors have to collaborate and coordinate with each other on an optimal response, and coordinate their operations. Most of the existing algorithms do not consider impact of node failure on coverage and connectivity. The harsh application environments make actors susceptible to physical damage and component malfunction. The other possible reasons could be energy depletion, hardware failure, communication link errors, malicious attacks, physical damage, etc. Failure of a critical actor, i.e., a cut-vertex node, may partition the inter-actor network into disjointed segments besides causing loss of coverage. Consequently, an inter-actor interaction may cease and the network becomes incapable of delivering a timely response to a serious event. Therefore, fault tolerance techniques must be an essence of these networks. Since WSANs operate autonomously in unattended setups, replacing the failed actor is often infeasible and the recovery should be a self-healing and agile process that involves reconfiguring the inter-actor topology. The recovery process should introduce minimal overhead on the resource-constrained network nodes in addition to satisfying application-level constraints.

Fault tolerance (Demirbas 2004) is the ability of a system to sustain its services despite the presence of faults. Since, nodes are prone to failures, it must be inculcated into most applications of WSANs. It has been considered one of the most important topics of research and extensively investigated for WSNs. The reader is referred to (Koushanfar et al. 2004; Misra et al. 2009) for an extensive survey. The issue of fault tolerance in different contexts for WSANs has only been studied in a few publications. For instance, the fault-tolerant model presented in (Ozaki et al. 2006) designates multiple actors to each sensor and multiple sensors to each actor in order to ensure guaranteed event notification in the cases of either failure or inaccessibility. Since, the topic of fault tolerance is very broad, we limit our discussion afterwards and considers the fault-tolerant model in the context of maintaining inter-actor connectivity lost due to failure of the actor node rather than reliable sensor-actor communication.

The existing connectivity restoration techniques mainly consist of two steps i.e. failure detection and recovery. Figure 2.1 shows the classification of failure detection and recovery schemes pursued in literature. Failure detection is used to discover a node or a component failure. It can be classified into single and collaborative diagnosis depending on the number of nodes involved. Single diagnosis can further be

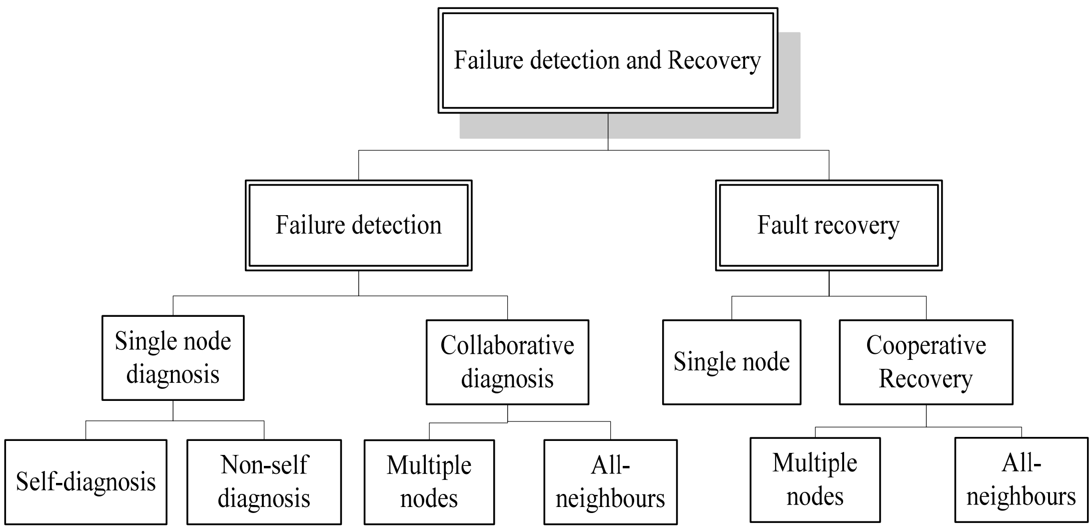


Figure 2.1: Classification of localized failure detection and recovery schemes.

categorized in to self and non-self diagnosis. In self diagnosis, a node can only determine some of its component failures. For instance, some of the faults can be determined by the actor node itself such as loss of actor action ability, remaining battery power, communication faults, etc. Non-self diagnosis involves one of the neighbor actor nodes detecting the failed actor based on the heartbeat messages that nodes exchange with each other as part of the network operation. For example, approaches like the PADRA (Akkaya et al. 2008) involve one of the neighbors detecting the failure. On the other hand, collaborative diagnosis employs more than one node to determine the failure of a particular actor. It can further be categorized based on the number of nodes involved. Whereas, the MPADRA employs two actor nodes and the RIM (Younis et al. 2008) involves all the neighbors in order to detect the failure.

Failure recovery is used to somehow compensate the network in order to achieve the expected services at the pre-failure level or continue its services with graceful degradation. Recovery schemes can be categorized in to single and cooperative depending upon the participation of actor nodes that trigger the recovery. For instance, the PADRA (Akkaya et al. 2008) only involves one of the designated neighbors triggering and executing the recovery, whereas, the RIM (Younis et al. 2008) , C^3R (Tamboli and Younis 2009), etc employ all the neighbors to trigger a recovery.

Mostly, WSANs operate in inaccessible terrain, therefore, replacing a failed node is infeasible or impractical. Moreover, since most applications of WSANs are critical in nature, they may not tolerate experiencing such failure for a long period. Another possibility is to deploy redundant actor nodes in the network. However, since, actors are quite expensive, it may not be affordable. Most of the published schemes reconfigure the topology in order to maintain connectivity of the network. Some researchers have proposed topology control algorithms to preserve fault tolerance. They used to construct a fault tolerant topology by adjusting the node transmission power. For example, two distributed heuristics were proposed in (Ramanathan and Rosales-Hain 2000) for homogeneous mobile networks to maintain connected topology using transmit power. The idea is to adjust the transmission power of nodes according to topology changes. Similarly, two localized algorithms for heterogeneous wireless networks were proposed in (Li and Hou 2004) to preserve a bi-connected network topology. Most of the schemes do not consider problems introduced due to increasing transmission power such as interference, hidden/exposed node problems, etc. Moreover, limited communication range may impose a bottleneck in maintaining connectivity.

Recent advancements pave the way for nodes to be mobile. Exploiting node mobility to maintain connectivity is recently attracting attention from the research community. Once the failure is detected, recovery is initiated by one or more concerned nodes depending on the recovery scheme. The idea is to relocate some of the nodes in such a way that the connectivity is restored. Mobility based schemes will be discussed in the following section.

2.2 Node Mobility

Exploiting node mobility as a means of data delivery optimization has been pursued by multiple researchers in the context of sensor networks and WSANs. Energy conservation (Kansal et al. 2004b), increased coverage (Liu et al. 2005) and connectivity (Akkaya and Younis 2007), minimized latency (Akkaya and Younis 2006), maximizing throughput and asset protection (Youssef et al. 2006; Youssef and Younis 2008), are the contemporary metrics targeted by node repositioning. For example, authors in (Jun and Hubaux 2005; Wang et al. 2005c; Chatzigiannakis et al. 2006) employ a mobile base station to increase the network lifetime by minimizing the energy consumed by stationary sensor nodes. Wang et al. (Wang et al. 2005b) prolong the stationary sensors lifetime by employing more capable mobile relay nodes. Similarly, Akkaya et al. (Akkaya et al. 2005) proposed gateway movement to increase sensors lifetime and throughput while minimizing latency. Almasaeid and Kamal employ mobile agents to collect data from fragmented wireless sensor networks in (Almasaeid and Kamal 2007).

Employing node mobility to mend severed topologies has just recently started to attract attention. The reader is referred to (Younis and Akkaya 2008) for a comprehensive survey of node repositioning strategies. The pursued node mobility approaches can be classified into three types based on the travel path (Pandya et al. 2008; Akkaya et al. 2010): random, predictable and controlled mobility (Kansal et al. 2004a). Figure 2.2 shows the classification of node mobility based

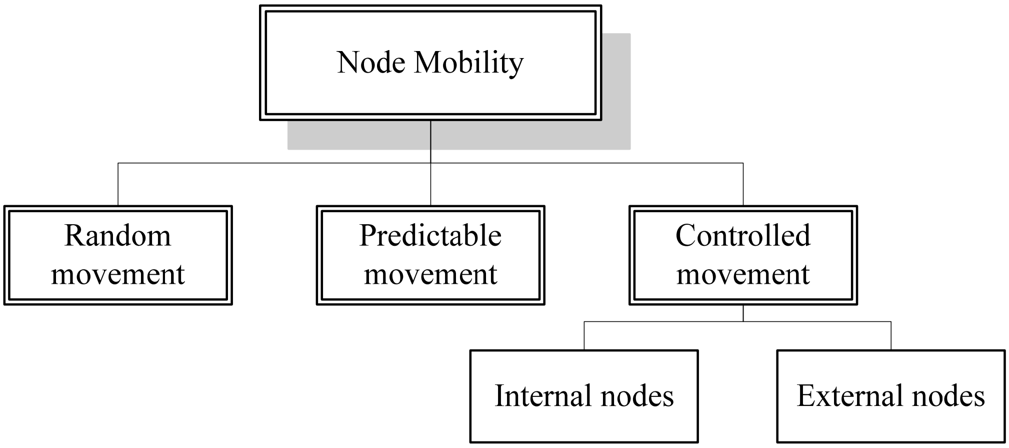


Figure 2.2: Classification of node mobility based on the travel path

approaches pursued in literature. In random mobility, nodes are assumed to move in an arbitrarily random fashion. The predictable mobility model assumes that the pattern of mobility of the mobile nodes is known and cannot be changed. In controlled mobility, nodes move on-demand and follow a predetermined path. Mobility control approaches can be categorized based on whether the internal mobile nodes (existing) are used or external nodes (Kansal et al. 2004b; Errol and Guoliang 2007) are introduced into the system. Introduction of additional nodes requires external intervention that may not be suitable for the autonomous, unattended nature of WSANs. There are several issues in moving an actor or a gateway node from one place to another. For example, randomly moving an actor may reduce coverage and loss of connectivity with other nodes. We limit our discussion afterwards to approaches that pursue controlled and coordinated relocation of existing nodes in order to preserve coverage and connectivity as will be discussed in the next section.

Moreover, Younis et al. (Younis and Akkaya 2008) have categorized the node relocation strategies intoo post-deployment and on-demand relocation based on when relocation is exploited. Post-deployment relocation is carried out at the final stage of deployment. For example, mobile sensors were moved to maximize the area coverage in (Wang et al. 2004). On the other hand, on-demand relocation is pursued to meet application-level demands while the network is operational. For instance, (Wang et al. 2005a) pursued on-demand node relocation to counter holes in coverage caused due to failure of sensors.

2.2.1 Controlled and Coordinated Multi-actor relocation

As stated in the previous section, random movement of an actor may disconnect it from other actors besides causing a significant loss of coverage. Moreover, predictable movement may not be suitable for asynchronous nature of WSANs because it is difficult to predict the location and the scope of the movement beforehand. Therefore, most of the published research has pursued movement control or controlled movement of nodes. Movement control approaches can further be categorized into coordinated and non-coordinated. In coordinated approaches, nodes inform each other about their movement so that the inter-actor topology can be adjusted accordingly. This is referred to as coordinated multi-node relocation (Younis and Akkaya 2008). On the other hand, nodes may not be able to reach each other and, as a result, become disconnected. Such approaches pursue non-coordinated relocation to re-establish connectivity among nodes.

Younis et al. (Younis and Akkaya 2008) have identified the problem of coordinated multi-node relocation as an open research issue. It becomes extremely challenging when only localized information is available to nodes as will be discussed later in the next section. Some efforts have been reported in the literature to tackle this challenging problem. For instance, an algorithm for Coordinated Relocation of gateways (CORE) has been proposed in (English et al. 2006) to maintain inter-gateway connectivity while moving multiple gateways to improve the lifetime of the network. Similarly, the COCOLA (Akkaya and Younis 2008) strives to maintain connectivity among gateways (i.e. cluster heads, CHs) while some of them move for reduced latency. However, both the above mentioned approaches do not handle gateway failures.

2.2.2 Globalized and Localized algorithms

Figure 2.3 shows the classification of existing recovery schemes that pursue movement control based approaches. They can be classified based on execution and the network state information that a node is required to maintain. Globalized schemes (Jorgic et al. 2004; Schmid and Wattenhofer 2006) require that one of the nodes or a base station should maintain the entire state of the network or be aware of the global network topology and execute the recovery scheme. This is counterproductive in terms of communication overhead because all the nodes are required to update their information every time they move. Such approaches might work for smaller networks but are not scalable for autonomous large scale networks. For example, a centralized movement control algorithm for fault-tolerant robot networks was presented in (Basu and Redi 2004). The proposed algorithm assumes that one of the robots or a base station is aware of the global state of the network. The objective was to minimize the total distance movement of all the nodes while forming a bi-connected network.

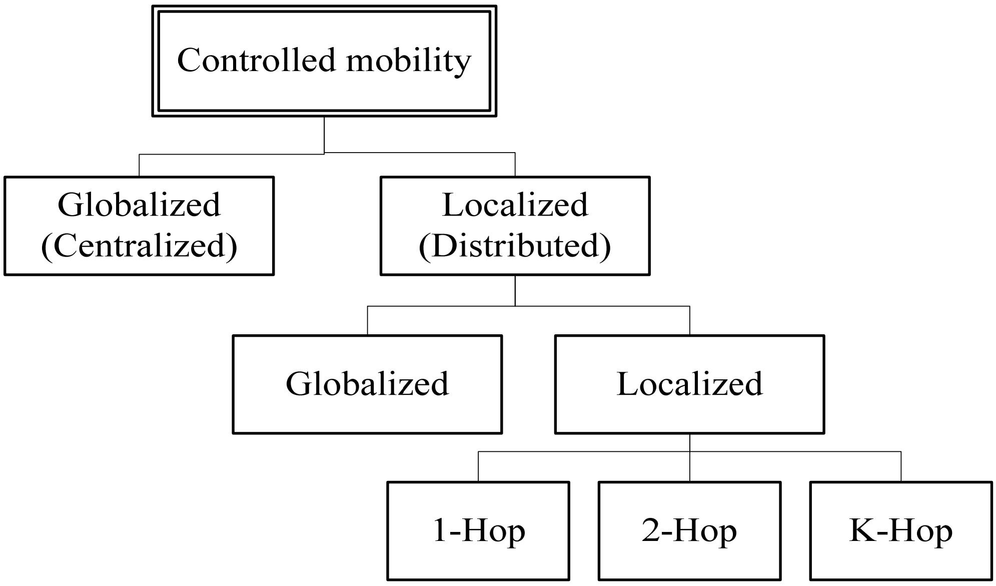


Figure 2.3: Classification of recovery schemes based on execution and the network state information a node is required to maintain.

A localized algorithm can also be implemented in a globalized distributed manner. In a globalized distributed implementation, the depth first search (DFS) can be performed in the network without global knowledge at any node, but with memorization at the nodes (Jorgic et al. 2004). Again, communication overhead due to running the DFS is significantly higher and the notion of message complexity motivates the introduction of localized algorithms (Schmid and Wattenhofer 2006).

Localized algorithms are distributed in nature and resemble greedy algorithms, where simple local behavior achieves a desired global objective. They are required to maintain only a limited local knowledge about the state of the network. The distributed and dynamic nature of sensor networks requires the design of localized algorithms to address scalability, robustness and energy efficiency issues. After deployment, nodes exchange HELLO messages with directly reachable nodes to acquire the information about their neighbors. The 1-hop neighbor list contains the information of nodes that are within the transmission range of a particular node. The detail description of maintaining and acquiring k-hop positional and topological information can be found in (Jorgic et al. 2004). For example, the DARA (Abbasi et al. 2007) and the PADRA (Akkaya et al. 2008) maintain a 2-hop neighbors list to recover from node failures. Whereas, the RIM (Younis et al. 2008) and the C^3R (Tamboli and Younis 2009) only maintain 1-hop neighbor information.

2.2.3 Proactive, Reactive and Hybrid algorithms

In order to tolerate critical node failure, three approaches are identified: (i) proactive (ii) reactive and (iii) hybrid. The proactive (pre-cautionary) approaches establish and maintain a bi-connected (i.e. each pair of nodes have two node independent paths) topology in order to provide fault tolerance. This necessitates a large actor count, and thus boosts the cost and becomes impractical. For instance, the approaches in (Basu and Redi 2004; Orozco-Barbosa et al. 2007) establish and maintain 2-connectivity even under link or node failure and the objective is to sustain such connectivity. On the other hand, in reactive (real-time) approaches, the network responds only when a failure occurs. We argue that real-time restoration better suits WSANs since they are asynchronous and reactive in nature and it is difficult to predict the location and the scope of the failure beforehand. For example, DARA (Abbasi et al. 2007), RIM (Younis et al. 2008), C^3R (Tamboli and Younis 2009), etc. trigger the recovery once the failure is detected. Such approaches are appropriate for delay-tolerant applications. However, reactive schemes might not be suitable for mission-critical time-sensitive applications that may not tolerate node failures for a long period.

In hybrid approaches, each actor proactively determines whether it is critical (cut vertex) to the network connectivity or not. Each critical actor selects and designates another appropriate actor to handle its failure when such a contingency arises in the future. The designated backup triggers a recovery process once the failure of critical actor is detected. We argue that a hybrid approach will better suit autonomous WSANs that are deployed for mission-critical time-sensitive applications due to the reduced recovery time and overhead. To the best of our knowledge, the PADRA (Akkaya et al. 2008) and MPADRA are the only hybrid approaches found in the literature. Table 2.1 reports on some of the proactive, reactive and hybrid algorithms proposed for connectivity restoration.

Table 2.1: List of a few proactive, reactive and hybrid algorithms for recovery

Proactive	**Reactive**	**Hybrid**
• Movement Control Algorithms for Realization of Fault-Tolerant Ad hoc Robot Networks (Basu and Redi 2004) • Localized Movement Control for Fault tolerance of mobile robot networks (Das et al. 2007) • Autonomous Bi-connected Networks of Mobile Robots (Butterfield et al. 2008)	• DARA • RIM • C^3R	• PADRA • MPADRA

2.3 Optimization objective

Employing node mobility to recover from node failures is an emerging area of research. Very little work has been done in this regard. Most of the studies have either considered coverage or connectivity restoration as their prime optimization objective. This section presents the noticeable efforts towards restoring coverage and connectivity while considering application-level constraints. Moreover, some works that handle multiple simultaneous failures are also mentioned.

2.3.1 Coverage and Connectivity in wireless sensor and actor networks

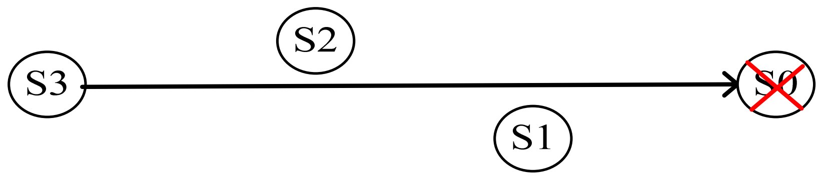
Coverage and connectivity were considered the most fundamental problems in the wireless sensor networks and have received significant attention from the research community (Wang et al. 2003; Zhang and Hou 2005; Datta et al. 2006; Ghosh and Das 2008). However, it remains largely unexplored for wireless sensor and actor networks. This section presents the efforts reported in literature.

2.3.1.1 Coverage restoration

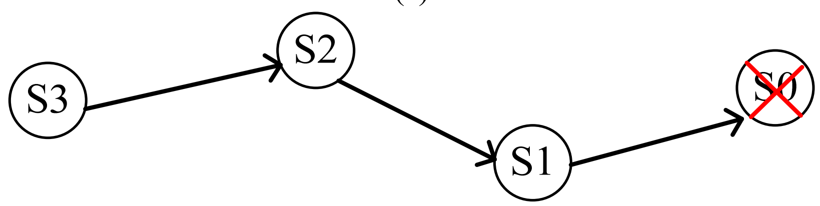
In most applications of WSANs, actor coverage is one of the most important design goals (Heo and Varshney 2003; Huang and Tseng 2003; Wang et al. 2004; Batalin and Sukhatme 2005; Akkaya and Younis 2006). Most of the existing work regarding coverage has been carried out in the context of sensor networks and mobile robot networks. Their definition of coverage is different from WSANs. For example, coverage in robot networks is defined as “the maximization of the total area covered by robots” (Batalin and Sukhatme 2002b; Batalin and Sukhatme 2003; Batalin and Sukhatme 2005). On the other hand, sensing coverage refers to the area in which sensors can detect an event of interest (Heo and Varshney 2003; Wang et al. 2004; Jie and Shuhui 2006). The existing work on coverage can be categorized into static and dynamic coverage (Batalin and Sukhatme 2005). Static coverage refers to “deploying nodes in a static configuration, in such a way that every point in the environment is covered at every instant of time” (Batalin and Sukhatme 2002b; Howard et al. 2002). Whereas, in dynamic coverage *“*nodes move around to cover the area and neither settle into a particular configuration, nor necessarily to a particular pattern of traversal” (Batalin and Sukhatme 2002a). Most of the existing efforts to improve the coverage were spent in post-deployment (Guiling et al. 2004; Wang et al. 2004; Nojeong and Varshney 2005). Very few works have been carried out which consider on-demand relocation to enhance coverage (Kansal et al. 2004c; Wang et al. 2005a).

Since our primary focus is on restoring inter-actor connectivity, we are dealing with an on-demand partial dynamic actor coverage problem.

We have considered the definition of actor coverage as: the area in which an actor can effectively/timely respond to an event reported by sensors in the vicinity (Akkaya and Younis 2006). Wang et al. exploited mobility of sensor nodes to fill the coverage holes caused by the failure of sensors in (Wang et al. 2005a). The approach is to determine nearby redundant nodes from the network and move them to the location of the failed nodes. The idea of cascaded relocation was proposed to conserve energy instead of directly moving a node over a long distance. The approach identifies intermediate nodes on the path (from redundant to the failed node) and relocates them gradually. Figure 2.4(a) depicts the direct movement of sensor S3 to the location of failed node S0. Whereas, a cascaded relocation in which all the nodes S3, S2 and S1 move concurrently is shown in Figure 2.4(b). The purpose is to minimize the total movement energy dissipated by individual sensors. However, connectivity is not considered in (Wang et al. 2005a).



(a)



(b)

Figure 2.4: Example of cascaded relocation of mobile sensors a) direct movement b) cascaded movement [redrawn from (Wang et al. 2005a)].

Other notable efforts for coverage maintenance based on sensor relocation were WCP (Guiling et al. 2004), ZONER (Xu and Santoro 2006) and MSRP (Li et al. 2007). Unlike the WCP and ZONER, the MSRP introduces a localized structure for discovering nearby replacement sensors. It strives to maintain network sensing coverage by replacing failed nodes with closely redundant sensors using a minimized time delay and balanced energy. They proposed a shifted node relocation method (a variant of the cascaded movement) based on a localized relocation path discovery that guarantees constant relocation delay and balanced energy consumption. All these approaches are only meant for mobile sensor networks and do not consider connectivity.

Recently, the coverage maintenance problem in the context of WSANs has only been considered in a few publications. A post-deployment actor placement mechanism for maximizing the actor coverage and minimizing data latency has been presented in the COLA (Akkaya and Younis 2006). However, the COLA neither deals with actor failures nor considers connectivity.

2.3.1.2 Connectivity restoration

Most of the recent research in WSANs focused on placement and coverage problems. Connectivity has been studied in the context of the deployment of actors. Meanwhile, employing node mobility to repair damaged network topologies has only recently started to attract attention. As stated earlier in Section 2.2, the proactive approaches strive to establish and/or maintain a bi-connected topology in order to tolerate node failures. The existing work on maintaining connectivity can be categorized into block (coordinated) and (independent) individual nodes movement. Block movement often requires a high pre-failure connectivity in order for the nodes to coordinate their response. An example of block movement based approaches is the work of Basu and Redi (Basu and Redi 2004), where the initial network is assumed to be 2-connected and the goal is to sustain such 2-connectivity even under link or node failure. The idea is to exploit movement of robots. However, their approach requires a centralized algorithm.

Das et al. (Das et al. 2007) presented a distributed approach to the similar problem that strives to establish 2-connectivity. The idea is to identify critical head robots based on p-hop neighbor information. Then, critical head robots direct their two neighbors to move toward each other and bi-connect their neighborhood. Butterfield et al. presented a distributed approach to establish a bi-connected robot network from the existing connected network. Unlike (Basu and Redi 2004), they do not assume accurate localization of the robots. Rather, they use a radio to determine gradients of signal strength and thereby estimate the relative bearing of the robots. Ahmadi and Stone (Ahmadi and Stone 2006) proposed an algorithm to check for bi-connectivity by listing all the doubly connected robots in a distributed fashion. They also proposed an algorithm to fix a bi-connected network of robots when robots are added to or removed from the existing bi-connected network. All of the above approaches were designed for robot networks.

The only approach that considers node failure and strives to maintain bi-connected topology in the context of WSANs is the DARA-2C (Abbasi et al. 2009b). The DARA-2C is a localized approach that maintains 2-hop neighbors information. It identifies critical actors that lost their bi-connectivity due to failure of a node. The neighbors of the failed node identify the best candidate to replace the failed node. The algorithm is recursively executed until the 2-connectivity is restored. However, our focus is on restoring 1-connectivity.

Block movements often become infeasible in the absence of a higher level of connectivity; consequently, the nodes have to react in an uncoordinated manner. Therefore, a few researchers pursue the cascaded node movement as discussed in the previous section (Wang et al. 2005a). Approaches pursuing cascaded relocations can be further categorized based on the network state that the individual nodes are assumed to maintain. Some approaches like DARA (Abbasi et al. 2007) and PADRA (Akkaya et al. 2008) base the node participation on having a list of 2-hop neighbors. Others, such as RIM (Younis et al. 2008), C^3R (Tamboli and Younis 2009), etc. avoid the increased overhead for tracking 2-hop neighbors and require each actor to be aware of their directly reachable nodes i.e. 1-hop neighbors.

To the best of our knowledge, only two hybrid approaches for connectivity restoration have been proposed in (Akkaya et al. 2008; Zamanifar et al. 2009) that handles a single actor failure. The PADRA (Akkaya et al. 2008) identifies a connected dominating set (CDS) of the whole network in order to detect cut-vertices. Since the CDS based method is not accurate for critical node detection, a depth-first search (DFS) is performed on each member of the CDS to confirm if the node is really a cut vertex or not. The idea is to identify cut-vertex nodes in advance and designate an appropriate neighbor to handle their failure. The designated node initiates a recovery and picks a dominate (i.e. non cut-vertex) to replace the failed node by pursuing the cascaded movement. Although, they use a distributed algorithm their solution still requires the 2-hop neighbors' information which increases messaging overhead. Another similar approach to the PADRA is presented in (Zamanifar et al. 2009) and uses a different algorithm to identify cut-vertices. They used the combination of cascaded and block movement relocation strategy. Although, the algorithm is localized, it still requires 2-hop information to detect cut-vertices. Both the algorithms are based on the CDS.

Recently, some localized movement control connectivity restorations have been proposed in DARA, RIM and C^3R. The DARA is a reactive approach that pursues coordinated multi-actor relocation in a cascaded manner to recover from cut-vertex node failure. The neighbors of cut-vertex node *F* detect the failure (through cooperative diagnosis) and initiate a recovery by looking for the best candidate among them based on the least node degree and distance. The DARA maintains 2-hop neighbor information; therefore, all the 1-hop neighbors of *F* remain connected and coordinate the recovery. Once the best candidate is decided, it moves to the location of *F*. The DARA is recursively applied to recover from a connectivity loss due to the movement of the best candidate. The DARA is a connectivity restoration approach and does not consider coverage. The DARA strives to restore connectivity lost due to failure of a cut-vertex. However, it requires more network state information in order to ensure convergence.

The RIM is a non-coordinated recovery approach that moves neighbors of *F* inwards until they become connected. Usually the repositioning of the neighbors of *F* causes more links to break and the relocation process repeats in a cascaded manner. The RIM strives to minimize the movement overhead on individual nodes. However, the scope of recovery (number of nodes involved in recovery) significantly increases. Although, the RIM and C^3R use 1-hop neighbor information to restore connectivity but they are purely reactive and do not differentiate between critical and non-critical nodes.

2.3.1.3 Coverage-aware Connectivity restoration

As discussed in previous sections, most of the schemes in literature either consider coverage or connectivity. However, some approaches cared for both connectivity and coverage. For example, a post-deployment actor repositioning algorithm for coverage improvement without losing connectivity is proposed in (Akkaya and Younis 2007). The idea of the C^2AP is to apply a repulsion force on neighbor actors in a distributed manner to maximize the actor coverage while maintaining inter-actor connectivity. Moreover, the C^2AP handles orphan actors and makes them connected with the other network. However, the C^2AP does not handle actor failures. Akkaya and Younis proposed a post-deployment actor placement mechanism the COCOLA (Akkaya and Younis 2008) that cares for coverage and connectivity besides latency. The COCOLA is an extension of the COLA (Akkaya and Younis 2006) that enforces connectivity among actors. It does not consider actor failures.

A distributed algorithm to establish a connected inter-actor network from disjointed inter-actor sub-networks is proposed in (Akkaya and Senel 2009; Senel et al. 2007). The idea is to pursue the coordinated movement of actors in order to establish connectivity among the sub-networks without violating the intra-connectivity of sub-networks. Akkaya et al. proposed an actor placement mechanism for maximized coverage and guaranteed inter-actor connectivity in (Akkaya et al. 2009). The definition of actor coverage is different in (Akkaya et al. 2009) and is based on the sensor density in the action range of the actors. Akkaya and Janapala (Akkaya and Janapala 2008) address inter-actor connectivity and coverage at the network setup time. Actors apply repelling forces to spread out and switch to an attraction force when the actors become disconnected. However, none of the approaches proposed in (Senel et al. 2007; Akkaya and Janapala 2008; Akkaya and Senel 2009; Akkaya et al. 2009) handle actor failures.

Tamboli and Younis (Tamboli and Younis 2009; Younis et al. 2010) pursued mobile sensor relocation to cope with the loss of coverage and connectivity when a node fails. Instead of reconfiguring the network topology, nodes move back and forth to replace the failed node in order to provide intermittent rather than permanent recovery. Obviously this solution leads to frequent topology changes, imposes lots of overhead and would thus become suitable only as a temporary solution until spare nodes are deployed.

2.3.2 Handling multiple simultaneous failures

As discussed in Chapter 1, there are some approaches that have been designed in the context of static WSNs such as (Lee 2010; Lee and Younis 2010a; Lee and Younis 2010b; Lee and Younis 2010c) to recover from large scale network damage that involves multiple sensor nodes. Most of these schemes are centralized and rely on deploying additional (external) relay nodes to restore the connectivity. However, centralized approaches may not be suitable for autonomous, unattended and large scale WSANs deployment. Moreover, most of the localized published schemes discussed so far only handle single node failure one at a time with no other nodes failing during recovery. The very first work to handle multiple simultaneous node failures in the context of sensor networks has been recently proposed in (Lee and Younis 2010d). They introduce relay nodes (RNs) to restore overall connectivity as soonest possible and then minimize the number of RNs. Akkaya et al. extended their work (Akkaya et al. 2008) by introducing a mutual exclusion mechanism in the MPADRA (Akkaya et al. 2010) to handle multiple simultaneous failures. The MPADRA reserves the nodes on the path in advance before actual relocation even when the failed nodes are far apart. The MPADRA maintains 2-hop network state information and requires primary and secondary failure handlers for each dominator.

2.3.3 Application-centric connectivity restoration

As stated in previous sections, most of the existing schemes have considered coverage and connectivity (Wang et al. 2005a; Abbasi et al. 2007; Li et al. 2007; Younis et al. 2008; Tamboli and Younis 2009) as the primary objectives. They do not consider application-level constraints on the mobility of actors. To the best of our knowledge, the only work that factors in application-level constraints on mobility of actors was recently proposed in the C^2AM (Abbasi et al. 2009a). The idea is to assign a mobility readiness index (MRI) to each actor based on the importance of the job currently being executed. The value of the MRI determines whether an actor is allowed to move or not. First, the C^2AM is purely a reactive approach that may not be suitable for mission-critical time-sensitive applications. Second, the C^2AM does not consider actor action ability while moving an actor in order to recover from a failure. Moving an incompetent actor may have a counterproductive effect on the application. Third, the C^2AM requires maintenance of 2-hop information and does not care for actor coverage.

Table 2.2 presents recently proposed algorithms that exploit node mobility to address the problem of coverage, connectivity and/or application-level constraints.

.

Table 2.2: Summary of the recent closely related works

Reference	Optimization objective	Limitations
(Abbasi et al. 2007)	Connectivity restoration	Reactive approach Maintain 2-hop information Neither consider coverage, nor application-level constraints
(Younis et al. 2008; Younis et al. 2010)	Connectivity restoration	Reactive approach Neither consider coverage, nor application-level constraints Overreact even in case of non-critical node failure Scope of recovery is high
(Tamboli and Younis 2009; Tamboli and Younis 2010)	Coverage and Connectivity	Reactive approach Do not consider application-level constraints Overreact even in case of non-critical node failure Temporary solution
(Abbasi et al. 2009a)	Connectivity and application-level constraints	Reactive approach Do not consider actor capabilities and coverage Maintain 2-hop information
(Akkaya et al. 2008)	Connectivity	CDS based approach and require 2-hop neighbor information Neither consider coverage, nor application-level interests
(Akkaya et al. 2010)	Connectivity and Coverage Handle multiple actor failure	CDS based approach and require 2-hop neighbor information Designate primary and secondary failure handler to each critical actor

Reference	Optimization objective	Limitations
(Guiling et al. 2004; Wang et al. 2005a; Xu and Santoro 2006)	Coverage maintenance	Rely on global/cross-network messaging Designed for Mobile sensor networks
(Li et al. 2007)	Coverage maintenance	Do not consider connectivity Designed for Mobile sensor networks
(Akkaya and Younis 2006; Akkaya and Younis 2008)	Coverage and latency	Post-deployment relocation Do not handle actor failure
(Akkaya and Younis 2007; Akkaya and Janapala 2008)	Coverage-aware connectivity	Post-deployment relocation Do not handle actor failure
(Senel et al. 2007; Akkaya and Senel 2009)	Establish connectivity among disjoint sub-networks	Do not handle actor failure
(Zamanifar et al. 2009)	Connectivity restoration	CDS based approach require 2-hop neighbor information Neither consider coverage, nor application-level constraints

2.4 Chapter Summary

In this chapter, we have presented the node failure recovery schemes published in the literature. Published failure detection and recovery techniques were categorized. We have presented several classifications of node mobility schemes pursued in literature. We have categorized the recovery schemes into proactive, reactive and hybrid based on when the recovery is initiated. We classify the published node relocation based recovery schemes based on their optimization objective. The contemporary recovery schemes pursued in different contexts were discussed.

The next chapter presents the proposed Volunteer-instigated Connectivity Restoration (VCR) algorithm for wireless sensor and actor networks.

CHAPTER 3

VOLUNTEER-INSTIGATED CONNECTIVITY RESTORATION ALGORITHM

As discussed in Section 2.2.3, reactive approaches to connectivity restoration better suits some applications of WSANs. In this chapter, we present a novel distributed Volunteer-instigated Connectivity Restoration (VCR) algorithm that engages neighbors of the failed actor *F* based on their proximity and how partially they currently utilize transmission range. These neighbors volunteer by increasing their transmission power and moving towards *F*. In order to avoid increased collision in the vicinity of *F*, VCR applies a diffusion force among volunteer actors based on their transmission range so that they spread while staying connected. Simulation results confirm the effectiveness of VCR and validate the superiority of its performance compared to published schemes.

The rest of the chapter is organized as follows. Section 3.1 introduces the system model and assumptions considered for the proposed approaches presented in chapter 3, 4 and 5. The design of the proposed VCR algorithm is provided in Section 3.2. The validation results are presented in Section 3.3. Finally, the chapter is concluded in Section 3.4.

3.1 System model and assumptions

This work is applicable to WSANs that involve sensors and actors. Both sensors and actors are deployed randomly in an area of interest because in most applications of WSANs, controlled deployment (Younis and Akkaya 2008) is risky and infeasible. Moreover, random placement (Younis and Akkaya 2008) becomes the inherent choice for such applications due to its ease of deployment and suitability for fault tolerance (Ishizuka and Aida 2004). Upon deployment, sensors are assumed to form a connected sensor-actor network and thus no partitions exist among the sensors. Two sensor nodes are connected if they are within the communication range of each other.

Since, we assume a symmetric communication range, the connectivity is bi-directional. Similarly, actors ascertain each other and establish an inter-actor or actor-actor network using some of the existing techniques such as (Akkaya and Younis 2006; Bao and Garcia-Luna-Aceves 2003). Maintaining inter-actor connectivity is the prime objective of this dissertation. However, the inter-actor network can be partitioned into disjointed segments due to failure of an actor. All the communication is over a single shared wireless channel (Akkaya and Senel 2009) .

Figure 3.1 depicts the considered autonomous WSAN model where sensors detect ambient environment conditions and report an event of interest to one or multiple actors. Actors receive reports from sensors, and process and collaborate with each other to plan an optimal coordinated response. An event is assumed to be detected if it appears within the sensing range (r_s) of some sensor. Sensing range refers to the area where sensors can sense and is assumed to be symmetric for all sensors. Typically, sensors are assumed to be stationary and can send their data to actors either directly or over multi-hop routes. On the other hand, actors are assumed to be able to move on demand in order to enhance coverage and inter-actor connectivity, and such relocation does not affect sensor-actor connectivity. However, an actor notifies its neighbors before moving so that the topology of the actor-actor network can be adjusted accordingly, and to avoid an actor being perceived as faulty (Bao and Garcia-Luna-

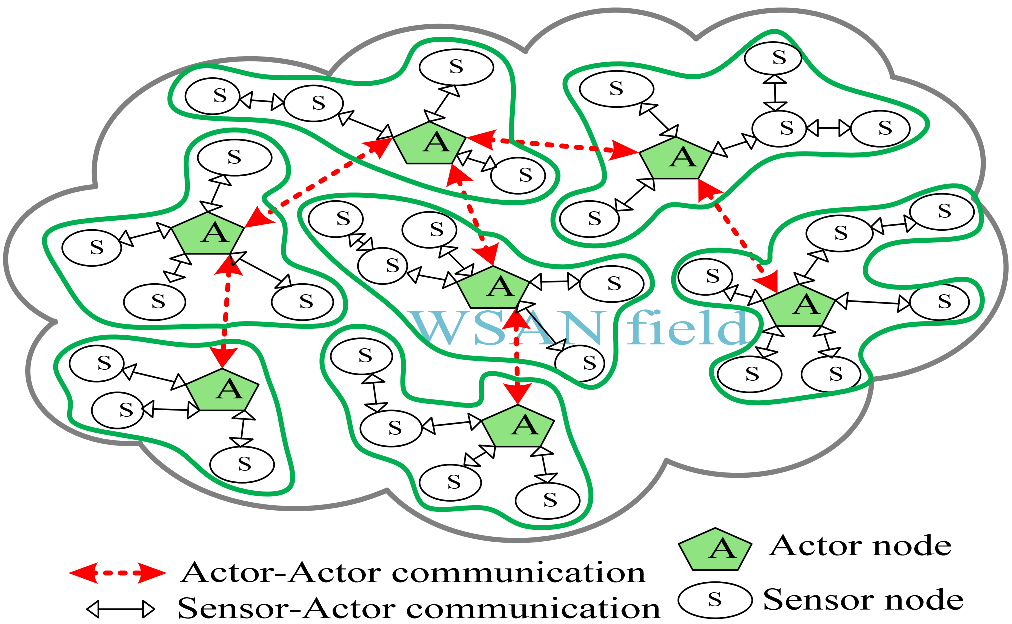


Figure 3.1: An autonomous WSAN system model

Aceves 2003). Moreover, actors are assumed to smoothly move to their destined locations without any obstacles. However, such movements are assumed to be more costly than message transmission (Wang et al. 2005a).

Sensors are deployed in abundance while actors are significantly fewer than sensors. Sensors are inexpensive but have scarce resources compared to actors in terms of energy, communication and computation (processing and memory). Due to the higher cost, availability of spare actors is infeasible. However, additional nodes may be deployed to increase the coverage level. Such inclusions do not affect the recovery process because we do not factor in the presence or absence of spare nodes (Tamboli and Younis 2010). The communication range (r_c) of an actor refers to the maximum Euclidean distance that its radio can reach and is assumed to be larger than that of sensors. This can be achieved through employing two radios on actor nodes i.e. one for sensor-actor and the other for actor-actor communication (Akkaya and Senel 2009). We assume that all actors can dynamically adjust the output power of their radio when transmitting. It is worth noting that commonly used radio hardware such as CC1000 and CC2420 allows adjusting the transmission power levels at run time (Lin et al. 2006). The maximum communication range of an actor is r_{max}. To simplify analysis, nodes are assumed to have the same communication range unless stated otherwise. We assume a free space propagation model for analysis (Younis et al. 2010).

The action range of an actor refers to the maximum area in which an actor can cover (Batalin and Sukhatme 2005) and is assumed to be equal for all actors. It is used to measure the coverage of an actor. We assume that all the nodes (sensors and actors) can determine their location through GPS or other localization techniques such as (Bulusu et al. 2000; Youssef et al. 2005). Each actor is assumed to maintain a list of directly reachable (1-hop) neighbors. This list is populated by sending HELLO messages right after deployment. The neighbor actors periodically exchange heartbeat messages to update the status of each other. An actor is assumed to have failed if its heartbeat messages are not received successively. The neighbor (s) of the failed actor detects the failure and initiates a recovery.

Although we consider a WSAN system model, yet it is worth noting that our algorithms are equally applicable for Mobile Sensor Networks (MSNs) and Mobile Robot Networks (MRNs). Mobile Sensor Networks (MSNs) (Younis et al.

2010) employ a batch of mobile sensors that form a flat network topology or a two-tier hierarchical network architecture in which stationary nodes are at the first tier. In hierarchical architecture, the network is a combination of static and mobile nodes, wherein, the later establish and maintain a connected inter-node network. Mobile nodes such as Micabot (McMickell et al. 2003), Robomote (Dantu et al. 2005), etc. are used to serve as gateways or actors and relay data to the base station. The model is adopted and depicted in (Younis et al. 2010).

3.2 Design of Volunteer-instigated Connectivity Restoration algorithm

This section presents the application scenario, problem definition, design, pseudo code and performance evaluation of the proposed VCR algorithm.

3.2.1 Application scenario and problem definition

As mentioned earlier in Chapter 1, most applications of WSANs require actors to establish and maintain a connected inter-actor topology in order to coordinate with each other on an optimal response and synchronize their operations. Nonetheless, the harsh environment that WSAN operates in makes actors susceptible to physical damage and component malfunction. An actor failure may partition the inter-actor network into disjointed segments and consequently hinder the inter-actor interaction. Since WSANs operate autonomously in unattended setups, replacing the failed actor is often infeasible and the recovery should be a self-healing and agile process that involves reconfiguring the inter-actor topology. In addition, restoring connectivity should introduce a minimal overhead on the resource-constrained network nodes.

The effect of an actor's failure depends on the position of that actor in the network topology. For example the loss of a leaf node, such as *K* in Figure 3.2, has no negative impact on the inter-actor reachability. Meanwhile, the failure of a cut-vertex such as *F* partitions the network into disjointed segments. Generally, two methodologies can be identified in order to tolerate the failure of a cut-vertex node: (i) pre-cautionary and (ii) real-time restoration. The pre-cautionary methodology provisions fault-tolerance by forming and maintaining a bi-connected topology. However, provisioning such a level of connectivity requires a large actor count, and thus boosts the cost and

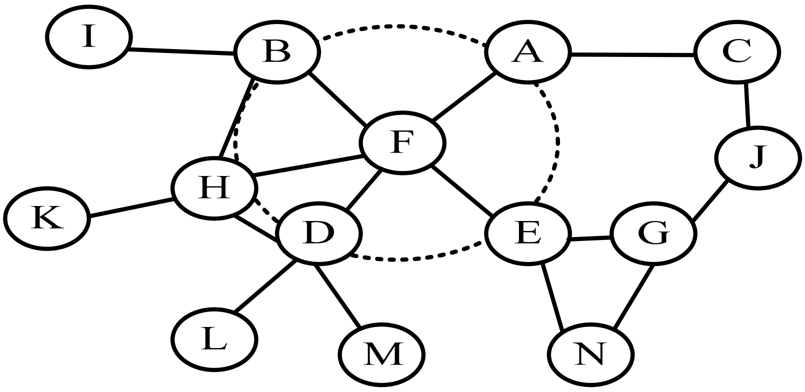


Figure 3.2: An example of a WSAN with connected inter-actor network

becomes impractical. On the other hand, with real-time restoration the network responds only when a failure occurs. We argue that real-time restoration better suits WSANs since they are asynchronous and reactive in nature, and it is difficult to predict the location and the scope of the failure beforehand. The proposed VCR algorithm assumes single actor failure at a time and no other node fails during the recovery.

3.2.2 VCR overview

In this section, we present a novel Volunteer-instigated Connectivity Restoration (VCR) algorithm. VCR is based on an instinctive social behaviour that can be observed frequently in most of creatures. For example, in the case of a person's death, the most closely concerned peoples among many acquaintances examine their availability based on their commitments and voluntarily take up the responsibilities accordingly in addition to their own. Similarly, in VCR the failure of *F* is detected by immediate neighbor actors, because they are directly affected, which are referred to as bereaved actors (BAs). These bereaved actors (BAs) examine their proximity to F and partially utilized transmission range in order to decide whether to participate in the recovery process or not. The ardent bereaved actors that voluntarily proffer to take part would hereafter be referred as volunteer actors (VAs). The VAs jointly take up the additional responsibility in order to restore the lost connectivity. Bereaved actors that could not help out in the recovery due to a lack of resources, unfavourable environmental conditions, etc., are called horrid actors (HA). VCR is described in detail in the following section.

3.2.3 Detailed VCR algorithm

In VCR, each actor maintains a list of 1-hop neighbors and monitors their heartbeats. The failure of actor F is detected by its neighbors via a cooperative diagnosis (Misra et al. 2009) through missing heartbeats. The VCR algorithm avoids performing a network-wide analysis to evaluate the impact of the actor failure on connectivity. Rather, it executes a recovery process without concern for whether the failed node F is critical or not. VCR employs simple recovery procedures to restore connectivity lost due to failure of a node. The recovery process consists of two phases. First, volunteer actors are identified. In the second phase the topology repair is performed through uncoordinated relocation of the volunteer actors while exploiting the partially utilized transmission range and actor diffusion. The following explains these two phases.

a. Volunteer declaration: Upon detecting the failure of *F*, bereaved actors, i.e., neighbors of *F*, decide on whether to participate in the recovery (volunteer) or not (horrid) based on the following criteria:

i. Proximity: An actor $A \in$ Neighbors (*F*) calculates its distance *d (A, F)* to *F*. If *d (A,F)* is more than $\alpha . r_{max}$, actor "*A*" is not required to participate in the recovery at this time, i.e., close neighbors to *F* are favored as volunteers. Assume uniform node placement of *N* nodes in a square area ($L \times L$). The distance between two nodes in the same row is $L/\sqrt{N}$ and the distance between two diagonally neighboring nodes is $L.\sqrt{2/N}$. Therefore, the initial value of α is set as the average proximity to neighbors:

$$\alpha = .5\,(\frac{L}{\sqrt{N}} + L.\sqrt{\frac{2}{N}} \qquad (3.1)$$

It is worth noting that α is increased if actor *A* is not connected within a preset time in order to increase the threshold for not participating. In other words, a bereaved can switch from the horrid to volunteer state depending on observed progress on the status of the connectivity restoration.

ii. Legibility factor: The transmission power of nodes significantly affects the network connectivity. While the power level at the transmitter determines the reachable range, i.e. how far the receiver can be, high power may increase interference and boost the count of exposed nodes (Correia et al. 2007). Therefore, power control is usually pursued in order to balance the interest in high connectivity and efficient utilization of the wireless channel. Particularly, nodes carefully set their transmission power to achieve signal-to-noise ratio (SNR) that suits the intended receiver and limits the potential of medium access collision with other nodes in the vicinity. In addition, power control is further employed in order to conserve energy. VCR exploits the fact that many actors are not utilizing their full range and would be able to boost their transmission power to reach other receivers further than their neighbors. Thus, nodes whose range is only partially utilized should be favored in the recovery process. The legibility factor (*LF*) of an actor captures the effect of the ratio of current range r_c to maximum range r_{max}, i.e.,

$$LF = 1 - (\frac{r_c}{r_{\max}}) \tag{3.2}$$

Actors with a high legibility factor are favored. A bereaved actor becomes a volunteer if its *LF* exceeds a preset threshold β. Initially, β can be approximated based on the actor density. Assume the size of the deployment region is "Area". For a uniform actor deployment, the value of r_c for establishing a connected network should be set such that:

$$Area = N.\pi\frac{1}{4}r_c^2 \tag{3.3}$$

whereas N is the number of actors. Using the equation 3.3,

$$r_c = \sqrt{\frac{4.Area}{N.\pi}} \tag{3.4}$$

can calculate an initial value of β. The initial β value is gradually decreased if no recovery is achieved in a certain time in order to increase the number of volunteers and restore connectivity.

b. <u>Topology Repair:</u> Volunteer actors carry out the recovery procedure by exploiting their partially unutilized transmission range and moving towards *F* as needed. Topology repair involves the following steps:

i. *Volunteer relocation*: The fact that an actor is using a fraction of its maximum communication range r_{max} indicates that this actor can move away from its current spot and make up for the increased proximity to its neighbors by boosting the output power of its radio. A volunteer actor "*V*" would exploit this capability by moving to a distance $\gamma.r_{max}$ from *F*. Prior to departing its current position actor "*V*" will notify its children. While moving, actor "*V*" will increase its transmission power to stay connected to its children. If *d (V, F)* exceeds ($r_{max} - r_c$), the children that are not bereaved actors will follow to stay connected to *V*, a step that is referred to in the literature as cascaded relocation (Abbasi et al. 2007). VCR opts to avoid or at least limit the scope of the cascaded relocation by favoring bereaved actors that are close to *F* and can increase their transmission power. At the start of the recovery process γ is set to 0.5. The rationale is that if all neighbors of *F* are a distance of $\frac{1}{2}.r_{max}$ away from *F,* the network becomes connected again (Younis et al. 2008). However, the value of γ will be further reduced if the volunteer actors sensed a high dose of interference in the vicinity of *F*, which in essence requires *V* and other volunteers to get closer to *F* in order to reach one another.

ii. *Connecting horrid actors*: Horrid nodes will wait for volunteers to re-establish connectivity. The rationale is that volunteers will end up in the vicinity of *F*, yet not at the position of *F*. Therefore, there is a high probability for horrid actors to be able to reach one of those volunteers without a need to incur overhead. If a preset time passes without hearing from a volunteer, a horrid actor increases the value of α and/or lowers β to become a volunteer. However, in this case, horrid actors will try to increase their transmission range first in order to find out whether other volunteers can be reached, before pursuing repositioning. When a horrid actor becomes connected to a volunteer, it declares the success of the recovery based on the following theorem:

<u>Theorem 3.1</u>: *The network becomes strongly connected if it was strongly connected before node F fails and if every horrid actor can reach a volunteer actor*.

<u>Proof</u>: If the network was strongly connected before *F* fails, every actor should have a path to every other actor in the network. The failure of *F* will affect the connectivity of the neighbors of *F*. Establishing links between those neighbors will make the network strongly connected again. When volunteer nodes are within a distance of ½ r_{max} from *F*, they become connected. Thus, if every horrid actor can reach a volunteer, all neighbors of F will be connected again. □

iii. *Spreading out volunteers*: Although, increasing the transmission power of *V* enables it to reach other neighbors of *F* and also its children while and after moving, actor *V* may negatively affect other nodes in the vicinity. In particular, if the network gets partitioned increasing the transmission power of volunteers from the same segment after they move close to *F, it* will boost medium access contention and radio signal interference. Therefore, upon reconnecting with one another and also with horrid actors, volunteers will apply a diffusion force based on the proximity to their neighbors. The cumulative effect of spreading the actors is like stretching the topology of the network that enables discovery of new connections. The diffusion force applied from actor *A* on actor *B* is defined as follows:

$$F_{A\to B} = \begin{cases} \frac{1}{2}(r_{max} - d_{AB}) & if\ r_{max} > d_{AB} \\ 0 & if\ r_{max} \leq d_{AB} \end{cases}, \tag{3.5}$$

where d_{AB} is the distance between *A* and *B*. The force is proportional to the difference between the maximum range and current distance. The division by 2 is because there is an equivalent force from *B* on *A*.

3.2.4 Pseudo code and illustrative example

Figure 3.3 shows a high level state diagram of the VCR algorithm. Upon detecting the failure of neighbor *F*, the actor switches from a normal to a bereaved state. The transition from the bereaved state would depend on the role that the actor decided for recovery. A neighbor with a close proximity to *F* and a high legibility factor (*LF*) transitions to the volunteer state. Otherwise, the actor transitions to the horrid state

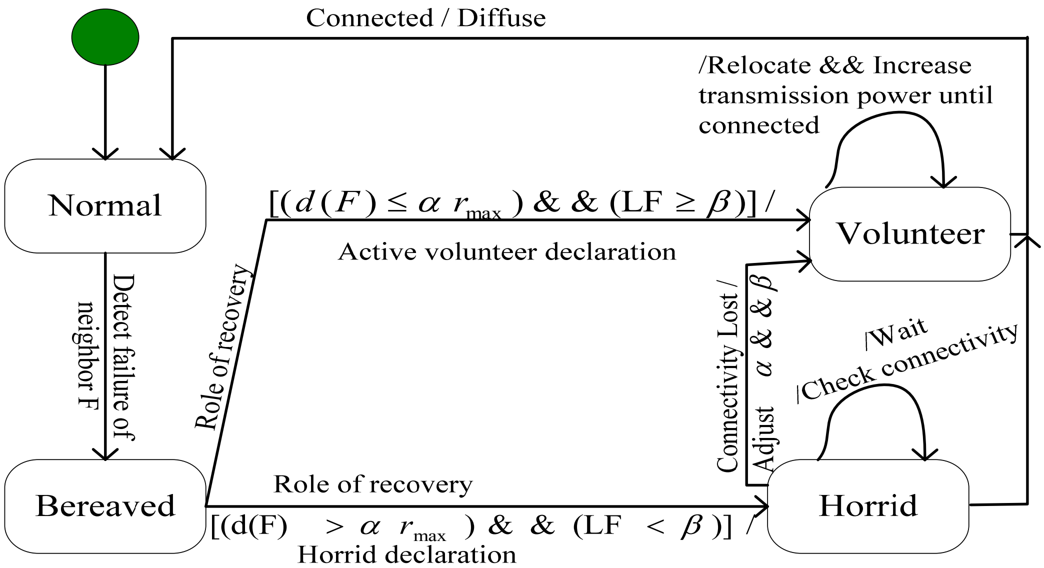


Figure 3.3: High level state diagram description of the VCR algorithm.

and watches progress in the connectivity restoration. In the volunteer state, the actor performs relocation along with increasing transmission power until either it becomes connected with other volunteers or becomes a distance of $\gamma.\ r_{max}$ away from *F*. When the connectivity is restored, a volunteer actor applies diffusion forces to spread away from the neighboring actors without breaking the communication links but reducing the potential of contention and interference. At the conclusion of the recovery, the actor switches back to the normal state. In the horrid state, actors wait for volunteers to re-establish connectivity. They continuously monitor the situation for a preset time. If the links are not established, the actor times-out and transitions to the volunteer state by increasing the value of α and/or decreasing β. Again, in the volunteer state, it performs the same actions as described above.

Figure 3.4 shows the pseudo-code of VCR. The algorithm is to run on all actors in a distributed manner. When actor *A* detects failure of neighbor *F,* it sets a flag to indicate that recovery is incomplete (lines 1-2). This mimics the "bereaved" state in Figure 3.3. The "while" loop (lines 3-22) reflects the actions in the volunteer and horrid states. The loop repeats until the recovery is complete. Lines 5-12 are the steps taken in the volunteer state where children are notified and the actor moves towards *F* while increasing the transmission power. Otherwise, a horrid state is declared in line 13 and the actor monitors its connectivity to the volunteer actors. After waiting for τ time units, a horrid actor times out and adjusts α and β in order to switch to the volunteer state. The loop does not terminate until the actor sets the "Recovered" indicator to true. Upon restoring connectivity, the actor employs diffusion forces to keep a suitable distance from its neighbor (line 23). The actor may not be a neighbor of *F* but is rather a child to one of the volunteer actors (lines 25-30). If actor *A* receives a notification message from a parent, it watches its link to that parent. If connectivity is lost, then it will perform cascaded relocation to re-establish it.

```
VCR (A)
If (neighboring actor F fails) {
  Recovered = False
  While (not Recovered)
    Switch (True) {
      Case: (d (A, F) ≤ α r_max) && (LF (A) ≥ β)
        // A becomes a volunteer actor
          NotifyMove   //notifies children before moving
        Move to a distance γ r_max from F
          Gradually increase transmission power
          If (connected to another volunteer V) {
              Recovered = True
          } // End If
      Case: (d (A, F) >  α r_max) && LF (A) < β)
        // A becomes a horrid actor
          Wait to hear from a volunteer within  τ time units
          If (message arrived from a volunteer V) {
               Recovered = True
          Else
             Reduce α and/or β
          } // End If
    } // End switch
  } // End while
  DiffuseActors( )
}  // End If
If (Notified by a volunteer actor)  {    // i.e., A is a child actor
  If (lost connection to parent) {
    NotifyMove   // inform the children or A
     Move towards parent & increase transmission power
  } // End If
} // End If
```

Figure 3.4: Pseudo code for the VCR algorithm

The example in Figure 3.5 illustrates how VCR restores connectivity after node *F* fails. Actors A, B, D, E and H detect the failure as shown Figure 3.5(a). Actors A, B, D and E declare themselves as volunteers based on their proximity and legibility factor, whereas, H becomes horrid. Volunteer actors notify their children and then move towards *F* while increasing their transmission power in order to maintain their connectivity with children. This movement may break some communication links like A$\rightarrow$C. Horrid actor H waits for volunteers B and D to become connected as shown in Figure 3.5(b). When a child actor “C” loses its link with parent “A”, it performs cascaded relocation to re-establish connectivity with “A” as shown in Figure 3.5(c). Boosting the transmission power may lead to increased contention and interference. Therefore, volunteer actors apply diffusion forces based on their proximity and transmission power as shown in Figure 3.5(d).

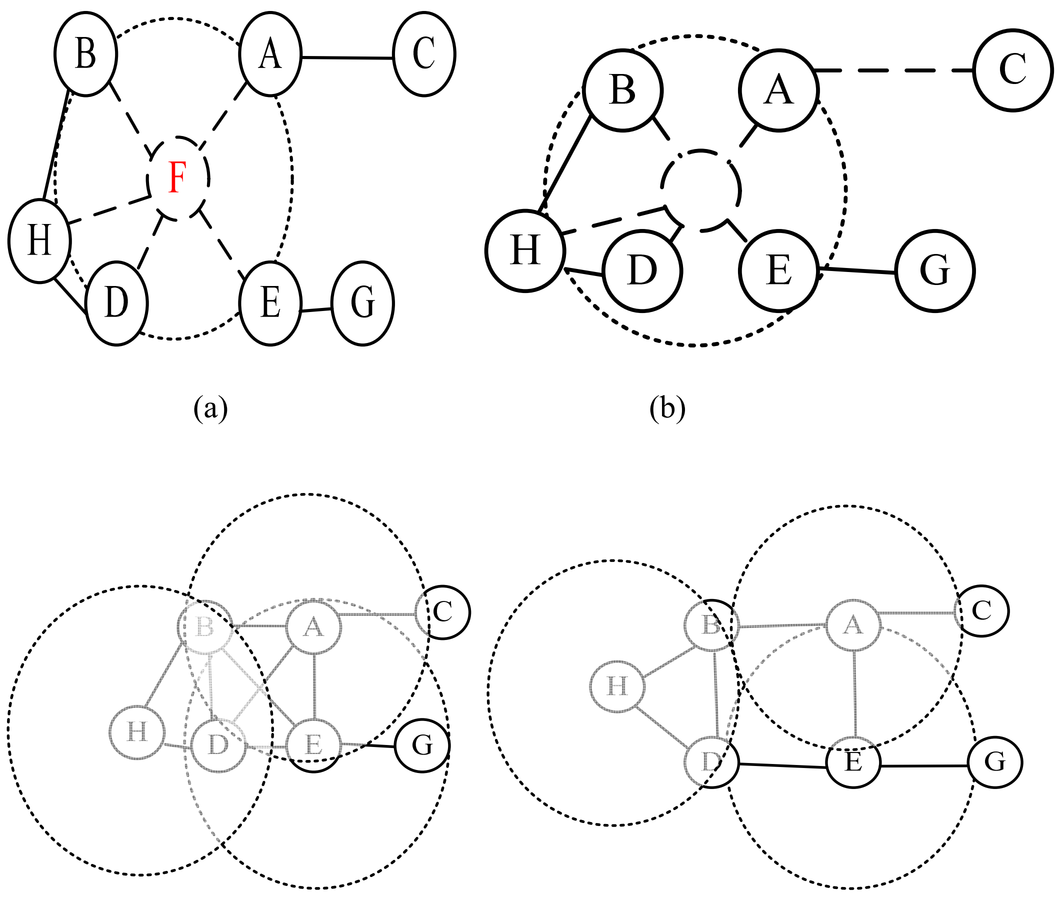


Figure 3.5: An example to demonstrate the operation of VCR

3.3 Performance evaluation

We have conducted an extensive survey of existing performance evaluation tools for wireless sensor networks and presented in (Imran et al. 2010a). The performance of the proposed algorithms is evaluated through extensive simulations. This section describes the network operation model, simulation environment, performance metrics, baseline approaches and experiment setup. The results of the experiments are analyzed and interpreted in the subsequent subsection.

3.3.1 Network operation model

The effectiveness of the proposed algorithms is validated through a simulated target tracking application environment. Sensors probe their surroundings, detect the target and report it to the actors. The actor nodes can receive information from the sensors and perform certain actions against the target. We have developed a simulator in Visual C++ and adapted the node parameters and network operation model of (Akkaya et al. 2005).

3.3.2 Simulation environment

In the simulation experiments, we have randomly placed 800 sensor nodes in an area of 1000 m × 600 m. The sensor nodes are assumed to have an initial energy of 5 joules and a buffer size to accommodate 15 packets. A node is considered non-functional if it is depleted of energy. The maximum transmission range of sensors is assumed to be 100 m (Atwood et al. 2000). A free space propagation channel model is assumed (Andersen et al. 1995) with the capacity set to 2 Mbps. Sensor nodes are grouped into clusters (Gupta and Younis 2003) in order to make the network scalable and each cluster uses the routing protocol proposed in (Younis et al. 2002).

We have created connected inter-actor topologies that consist of varying the number of actor nodes (20-100) and placing them randomly in the same area. Actors are assumed to communicate with each other. We have varied the initial transmission range of the actors (50-200) so that the topology becomes strongly connected.

Moreover, we assumed an action range of 50 m for each actor that is a circular area where that actor can perform certain actions (e.g. extinguish fire).

3.3.3 Performance metrics

The performance is assessed using the metrics presented in Table 3.1.

Table 3.1: Performance metrics for VCR algorithm

Performance Metrics	Description
Total distance moved	The *total distance moved* by all nodes involved in the recovery: This gauges the efficiency of VCR in terms of the overhead involved. We measure the distance until the connectivity is restored and the total travelled distance after the diffusion forces are applied to spread the nodes.
Number of nodes	The *number of nodes* moved during the recovery: This metric reflects the scope of the recovery process.
Number of messages	The *number of messages exchanged* among nodes: Again this metric indicates the recovery overhead.
Percentage of coverage reduction	The *Percentage of coverage change* (increment or decrement) relative to the pre-failure level: Although connectivity is the main objective of VCR, node coverage is important for many setups. The loss of a node usually has a negative impact on coverage. This metric assesses whether VCR alleviates or worsens the coverage loss.

The following parameters were used to vary the WSAN configuration in the simulation experiments:

- The *number of deployed nodes (N)* in the network affects the node density and the inter-actor connectivity.
- The *node communication range (r)* influences the network connectivity and highly affects the recovery overhead in terms of distance traveled and the number of actors involved.

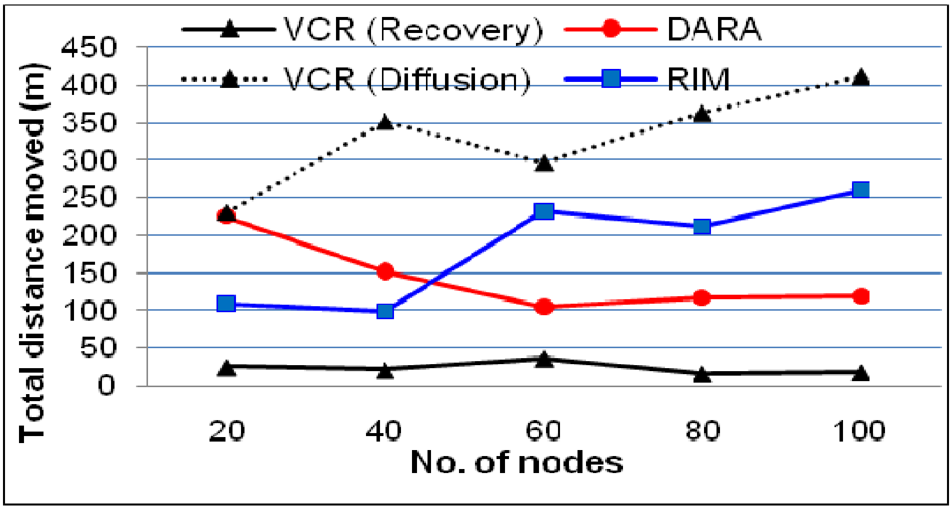
3.3.4 Baseline approaches

We compare the performance of VCR to that of DARA (Abbasi et al. 2007; Abbasi et al. 2009b) and RIM (Younis et al. 2010; Younis et al. 2008). Both DARA and RIM are distributed algorithms and are similar to VCR in the sense that both exploit node relocation in order to restore connectivity. However, their procedure is different. When node *F* fails, DARA selects the best candidate *A* among its 1-hop neighbors and replaces it. The algorithm is recursively applied to tolerate connectivity loss due to movement, i.e., *A* will be replaced with one of its neighbors and so on. On the other hand, RIM moves all the 1-hop neighbors towards *F* until they become connected. Like DARA, RIM is applied recursively to re-establish the links that get severed by node movement.

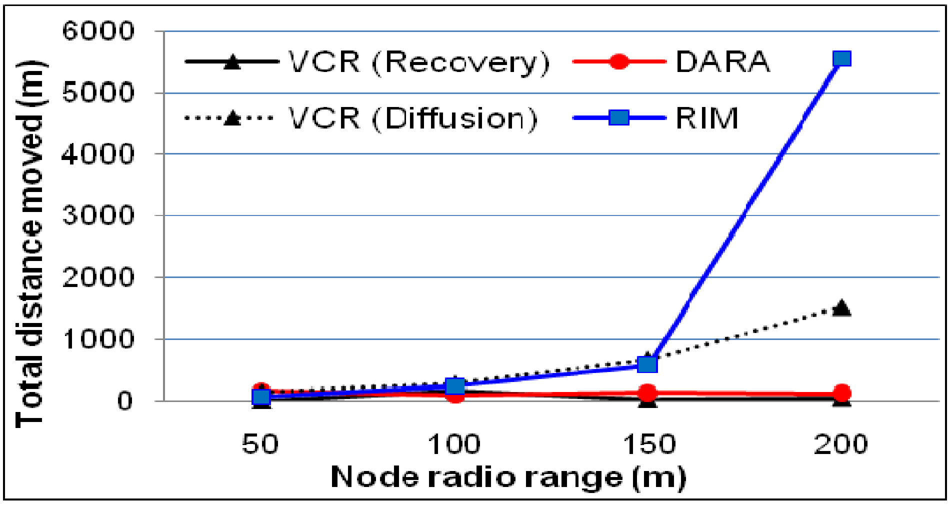
3.3.5 Results and analysis

The simulation experiments involve randomly generated WSAN topologies with a varying number of actors and their communication ranges. The number of actors has been set to 20, 40, 60, 80 and 100. The communication range of actors is changed among 50, 100, 150 and 200. When changing the node count, "*r*" is fixed at 100m; and "*N*" is set to 60 while varying the communication range. The results of individual experiments are averaged over 30 trials. All results are subject to 90% confidence interval analysis and stay within 10% of the sample mean.

a. **Total distance moved:** Figure 3.6 shows the distance travelled by all nodes until the connectivity is restored as well as after the self-spreading step of VCR is complete. As far as restoring the connectivity is concerned, VCR significantly outperforms both DARA and RIM because it only moves nodes in the close vicinity of *F*. As both graphs in the figure indicate, the performance advantage of VCR remains consistent even with higher node density and transmission range. This is because VCR strives to limit the involvement of nodes that are far from the failed actor and limit the scope of cascaded actor relocation by pursuing a higher transmission range. Figure 3.6-(a) indicates that the performance of VCR without applying the diffusion forces scales very well and is not affected by the node density given the optimized selection of volunteers as explained in Section 3.2.3.



(a)



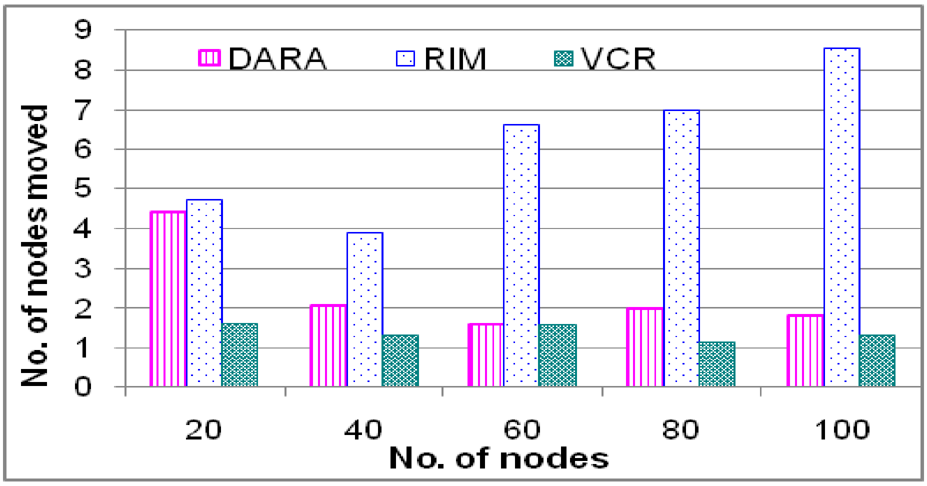
(b)

Figure 3.6: The total distance moved by all nodes during the recovery until restoring connectivity, as a function of *N* in (a) and *r* in (b).

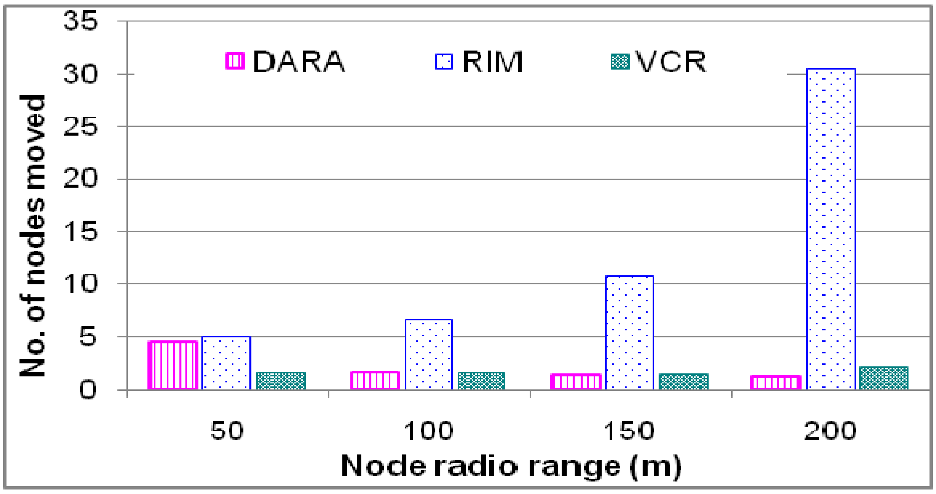
Similar observation can be made for the communication range (Figure 3.6(b)), where the connectivity-restoration overhead is minor compared to the baseline approaches.

However, Figure 3.6 also shows that the self-spreading step is costly in terms of the motion overhead. This is mainly because the scope of the motion is wider and involves nodes that do not have to relocate in order to restore the connectivity. It is worth noting that signal interference is not cared for in RIM and DARA and spreading the nodes is not applied. Also, the self-spreading step will boost the coverage achieved by VCR as shown later in this section.

b. **<u>Number of nodes moved:</u>** Figure 3.7 shows the number of recovery participants when VCR and the baseline approaches are applied. The performance graphs confirm the advantage of VCR which moves fewer actors than RIM and DARA because it limits the scope of recovery and avoids recursive cascaded relocations by exploiting partially utilized transmission ranges. Furthermore, the performance of VCR remains almost constant while varying the number of nodes and their radio range, which indicates great scalability.



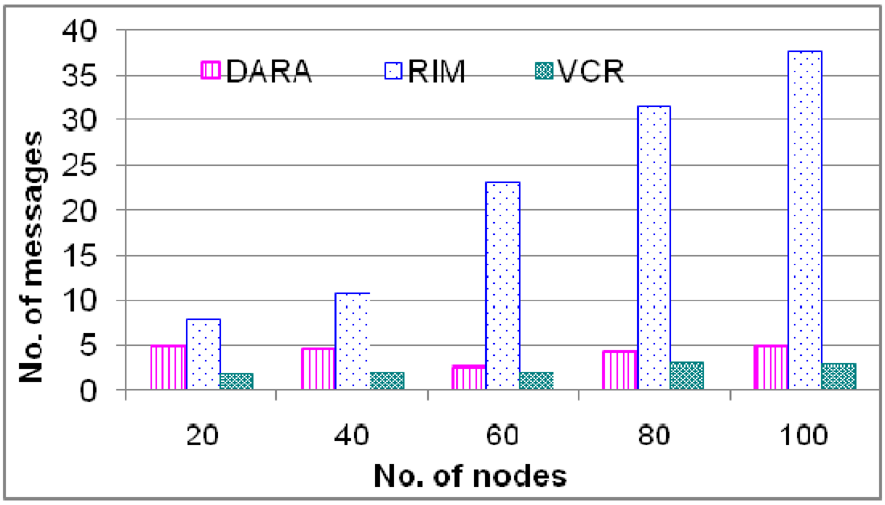
(a)



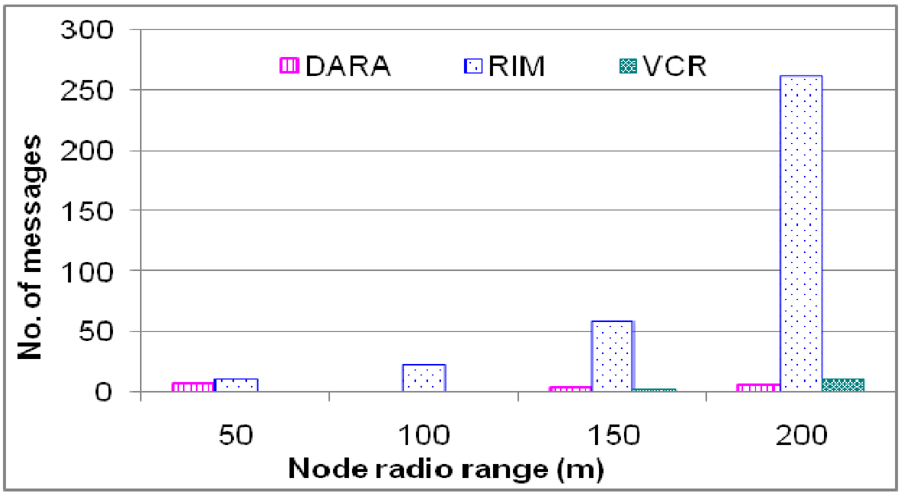
(b)

Figure 3.7: The number of nodes moved during the recovery, while varying the network size (a) and radio range (b).

c. **Number of messages exchanged:** Figure 3.8 reports on the messaging overhead as a function of the network size and radio range. As indicated in the figure, VCR introduces far fewer messages than DARA and RIM. This is because VCR strives to engage only the closest nodes among the 1-hop neighbors of *F*. On the other hand, Figure 3.8 indicates that the messaging overhead in RIM significantly grows for a high actor density and long communication range because the number of neighbors increases in both cases.



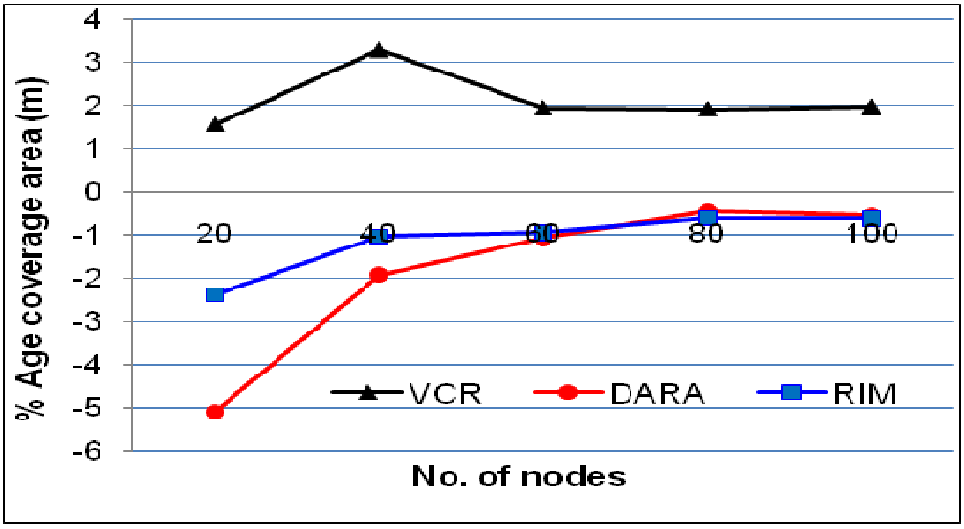
(a)



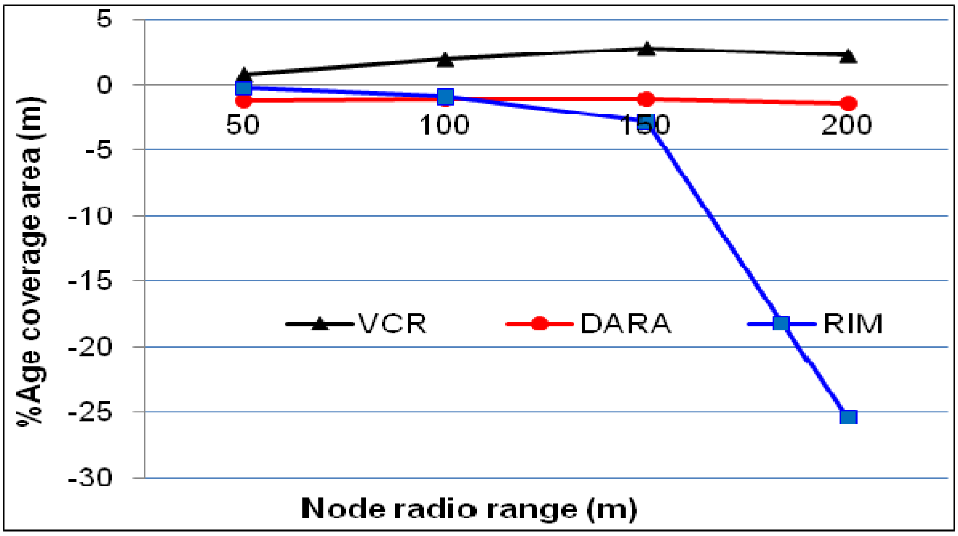
(b)

Figure 3.8: The effect of changing *N* (a) and *r* (b) on the total number of messages exchanged by all nodes during the recovery.

d. **Percentage of coverage improvement:** Figure 3.9 shows the impact of change *N* and *r* on coverage measured in terms of percentage of coverage improvement relative to the pre-failure level. The action range is set to 50 m in these experiments. Overall, VCR improves coverage by about 2% and consistently outperforms both DARA and RIM. While increasing the node density helps, DARA and RIM still do not make up for the coverage loss and definitely do not



(a)



(b)

Figure 3.9: The coverage improvement after recovery, as a function of *N* in (a) and *r* in (b).

match VCR's performance. The advantage of VCR in terms of coverage is obviously attributed to the actor diffusion performed to limit the effect of signal interference after restoring connectivity. Figure 3.9(b) indicates that for VCR the coverage grows with increasing the communication range while the performance of DARA is not affected much under this metric. On the other hand, the performance of RIM significantly worsens when growing the communication range. With the increased value of r, the network becomes more connected and the number of neighbors of F grows. RIM moves nodes inwards making the area around F more crowded than at the network periphery and thus causing a significant loss of coverage.

General comments: Generally, each application has different priorities such as coverage, resource efficiency, interference etc. depending upon the environment and nature of the application. The results presented in the experiments are generic in nature. The application designers are expected to assess the application-level interests and configure the priorities accordingly. The performance results presented above shows that the self-spreading step of VCR is costly in terms of the movement overhead. However, it significantly improves the coverage.

3.4 Chapter Summary

In this chapter, we have presented a distributed Volunteer-instigated Connectivity Restoration (VCR) algorithm. VCR is a reactive approach that avoids performing a network-wide analysis to assess the impact of an actor failure. Neighbors of the failed actor *F* detect the failure and volunteer by increasing their partially utilized transmission range and moving towards *F*. VCR applies diffusion force among volunteer actors based on their transmission range in order to minimize the possibility of interference and spread them for better coverage. The simulation results have confirmed the effectiveness of VCR and validated the superiority of its performance compared to published schemes.

Most of the existing localized approaches impose unnecessary recovery overhead because they do not assess the impact of node failure on connectivity. Moreover, reactive approaches might not be suitable for most applications of WSANs because they may not tolerate actor failure over a long period. In the next chapter, we present a hybrid <u>P</u>artitioning detection and <u>C</u>onnectivity <u>R</u>estoration (PCR) algorithm that strives to minimize recovery time and overhead.

CHAPTER 4

PARTITIONING DETECTION AND CONNECTIVITY RESTORATION ALGORITHMS

As discussed in chapter 2, most of the existing real time connectivity restoration approaches may not be suitable for mission-critical time-sensitive WSAN applications due to higher recovery time and overhead. This chapter presents a novel hybrid Partitioning detection and Connectivity Restoration (PCR) algorithm to recover from an actor failure. In this chapter, we present a variant of PCR named partitioning Detection and Connectivity Restoration (DCR). This chapter also presents a novel hybrid Recovery Algorithm to handle a special case of Multiple (RAM) actor failures. The rest of the chapter is organized as follows. Section 4.1 presents an overview of the proposed algorithms PCR, DCR and RAM. The PCR and DCR algorithm details, analysis and performance evaluation are presented in Section 4.2. Section 4.3 presents the design, analysis and performance evaluation of RAM. The chapter is finally concluded in Section 4.4.

4.1 Overview of partitioning detection and connectivity restoration algorithms

Due to high recovery overhead, real time or reactive approaches may not be suitable for resource constrained WSANs. Moreover, mission-critical time-sensitive applications may not tolerate actor failure over a long period. Therefore, this chapter presents a novel distributed hybrid Partitioning detection and Connectivity Restoration (PCR) algorithm which proactively determines potential critical actors and rapidly repairs the topology with little overhead. First, each actor proactively assesses its criticality, i.e., being a cut-vertex in the network topology, in a distributed manner based on the local information. Each critical (primary) actor designates an appropriate neighbor (preferably non-critical) as its backup. The backup actor continuously monitors its primary for possible failure. Upon failure detection, the backup actor initiates a recovery process that may involve coordinated relocation of multiple actors. This chapter also presents DCR; a distributed hybrid partitioning Detection and Connectivity Restoration algorithm that improve the backup selection criteria of PCR. We analyze the performance of PCR confirmed through simulation. Simulation results validate the performance of PCR and DCR that are effective and efficient compared to contemporary schemes found in literature.

Both PCR and DCR assume single critical actor failure at a time and no other node fails during the recovery process. Although, the possibility of concurrent multiple actor failure is exceptional, it may be precipitated by a harsh environment or disastrous events such as explosions in a battlefield. Recovery from such failures is very challenging and requires careful consideration especially when the failed actors are neighbors. We extend DCR to handle a special case of multi-node failure. This chapter also presents a novel Recovery Algorithm to handle simultaneous Multiple node failure (RAM). RAM identifies critical actors and designates for them distinct backups. The designated backups detect the failure of the primary (s) and execute the recovery process concurrently. Like DCR, the recovery procedure is applied recursively until connectivity is restored. To the best of our knowledge, RAM is the first localized hybrid approach that recovers from simultaneous failure of multiple actors while maintaining minimal network state information. This work analyzes the performance of RAM. Simulation results validate the performance of RAM in terms of incurred overhead.

4.2 Recovery from single critical actor failure

As discussed earlier in Section 1.4, failure of an actor is more likely in WSANs and impact of an actor failure can be so significant that the network may become dysfunctional. This section details the PCR and DCR algorithms that are designed to recover from the failure of a critical actor. The details of the algorithms are in the following subsection respectively.

4.2.1 Design of partitioning detection and connectivity restoration algorithm (PCR)

This subsection presents the system model, application scenario, problem definition, design of the proposed PCR algorithm, PCR analysis, pseudo code of PCR and performance evaluation. PCR is a hybrid algorithm that proactively identifies critical actors and assigns them backups to avoid recovery delays and unnecessary movements. The designated backup continuously monitors their primary through heartbeats. Once the failure is detected, the backup initiates a recovery process that may involve controlled and coordinated multi-actor relocation.

4.2.1.1 System model, application scenario and problem definition

We have adopted the same system model described in Section 3.1. As mentioned earlier, actors need to collaborate and coordinate with each other on planning an optimal response and synchronize their operations. For example, in Urban Search and Rescue (USAR) applications, sensors and actors are deployed in an area that got damaged by events such as fire, earthquake, disaster, etc. For instance, search and rescue robots from the University of South Florida (USF), Foster-Miller, iRobot, Space and Naval Warfare Systems Centre (SPAWAR), UROBOT, etc. participated in order to search for disaster survivors during 9/11 in situations too difficult or dangerous for humans (Casper and Murphy 2003). The sensors detect the presence of survivors in the vicinity and report it to the actors. The actors equipped with necessary life support equipment receive the sensors data, process it and share it with peer actors to identify the most appropriate set of actors. These actors are responsible for rescuing survivors immediately or providing them with life saving necessities such as water, oxygen or even some sort of medicine within a short period until the rescue team arrives. The role of actors is extremely crucial for a timely response to prevent serious consequences. Therefore, the actors should collaboratively identify the most appropriate set of actors based on their capabilities, proximity, etc. that will participate in the operation. This requires that actors can communicate with each other and cover the particular area. Therefore, actors establish and maintain inter-actor topology in order to enable such communication.

In such WSAN applications, failure of one or multiple actors not only partition the inter-actor network into disjointed segments but leave a coverage hole where there is no actor to receive and respond to the sensors data. Consequently, an inter-actor interaction may cease and the network becomes incapable of delivering a timely response to a serious event. This may risk the life of some survivors and lead to catastrophic events. Therefore, recovery from an actor failure is of utmost importance. Since, WSANs operate autonomously in unattended setups, replacing the failed actor is often infeasible and the recovery should be a self-healing and agile process that involves reconfiguring the inter-actor topology. The criticality of the applications and the resource constrained nature of these networks necessitate low restoration time and reduced overhead.

The impact of an actor's failure depends on the position of that actor in the network topology. A node is said to be critical, cut-vertex in graph theory terminology, if its removal partitions the network into disjointed segments (Jorgic et al. 2004). The failure of a critical actor not only affects the actor coverage but significantly impacts inter-actor connectivity. For example, consider a network topology depicted in Figure 4.1. Losing a leaf/non-critical node, such as *G* does not affect inter-actor connectivity. Meanwhile, the failure of critical node such as *F* partitions the network into disjointed blocks. In order to tolerate critical node failure, three approaches are identified: (i) proactive (ii) reactive and (iii) hybrid. Proactive approaches establish and maintain a bi-connected topology in order to provide fault tolerance. This necessitates a large actor count that leads to higher cost and becomes

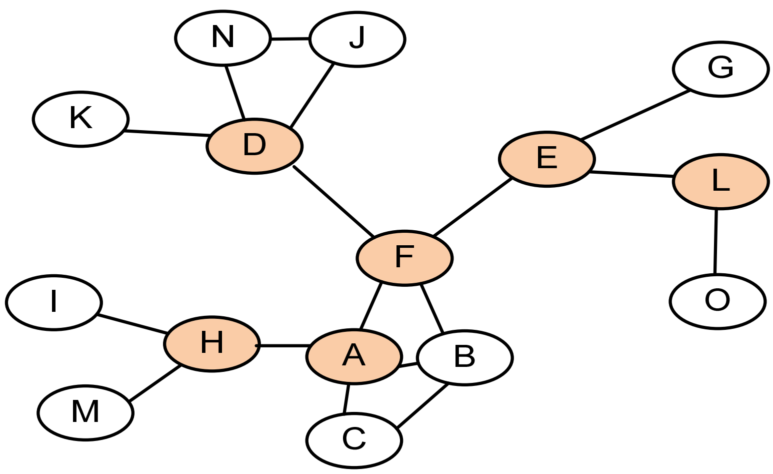


Figure 4.1: An example of connected inter-actor network topology.

impractical. On the other hand, in reactive approaches the network responds only when a failure occurs. Therefore, reactive approaches might not be suitable for mission-critical time-sensitive applications. While in hybrid approaches, each critical actor proactively designates another appropriate actor to handle its failure when such a contingency arises in the future. We argue that a hybrid approach will better suit autonomous WSANs that are deployed for time-sensitive applications due to the reduced recovery time and overhead. This section focuses on restoring inter-actor connectivity lost due to failure of a critical actor. In this section, we consider one failure at a time and no other node fails during the recovery.

4.2.1.2 Approach overview

As mentioned in the previous section, hybrid algorithms better suit time-sensitive applications that require a rapid recovery. The proposed PCR algorithm is hybrid in the sense it consists of two parts i.e. proactive and reactive. In the proactive part, critical actors are determined using a localized algorithm that only requires 1-hop positional information. Once critical nodes (primary) are determined, they select and designate an appropriate neighbor (backup) to handle their failure when such contingency arises in the future. Each backup starts monitoring its primary through heartbeat messages. In the reactive part, a backup initiates a recovery process when the primary fails. The backup replaces the primary and cascaded relocations are performed until the recovery is complete. The detailed algorithm is described in following sections.

4.2.1.3 Identifying Critical Actors

As described earlier, the failure of a critical actor divides the inter-actor network into disjointed segments. PCR opts to identify a backup for each of these critical actors. Several algorithms to identify cut-vertices in a graph have been proposed in the literature. These algorithms can be categorized into centralized and distributed. Centralized algorithms (Duque-Anton et al. 2000; Goyal and J. Caffery 2002) require one of the nodes or the base station to be aware of the global topology. These methods involve huge communication overhead due to the dynamic nature of these networks. Frequent changes in the WSAN topology favours distributed and highly localized algorithms. Distributed detection algorithms, e.g., (Akkaya et al. 2008; Zamanifar et al. 2009), are based on CDS and require 2-hop neighbor information. Some localized algorithms such as (Jorgic et al. 2004) require only 1-hop neighbor's positional information at the expense of lower accuracy of cut-vertices identification. Basically, some nodes are marked as critical while they are not cut-vertices. However, no critical node will be missed. Given that PCR assigns backups that are preferably non-critical, such a category of approaches fits well and the reduced accuracy is not a major concern as will be discussed later in this section. Therefore, PCR employs a simple localized cut-vertex detection procedure that only requires 1-hop positional information to detect critical nodes. The procedure is based on (Jorgic et al. 2004) and runs on each node in a distributed manner to determine locally whether a node is critical or not.

Each actor determines locally whether it is critical or not based on its neighbor's position information. It calculates the distance between neighbors based on their positions. If the distance is less than their communication range, the actor is considered non-critical because neighbors would stay connected without it. On the other hand, if the 1-hop neighbors of an actor can be partitioned into more than one segment, the actor is 1-hop critical. For instance, Figure 4.2 shows a localized scope of non-critical node *A* and critical node *F*. Nodes *B*, *C*, *D* and *E* are 1-hop neighbors of node *A* as shown in Figure 4.2(a). Node *A* is 1-hop positional non-critical because its neighbors remain connected without *A*. On the other hand, neighbors of *F* can be divided in to two sub graphs i.e. {B, C} and {G, H, I}. Therefore, *F* is 1-hop positional critical as illustrated in Figure 4.2(b). Furthermore, leaf nodes such as *I* are

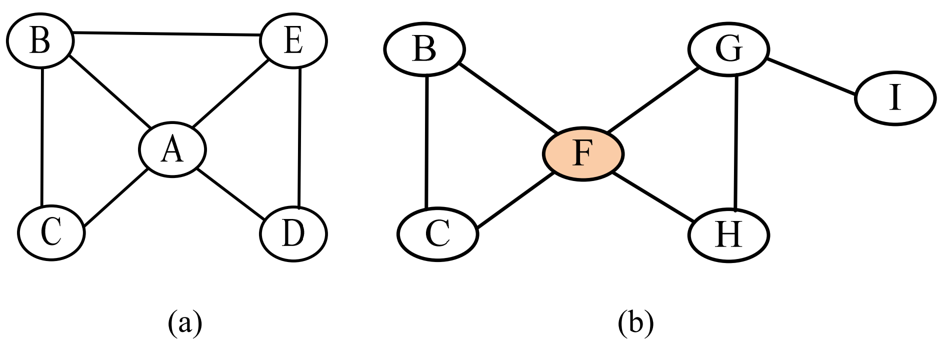


Figure 4.2: 1-hop positional critical/non-critical nodes

detected as non-critical, since there failure does not inflict inter-actor connectivity. Again, 1-hop positional critical nodes are not indeed cut-vertices all the time; obviously, the opposite is true. However, PCR pursues such approximate state determination in order to cut down on messaging overhead.

4.2.1.4 Backup selection and failure detection

Once the critical actors (primary) are identified, the next step is to select and designate an appropriate neighbor as backup. The purpose of the pre-nomination of backup nodes is to instantaneously react to the failure of a critical and avoid the possible network partitioning caused by such a failure.

a. Selection of backup: The actors maintain minimum state information (i.e. 1-hop neighbors) to avoid extra overhead of messaging. Since, neighbors become disconnected when a critical actor fails, backup actors are determined and notified before a failure of critical nodes takes place. The selection of a backup among 1-hop neighbors is based on the following ordered criteria:
   i. Neighbor actor status (NAS): As discussed above, each actor determines whether it is critical or non-critical. A non-critical neighbor actor is preferred to serve as backup. This will limit the scope of recovery and reduce incurred overhead.
   ii. Actor degree (AD): A non-critical neighboring actor with the least degree is a more suitable candidate for backup since few nodes will lose direct communication links to that backup when it moves. On the other hand, if the backup actor is a critical node, repositioning this actor causes more links to break and triggers a series of cascaded relocations by other nodes. A critical backup with few neighbors will limit the scope of the cascaded relocation and thus lower the overhead.
   iii. Inter-actor distance (ID): The least degree close backup actor is preferred in order to reduce the movement overhead and shorten the recovery time.

Once each critical actor selects an appropriate backup, it is notified in one of the exchanged heartbeat messages. An actor may be selected as backup for more than one actor. In case a backup actor fails or moves outside the range of the primary, its primary detects this through successive missing heartbeats and selects another backup using the same procedure specified above. Figure 4.3 shows the setup where each critical actor designates another as backup. The arrow head point towards the primary. Note that PCR does not require extra actors for serving as backup. It employs existing actors just to take care of each other.

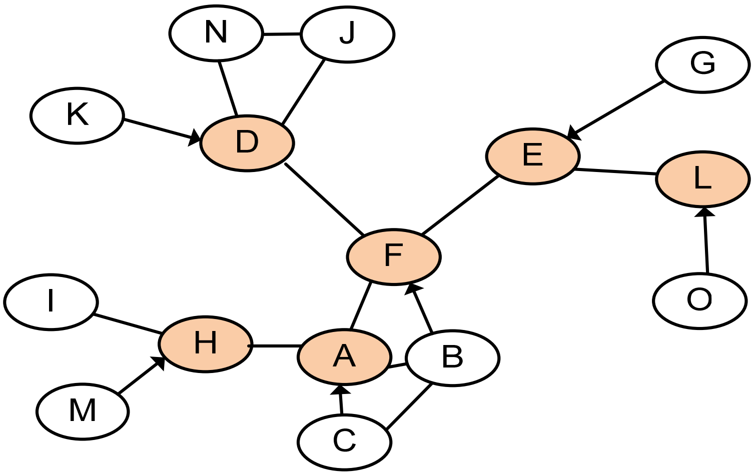


Figure 4.3: Critical actors select and designate their backup using PCR.

b. <u>Failure detection:</u> Once an actor receives a BACKUP message, it starts monitoring the primary through heartbeat messages. The failure of the primary is detected by the corresponding backup through successive misses of heartbeats. Figure 4.4 indicates that the backup node *B* detects the failure of primary *F* and triggers the recovery process as detailed in the following section.

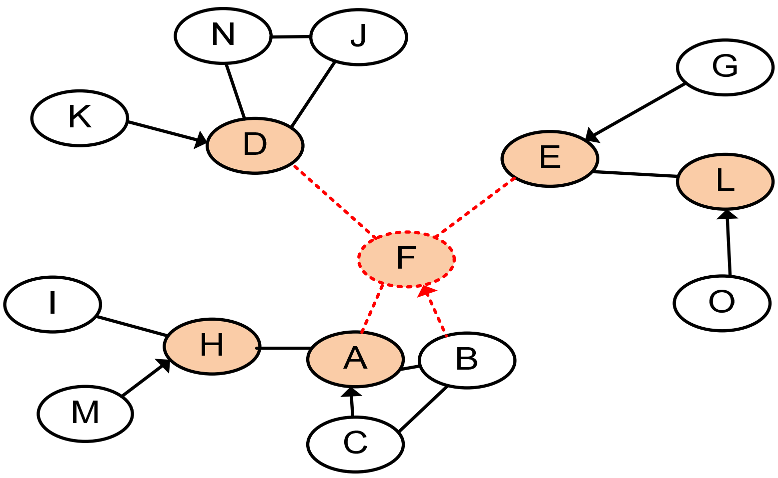


Figure 4.4: The backup actor *B* detects failure of the primary *F*

4.2.1.5 Recovery process

The reactive recovery process is initiated by the backup upon failure detection of the primary. The scope of recovery depends on the NAS. If the backup is a non-critical actor, then it simply replaces the primary and the recovery would be complete. However, if the backup is also critical node, cascaded relocation is performed. Basically, repositioning of actor A_i in response to the failure of A_f will be interpreted by its backup A_j as if A_i is lost and A_j will thus move to replace A_i. The recovery process consists of the following steps:

a. Primary recovery: The backup actor immediately initiates a recovery process once it detects failure of its primary. The scope of recovery depends on the position of the backup actor which can be among the following three scenarios. First, if a backup is a non-critical node then the scope of recovery is limited because it does not require further relocations. The backup actor moves to the location of the failed primary and exchanges heartbeat messages with its new neighbors. It selects and designates a new backup since it has become a critical node at the new position. This movement alerts the other primary nodes (if any) at the previous location to choose a new backup. An illustrative example is provided in Figure 4.5, where non-critical backup B simply replaces its primary (i.e. F) and selects a backup for itself.

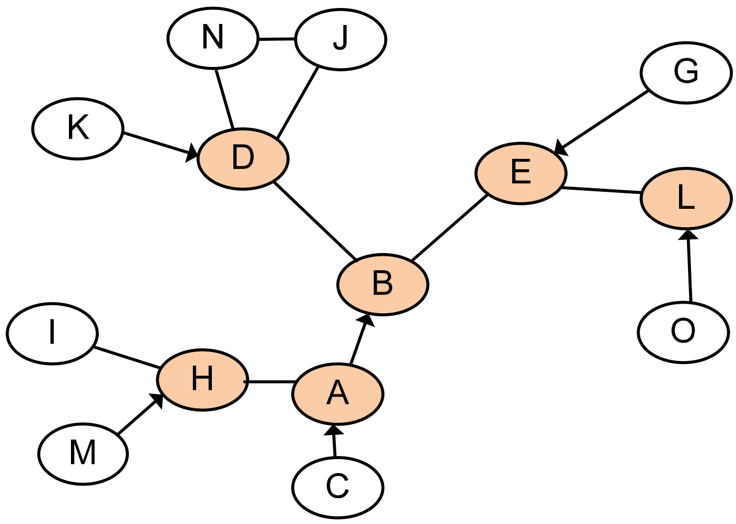


Figure 4.5: Recovery process when a non-critical backup replaces the failed primary

The second scenario is, when the backup is also a critical node. In this case, the backup actor will notify its own backup so that the network stays connected. This scenario may trigger a series of cascaded repositioning of nodes as explained in subsequent subsection. The third scenario is when the failed (primary) and backup node are both critical nodes and simultaneously serving as backup for each other. This scenario is articulated in Figure 4.6. Actor *F* is serving as backup of another actor *B* and vice versa as shown in Figure 4.6(a). The actor *B* detects the failure of *F* as shown in Figure 4.6(b). Since, *B* is also a critical actor and its backup *F* has already failed, therefore, it selects another actor "*A*" as backup, as shown in Figure 4.6(c). Then *B* sends a movement notification message to its newly assigned backup (i.e. *A*) and moves to the position of *F* as illustrated in Figure 4.6(d). The backup actor *A* receives a movement notification message from primary node *B* and performs a cascaded relocation as discussed below and is shown in Figure 4.6(e) with *A* replacing *B*. Similarly, the algorithm is recursively

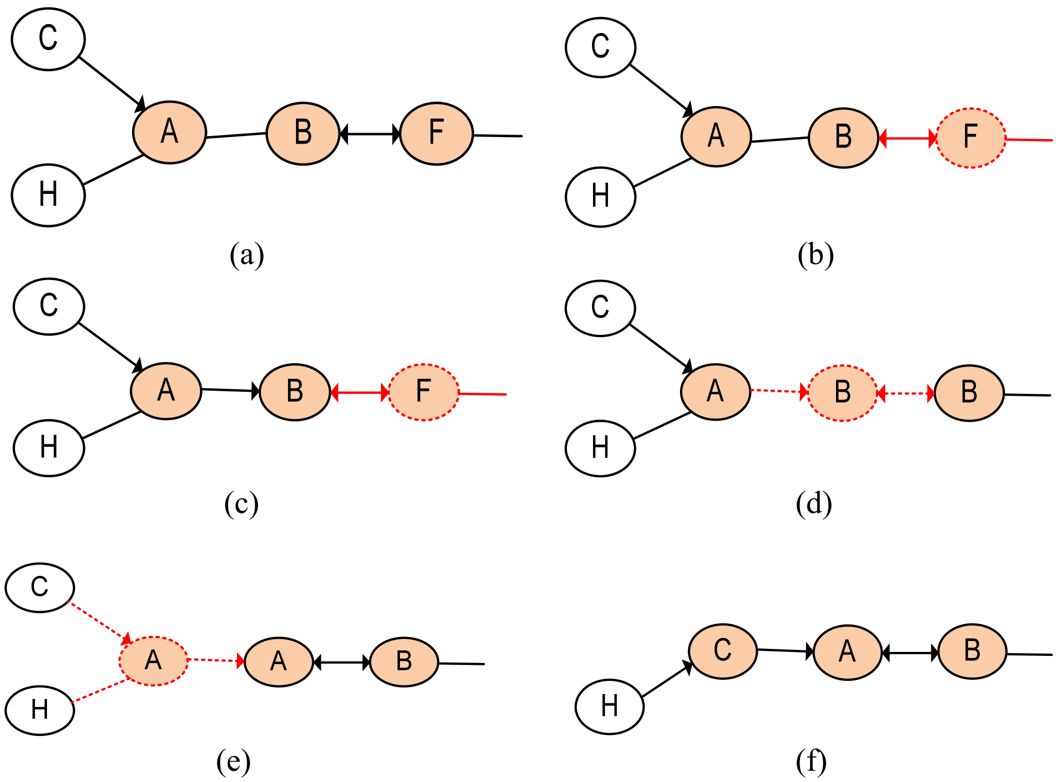


Figure 4.6: Applying the recovery process when two actors are simultaneously the primary-backup of each other.

executed on each backup until a leaf or a non-critical node replaces the primary. Figure 4.6(f) shows that the non-critical leaf actor *C* replaces the *A* and recovery is complete.

b. Cascaded relocation: As mentioned earlier, the position of the backup determines the scope of recovery. In particular, in the second scenario the recovery process is repeated to handle the departure of a backup node. Basically, when the critical backup actor *B* moves to the location of the failed actor, it waits to receive heartbeat messages from its own backup *BB*. Once node *B* receives heartbeat messages from *BB*, it selects and designates a new backup based on the new neighborhood that it has joined. This process may be again applied by *BB* and so on until a non-critical backup replaces a primary. Figure 4.7 illustrates this scenario where a backup actor is also critical and the recovery process continues in a cascaded manner. Failed actor *B* is replaced by another critical actor *A* (i.e. backup) as shown in Figure 4.7 (a). Since moving critical actor *A* further partitions the network, a cascaded relocation is triggered. Figure 4.7 (b) depicts where a non-critical backup actor *C* replaces a critical primary actor *A* and the connectivity is restored.

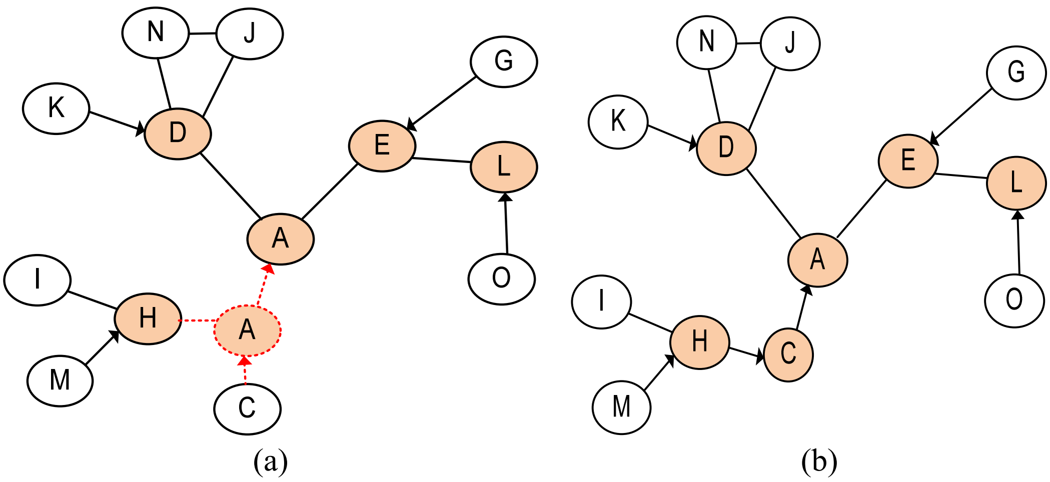


Figure 4.7: Recovery process when moving a critical backup actor triggers cascaded relocations.

4.2.1.6 PCR algorithm analysis

In this subsection, we analyze the performance of PCR algorithm. We show that PCR converge and successfully restore the connectivity of the network lost due to failure of critical actor. We prove the correctness and analyze the performance of the PCR algorithm. We introduce the following theorems:

Theorem 4.1: *PCR converge to form a connected topology, irrespective of the number of network segments created due to failure of a critical actor.*

Proof: Since actors have the symmetric communication links and the backup node will reposition at the location of the primary, all links in the vicinity of the primary will be restored to the pre-failure (pre-departure in case of cascaded relocation) level. Moreover, the cascaded relocations will stop when a non-critical node replaces a critical primary. Since each backup will only move once, PCR will be guaranteed to terminate. The worst case performance is when a critical node fails in the centre of the network and a non-critical backup is only available at the network periphery, i.e., a leaf node that has a node degree of 1.

Theorem 4.2: *PCR imposes a maximum travel distance overhead of r on each backup actor, where, r is the transmission range of the actor.*

Proof: As mentioned earlier, backup actors are selected among neighbors of a critical actor. Since we assume a free space propagation model, the maximum distance among the primary and the backup is equivalent to an actor radio range, i.e. r. Thus, the maximum distance a backup actor is required to travel to substitute the failed primary is r. Similarly, if the backup is also a critical node, it will be replaced by moving its backup in a maximum of r. PCR moves each backup only once, therefore, the maximum movement distance for each of the involved backup nodes is r.

Theorem 4.3: *PCR does not introduce new critical actors as a result of the recovery process when a non-critical (leaf actor) is involved in recovery.*

Proof: We prove this theorem by showing that PCR maintains existing links between nodes during the recovery. As discussed in Section 4.2.1.5, the recovery process consists of two steps. First, replacing a failed actor with a backup will re-establish all broken links with its neighbors. Second, if successive cascaded relocations are required, all critical nodes will be replaced by their backups. As shown in theorem 4.1

above, PCR terminates when a non-critical node moves. Therefore, PCR guarantees not to introduce new critical actors as a consequence of recovery.

Theorem 4.4: *The time it takes the PCR algorithm to converge while restoring inter-actor connectivity is O(r) where r is the communication range of actors.*

Proof: Since PCR proactively (before failure) designates for each critical actor a backup to handle its failure, the maximum time it takes a backup to substitute the failed actor is proportional to r, as proved in Theorem 4.2. If moving a critical backup further triggers c relocations, the total recovery time will be proportional to $(c + 1) *r$ because of sequential relocations. Thus PCR convergence time to restore connectivity in the worst case is $(c +1) *r$ which is $O(r)$.

Theorem 4.5: *The total message complexity of PCR is O(N) where N is the number of actors in the WSAN.*

Proof: Since PCR only maintains 1-hop neighbor information to rejuvenate inter-actor connectivity; this requires 1 message for each actor. Moreover, every critical node participating in the recovery has to send 1 movement notification message to its backup. PCR does not count message exchange with neighbors at the new location, it considers it as part of the regular status update for maintaining a 1-hop table. Thus, in the worst case, when all the critical nodes $(c-1)$ move, the total number of messages will be $(N+c-1)$. Therefore, PCR incurs a total message complexity of $O (N)$.

4.2.1.7 Pseudo code of PCR algorithm

Figure 4.8 shows the high level pseudo code of the PCR algorithm running on each actor in a distributed manner. Initially, all the actors are initialized as non-critical (line 1). The localized cut-vertex detection procedure determines whether node A is critical or not (lines 2-4). If actor A is critical, then it will select and designate an appropriate backup actor from among the neighbors (lines 5-8). The selection of the backup is made based on the criteria specified in section 4.2.1.4. The backup actor A detects failure of its primary by continuously monitoring its health through HEARTBEAT messages. Upon detecting the failure of the primary, it initiates the recovery process (lines 9-11). If backup actor A is non-critical, it simply moves to the location of F (lines 12-13). If node A is critical and simultaneously a primary and the backup of node *F* (i.e. SimPB), then it selects another node as backup. In other words, node *A* and *F* were serving as backup for each other. Since, in the case of failure of *F,* node *A* not only loses its primary but its backup also. Therefore, it appoints another backup before going to replace *F* (lines 14-16). Node *A* notifies its movement to the

```
PCR (A)
1   critical-node(A) ←false
2   if Neighbors (A) becomes disconnected without A then
3       critical-node (A) ← True
4   endif
5   if (isCritical(A) == true)
6       BackUpID = FindBackUp(Neighbors(A))
7       AssignBackUp (BackUpID)
8   endif
9   If (Primary actor F fails) then
10              MoveToLocation(F, A)
11  endif
MoveToLocation (F, A)
12  If (critical-node (A) == false) then
13      Move (A, F)
14  else if (SimPB == true) then
15      BackUpID = FindBackUp(Neighbors(A))
16      AssignBackUp (BackUpID)
17      NotifyMove(A)
18      Move(A, F)
19      BackUpMoveOptimizer(A, Backup)
20  else
21      NotifyMove(A)
22      Move(A, F)
23      BackUpMoveOptimizer(A, Backup)
24  endif
```

Figure 4.8: High level pseudo code for PCR algorithm

newly assigned backup and moves to the location of *F* (lines 17-18). The newly assigned backup follows the same procedure to replace the primary *A* and the relocation process continues until a non critical node replaces the primary. The recursive procedure BackUpMoveOptimizer(A, Backup) is triggered on *Backup* to perform successive cascaded relocations each time a critical actor *A* moves (line 19). If node *A* is critical then it notifies its backup (say *Backup*) and moves to the location of *F*. This triggers the BackUpMoveOptimizer(A, Backup) to perform cascaded relocation until a non-critical node replaces the primary (lines 20-24).

4.2.2 Performance evaluation

This section describes the simulation environment, performance metrics, experiment setup and results and analysis.

4.2.2.1 Simulation environment

The experiments involve randomly generated topologies with a varying actor count and communication range. The number of actors has been set to 20, 40, 60, 80 and 100. The communication range of actors is changed among 50, 100, 150 and 200. When changing the node count, "r" is fixed at 100 m; and "N" is set to 60 while varying the communication range. The results of individual experiments are averaged over 30 trials. All results are subject to 90% confidence interval analysis and stay within 10% of the sample mean.

4.2.2.2 Performance metrics

The performance is assessed using the metrics presented in Table 4.1.

Table 4.1: Performance metrics for the PCR algorithm

Performance Metrics	Description
Total distance moved	The *total distance moved* by all nodes involved in the recovery: This gauges the efficiency of PCR in terms of energy efficiency and overhead involved in the recovery.
Number of nodes	The *number of nodes* moved during the recovery: This metric reflects the scope of the recovery process.
Number of messages	The *number of messages exchanged* among nodes: Again this metric indicates the energy dissipation and recovery overhead.
Percentage of coverage reduction	The *Percentage of coverage reduction* relative to the pre-failure level: Although connectivity is the main objective of PCR, node coverage is important for many setups. The loss of a node usually has a negative impact on coverage. This metric assesses the effectiveness of PCR in terms of coverage reduction.

The following parameters were used to vary the WSAN configuration in the simulation experiments:

- The *number of deployed nodes (N)* in the network affects the node density and the inter-actor connectivity.
- The *node communication range (r)* influences the network connectivity and highly affects the recovery overhead in terms of the distance traveled and the number of actors involved.

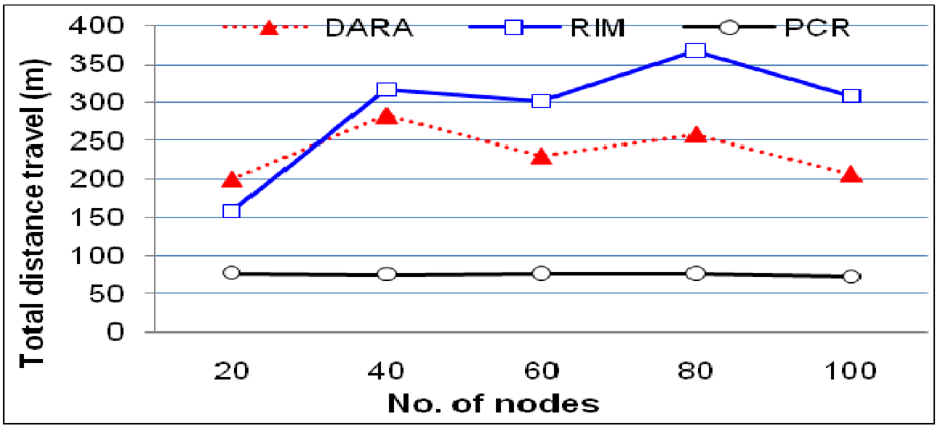
4.2.2.3 Baseline approaches

We compare the performance of PCR to that of DARA (Abbasi et al. 2007; Abbasi et al. 2009b) and RIM (Younis et al. 2010; Younis et al. 2008). Both DARA and RIM are distributed algorithms and are similar to PCR in the sense that both exploit node relocation in order to restore connectivity. However, their procedure is different. When node *F* fails, DARA selects the best candidate *A* from among its 1-hop neighbors and replaces it. The algorithm is recursively applied to tolerate connectivity loss due to movement, i.e., *A* will be replaced with one of its neighbors and so on. On the other hand, RIM moves all the 1-hop neighbors towards *F* until they become connected. Like DARA, RIM is applied recursively to re-establish links affected by the nodes movement. Both DARA and RIM are reactive approaches and do not provide for recovery ahead of time.

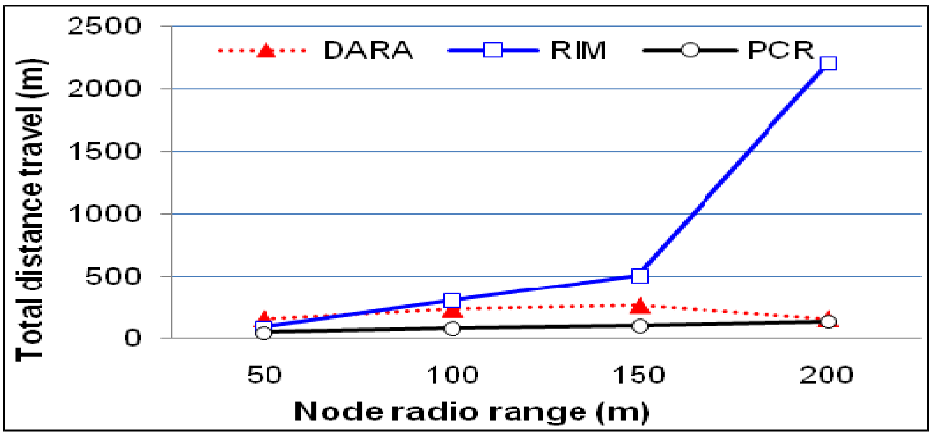
4.2.2.4 Results and analysis

The experiments involve randomly generated topologies with a varying actor count and communication range. The number of actors has been set to 20, 40, 60, 80 and 100. The communication range of actors is changed among 50, 100, 150 and 200. When changing the node count, "r" is fixed at 100 m and "N" is set to 60 while varying the communication range. The results of individual experiments are averaged over 30 trials. All results are subject to 90% confidence interval analysis and stay within 10% of the sample mean.

a. **<u>Total distance moved:</u>** Figure 4.9 shows the distance traveled by all nodes until the connectivity is restored. PCR significantly outperforms both DARA and RIM because it strives to move non-critical nodes to avoid cascaded relocations. As both graphs in the figure indicate, the performance advantage of PCR remains consistent even with higher node density and transmission range. This is because PCR strives to avoid moving critical nodes that cause further partitioning and require successive relocations. Furthermore, PCR performs cascaded relocations only when non-critical nodes in the neighborhood of the failed actor are not available. Figure 4.9-(a) indicates that the performance of PCR scales very well and is not affected by the node density because of choosing non-critical nodes as



(a)

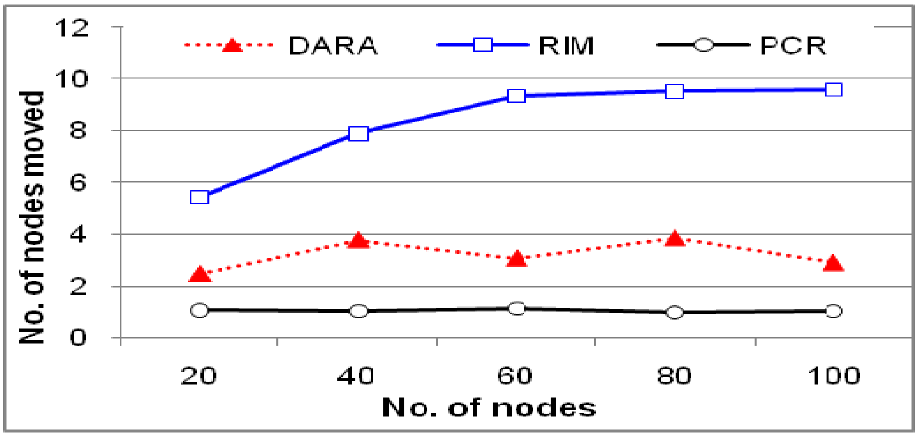


(b)

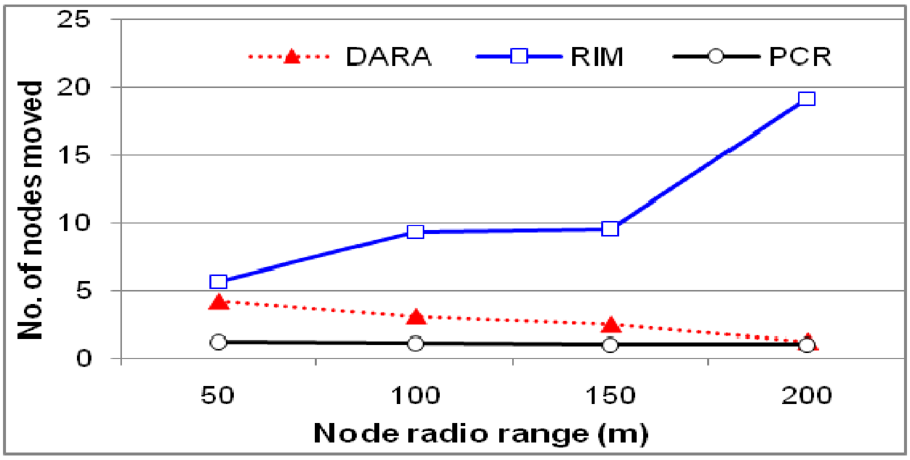
Figure 4.9: The total distance travelled by all nodes during the recovery until connectivity is restored, as a function of *N* in (a) and *r* in (b).

backup. Similar observations can be made for the communication range (Figure 4.9(b)), where the connectivity-restoration overhead is significantly less as compared to the baseline approaches.

b. **<u>Number of moved nodes:</u>** Figure 4.10 shows the number of nodes that are involved in the recovery when PCR and the baseline approaches are applied. The performance graphs confirm the advantage of PCR which moves fewer actors than RIM and DARA. This is because PCR limits the scope of recovery and avoids successive cascaded relocations by choosing non-critical nodes as backup. Furthermore, the performance of PCR remains almost constant while varying the number of nodes and their radio range, which indicates great scalability.



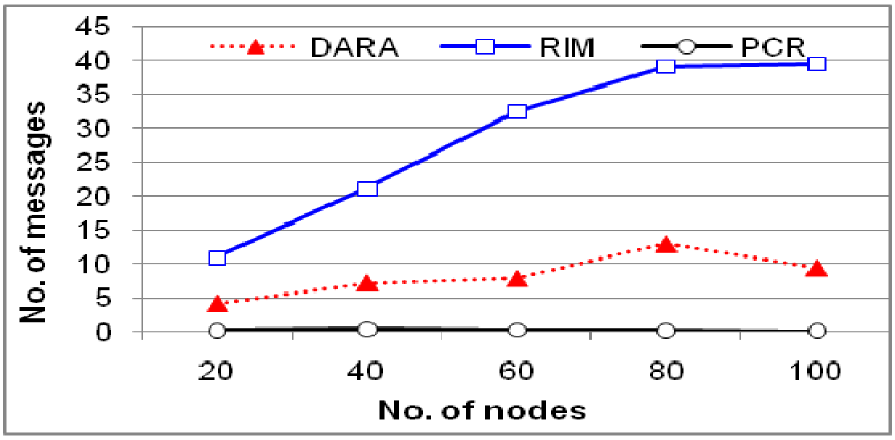
(a)



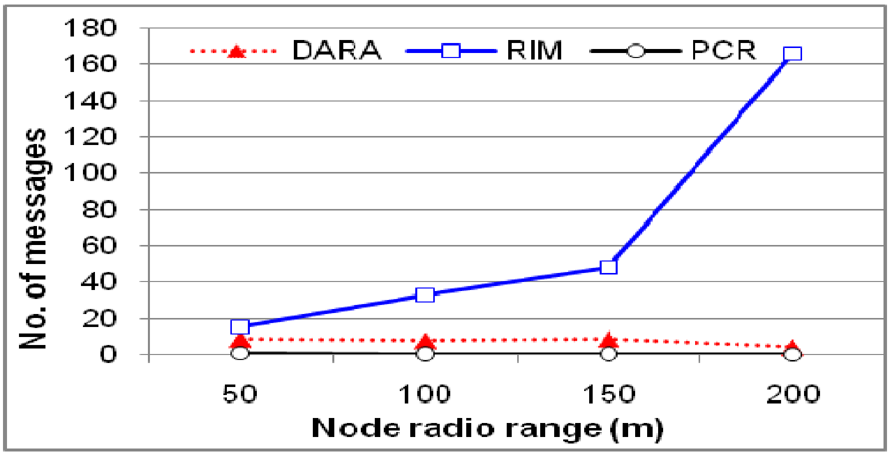
(b)

Figure 4.10: The number of nodes moved during the recovery, as a function of *N* in (a) and *r* in (b).

c. <u>**Number of messages exchanged:**</u> Figure 4.11 reports on the messaging overhead as a function of the network size and radio range. As the figure indicates, PCR incurs far less messaging overhead than DARA and RIM. This is because PCR limits message exchange only between a pair of primary and backup nodes instead of all 1-hop and 2-hop neighbors as is the case in RIM and DARA, respectively. Moreover, unlike DARA and RIM, PCR prefers to involve non-critical nodes in the recovery which limits the need for cascaded relocation and thus reduces the number of notification messages. Furthermore, heartbeat messages are not counted because they are considered as part of the network operation. Therefore, PCR makes the best use of these messages, since most of the backup nodes are non-critical and they are not required to send any message. The average number of



(a)

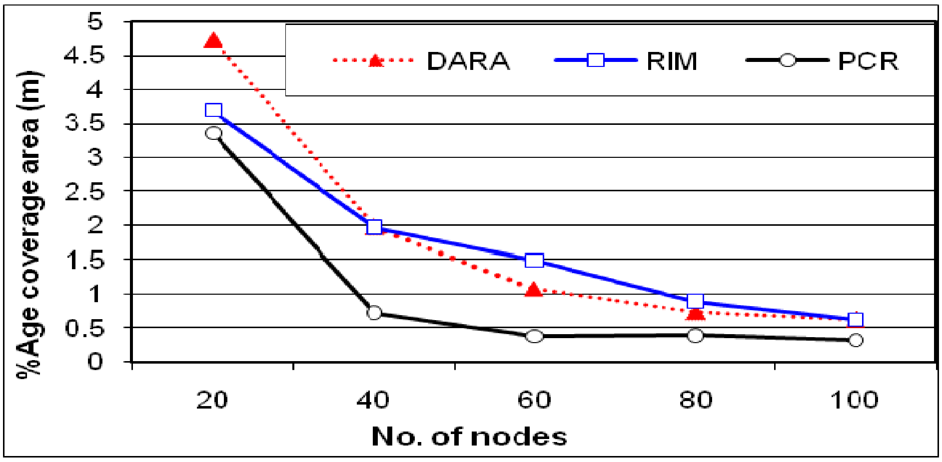


(b)

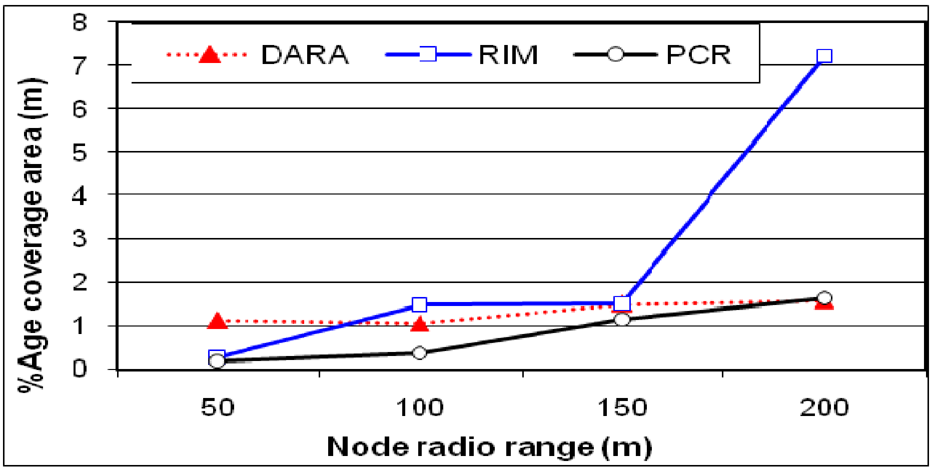
Figure 4.11: The effect of changing network size (a) and radio range (b) on the total number of messages exchanged by all nodes during the recovery.

notification messages sent by PCR in figure 4.11(a) and (b) are 0.18-0.39 and 0.16-0.57 respectively. On the other hand, Figure 4.11 indicates that the messaging overhead in RIM significantly grows for a high actor density and long communication range because the number of recovery participants increases in both cases.

d. **Percentage of coverage reduction:** Figure 4.12 shows the impact on coverage, measured in terms of percentage of coverage reduction relative to the pre-failure level, while changing the *N* and *r*. The action range is set to 50 m in these experiments. Overall, PCR limits the coverage loss and consistently outperforms both DARA and RIM. Although increasing the node density helps, DARA and



(a)



(b)

Figure 4.12: The coverage reduction after recovery, as a function of *N* in (a) and *r* in (b).

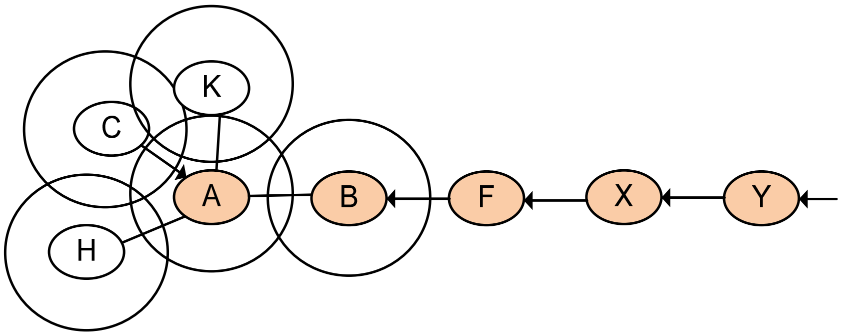
RIM still do not make up for the coverage loss and definitely do not match PCR's performance. The advantage of PCR in terms of coverage is obviously due to the limited scope of node relocation, which causes a coverage loss at the network periphery. Figure 4.12(b) indicates that the performance of PCR and DARA in terms of coverage reduction is not much affected with increasing the communication range. On the other hand, the performance of RIM significantly worsens when growing the communication range. With the increased value of r, the network becomes more connected and the number of neighbors of F grows. RIM move nodes inwards making the area around F more crowded while leaving uncovered parts at the network periphery and thus causing a significant loss of coverage.

4.2.3 Design of Partitioning Detection and Connectivity Restoration (DCR)

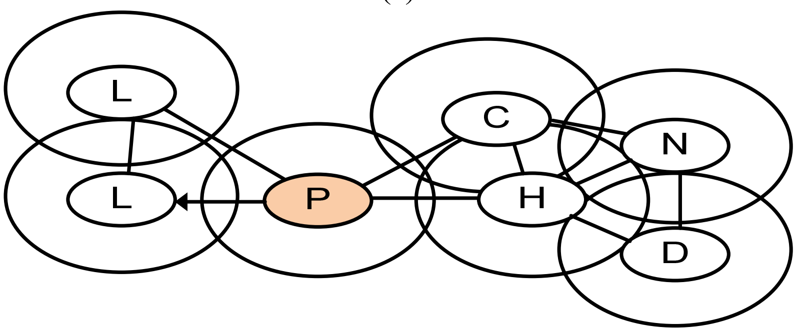
This subsection presents the problem definition, design, pseudo code and performance evaluation of the proposed DCR algorithm.

4.2.3.1 Problem Definition

As stated in the preceding Section 4.2.1.4, PCR prefers to designate the least degree actor (non-critical/critical) as backup. The rationale is that few nodes will lose direct communication links to that backup when it moves to recover from a primary failure. In other words, moving a weakly connected node will have a minimum impact on inter-actor connectivity. However, selection of least degree actors (except leaf) may lead to increased recovery overhead besides loss of coverage. Generally, weakly connected actors are critical and moving a critical node triggers a series of cascaded relocations. For example, Figure 4.13 (a) depicts the scenario where a critical primary actor *B* prefers to choose a least degree critical node *F* as backup. Upon failure



(a)



(b)

Figure 4.13: Impact of choosing a least degree actor as backup on inter-actor connectivity and coverage a) critical b) non-critical

detection of node *B*, backup *F* will initiate a recovery that leads to a series of successive shifted relocations until a non-critical node replaces the primary or network periphery is reached. This not only imposes an undesirable recovery overhead but causes a significant loss of coverage as well. Similarly, choosing a least degree non-critical node may not impact inter-actor connectivity but may cause reduction of coverage as shown in Figure 4.13 (b).

We propose an optimization to the backup selection procedure of PCR in order to reduce undesirable recovery overhead and preserve coverage. The partitioning detection and connectivity restoration (DCR) algorithm is also a hybrid approach but the criterion for backup selection is slightly different. DCR determines critical actors using the same procedure explained earlier in Section-4.1.1.3. Unlike PCR, DCR appoints high degree actors as backup to handle failure of primary actors. The details of the backup selection procedure are as shown below:

4.2.3.2 DCR algorithm design

This subsection presents the backup selection criteria of the partitioning detection and connectivity restoration (DCR) algorithm. Like PCR, the DCR is also a hybrid approach that determines critical actors in advance and designates them as backups. The designated backups continuously monitor their primary through heartbeats. Once the failure is detected, the backup initiates a recovery process that may involve controlled and coordinated multi-actor relocation.

Backup selection:

Once the critical actors are determined, DCR appoints each critical actor with a backup based on the following criteria. Like PCR, DCR prefers to appoint non-critical actors as backup. However, DCR differentiates non-critical actors into leaf and intermediate. DCR strives to get benefit out of both. A non-critical leaf actor (with the least degree) is a more suitable candidate for backup since moving that node will have a minimum impact on inter-actor connectivity. Otherwise, if a leaf node is not available in the neighborhood then DCR prefers to choose a high degree non-critical node because it will have more overlapping coverage. Moving such nodes will have a minimum impact on coverage. In other words, DCR strives to balance between coverage and connectivity. In case a non-critical node is not available in the neighborhood, DCR prefers to appoint a strongly connected critical node (with high degree) because there is more probability in having non-critical nodes in the neighborhood. This will not only limit the scope of cascaded relocations and lower the recovery overhead but will have a minimum impact on coverage. The rest of the backup selection and recovery procedure is similar to PCR.

4.2.3.3 Pseudo code of DCR

Figure 4.14 shows the high level pseudo code of DCR which will run on each actor "*A*" in a distributed manner. If an actor *A* is critical, it will select an appropriate backup actor using the AssignBackup() procedure (lines 1-3). While serving as backup to node *F*, if actor *A* either detects the failure of *F* or receives a movement notification message from *F*, it initiates a recovery process (lines 4-6).

The critical actor *A* finds an appropriate backup from among the neighbors. The AssignBackup() procedure preferably designates a non-critical neighbor (either leaf or with the highest degree) as backup. In case a non-critical node is not available, it chooses a critical actor with the highest degree and the least distance to *A* (line 7). The recovery procedure is executed on backup actor *A*, if it either detects the failure of primary *F* or receives a message from *F.* While executing the recovery procedure, *A* checks whether it is critical or not (line 8). If it is critical, it checks the status of its backup BackupStatus() before moving. If the backup of *A* has failed, it selects another node as backup. It then sends a movement notification message to inform the newly assigned backup or its pre-designated backup (lines 9-13). Now actor *A* can move to replace *F* (line 14).

```
DCR (A)
if (IsCritical(A) == true) then
    AssignBackup (A)
end if
if (A detects the failure of primary actor F or receives a movement
notification message from F) then
    Recovery (A, F)
end if
AssignBackup(A)
//Assign a non-critical neighbor with the highest degree and the least
   distance node as backup
Recovery(A, F)
if (IsCritical(A) == true) then
    if (A→BackupStatus() == Failed) then
        AssignBackup(A)
    end if
    NotifyBackUp(A)
end if
MoveToLocation (A, F)
```

Figure 4.14: High level pseudo code for DCR algorithm

4.2.4 Performance evaluation

The performance of DCR is validated through extensive simulations. This section describes the experiment setup, performance metrics, baseline approaches and experimental results.

4.2.4.1 Experiment setup and performance Metrics

We have created inter-actor topologies that consist of a varying number of nodes (20-100). Nodes are randomly placed in an area of 1000 m × 600 m. We have varied the transmission range of the actors (50-200) so that the topology becomes strongly connected. The performance is assessed using the metrics described in Table 4.2.

Table 4.2: Performance metrics for DCR algorithm

Performance Metrics	Description
Total distance moved	The *total distance moved* by all nodes involved in the recovery: This gauges the efficiency of DCR in terms of energy efficiency and the overhead involved in the recovery.
Number of nodes	The *number of nodes* moved during the recovery: This metric reflects the scope of the recovery process.
Number of messages	The *number of messages exchanged* among nodes: Again this metric indicates the energy dissipation and recovery overhead.
Percentage of coverage reduction	The *Percentage of coverage reduction* relative to the pre-failure level: Although connectivity is the main objective of DCR, node coverage is important for many setups. The loss of a node usually has a negative impact on coverage. This metric assesses the effectiveness of DCR in terms of coverage reduction.
Average node degree	*Average node degree:* measures the level of inter-actor connectivity and availability of alternative paths after recovery.

The following parameters were used to vary the WSAN configuration in the experiments:

- The *number of deployed nodes (N)* in the network affects the node density and the inter-actor connectivity.
- The *node communication range (r)* influences the network connectivity and highly affects the recovery overhead in terms of distance traveled and the number of actors involved.

4.2.4.2 Baseline approaches

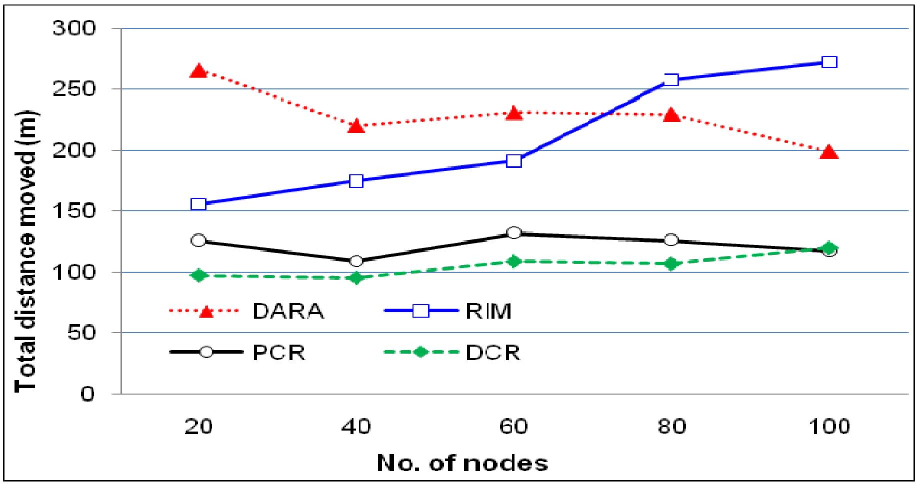
We compare the performance of DCR to that of DARA (Abbasi et al. 2007; Abbasi et al. 2009b), RIM (Younis et al. 2010; Younis et al. 2008) and PCR (Imran et al. 2010c). Like DCR, all the three algorithms are distributed and exploit node relocation to recover from node failure. However, their procedure is different. When node *F* fails, DARA selects the best candidate *A* among its 1-hop neighbors and replaces it. The algorithm is recursively applied to tolerate connectivity loss due to movement, i.e., *A* will be replaced with one of its neighbors and so on. On the other hand, RIM moves all the 1-hop neighbors towards *F* until they become connected. Like DARA, RIM is applied recursively to re-establish links affected by the nodes movement. Both DARA and RIM are reactive approaches and have no provision for recovery ahead of time. Like DCR, PCR is a hybrid approach that designates a backup node to each critical actor to recover from its failure. The pre-assigned backup detects the failure of *F* and moves to replace *F*. Like DARA and RIM, this algorithm is recursively applied to rejuvenate the broken links due to actor movement. The backup selection criterion of DCR is slightly different from PCR.

4.2.4.3 Results and analysis

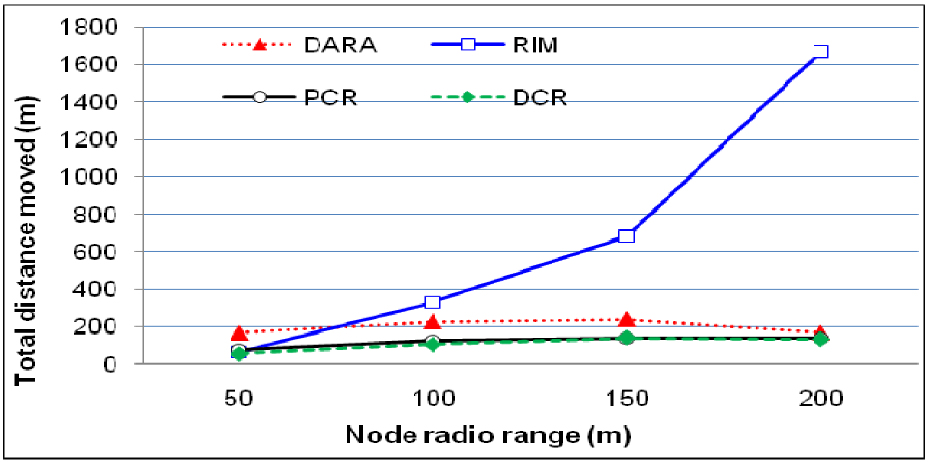
The experiments involve randomly generated topologies with varying actor count and communication range. The number of actors has been set to 20, 40, 60, 80 and 100. The communication range of actors is changed among 50, 100, 150 and 200. When changing the node count, "*r*" is fixed at 100 m and "*N*" is set to 60 while varying the communication range. The results of individual experiments are averaged over

30 trials. All results are subject to 90% confidence interval analysis and stay within 10% of the sample mean.

a. **Total distance moved:** Figure 4.15 shows the total distance traveled by all nodes involved in recovery. DCR significantly outperforms both DARA and RIM because it only moves non-critical nodes to avoid cascaded relocations. Both graphs indicate that the performance advantage of DCR remains almost consistent even with higher node density and transmission range. This is because DCR strives to avoid moving critical nodes that causes further partitioning and requires successive relocations. Furthermore, DCR perform cascaded relocations only



(a)

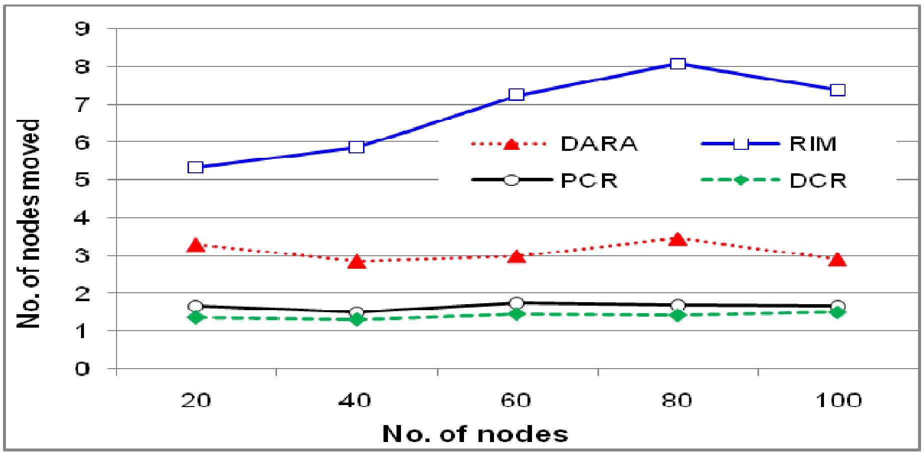


(b)

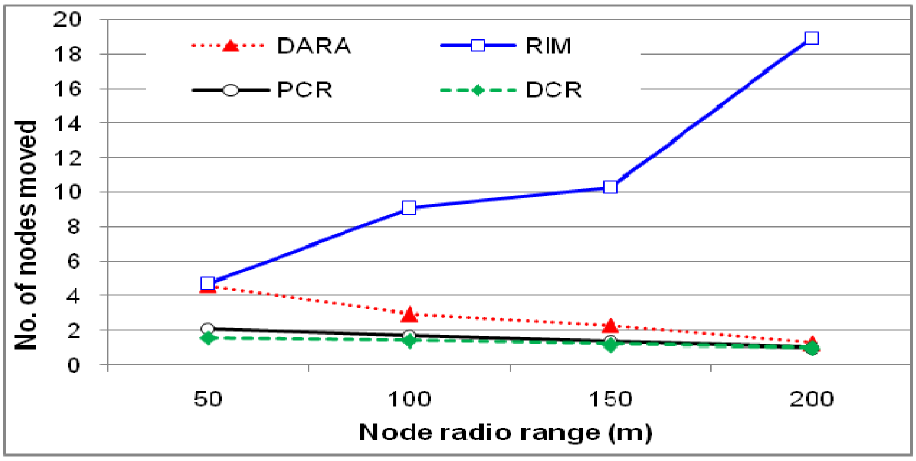
Figure 4.15: The total distance moved by all nodes until connectivity is restored, while varying the network size (a) and radio range (b).

when non-critical nodes in the neighborhood of the failed actor are not available. Figure 4.15-(a) indicates that the performance of DCR scales very well and is not affected by the node density because of choosing non-critical nodes as backup. Similar observations can be made for the communication range (Figure 4.15(b)), where the connectivity-restoration overhead is significantly less as compared to the baseline approaches. The distance movement in DCR is slightly less than PCR because DCR relocates strongly connected nodes that have non-critical nodes in the neighborhood.

b. **Number of moved nodes:** Figure 4.16 shows the number of nodes that were involved in the recovery when DCR and the baseline approaches are applied. The



(a)

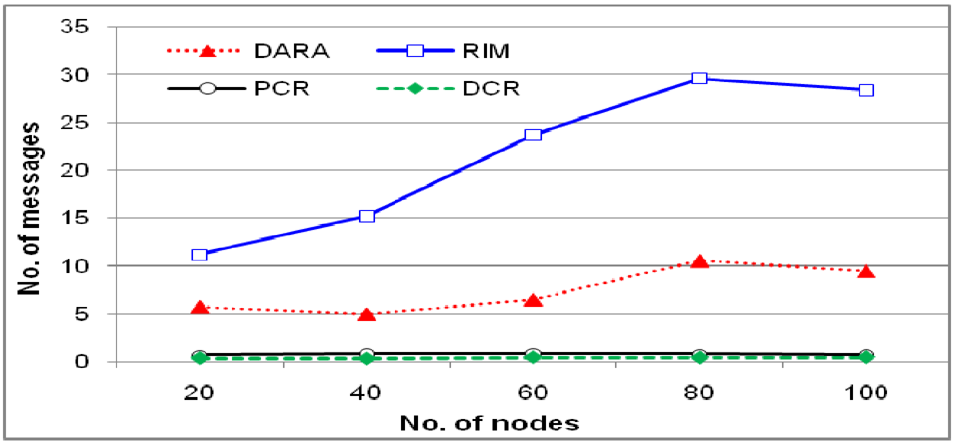


(b)

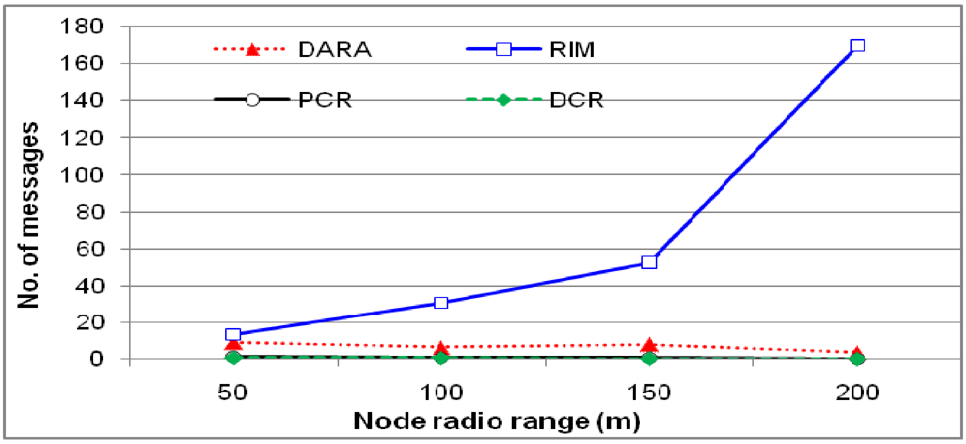
Figure 4.16: The total number of nodes moved during the recovery, while varying the network size (a) and radio range (b).

performance graphs confirm the advantage of DCR which moves fewer actors than RIM and DARA. This is because DCR limits the scope of recovery and avoids successive cascaded relocations by choosing non-critical nodes as backup. Moreover, DCR moves high degree nodes that often have non-critical nodes in the neighborhood. As both the graphs indicate, the performance gap between PCR and DCR is minor relative to DARA and RIM. This is because both PCR and DCR prefer to appoint non-critical nodes as backup. Furthermore, the performance of PCR and DCR remains almost constant while varying the number of nodes and their radio range, which indicates great scalability.

c. **<u>Number of messages exchanged:</u>** Figure 4.17 reports on the messaging overhead as a function of the network size and radio range. As the figure indicates, DCR



(a)

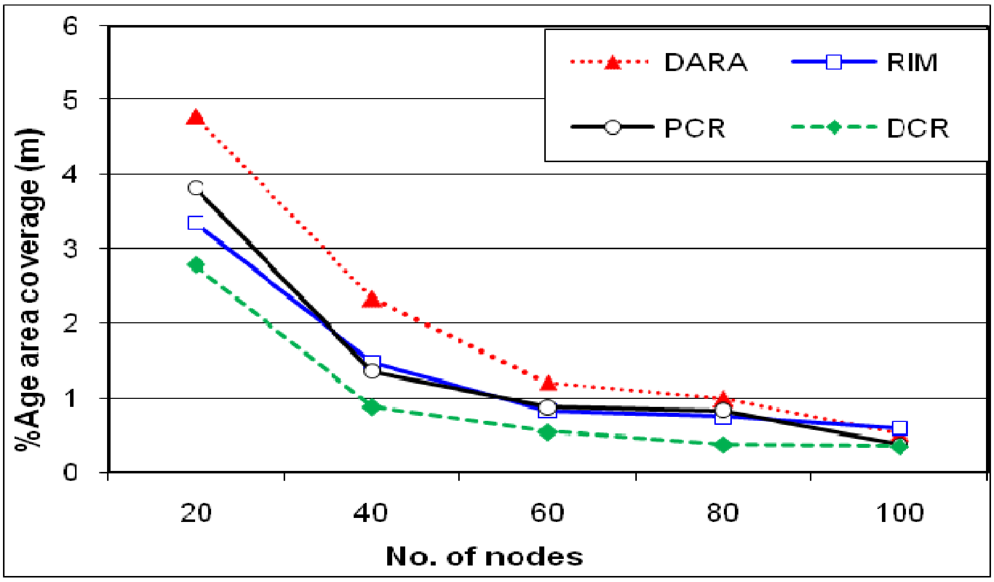


(b)

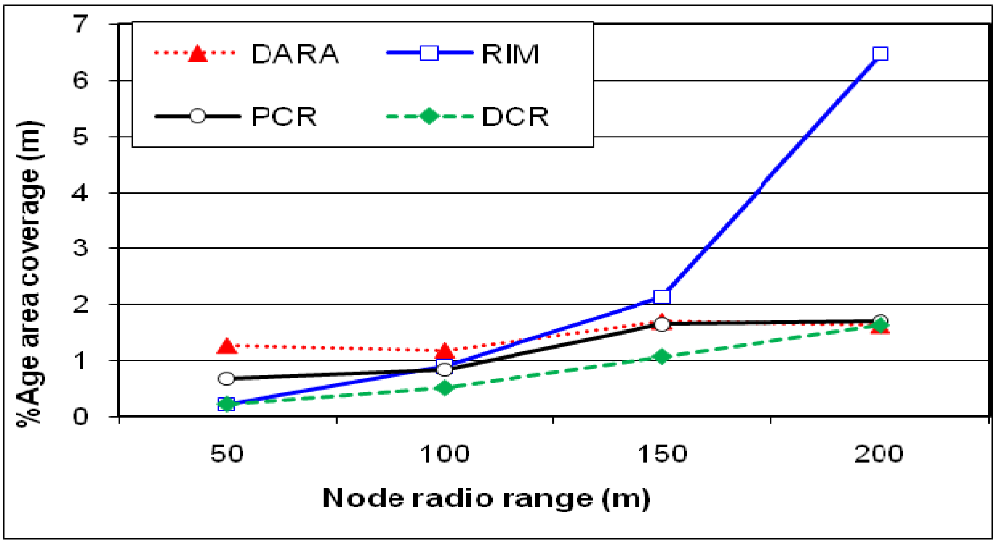
Figure 4.17: The total number of messages exchanged by all nodes during the connectivity restoration, as a function of N in (a) and r in (b).

incurs far less messaging overhead than DARA and RIM. This is because DCR limits message exchange only between a pair of primary and backup nodes instead of all 1-hop and 2-hop neighbors as is the case in RIM and DARA, respectively. Moreover, unlike DARA and RIM, DCR strives to involve non-critical nodes in the recovery which limits the need for cascaded relocation and thus reduces the number of notification messages. Furthermore, DCR limits the scope of recovery by involving high degree nodes that have non-critical nodes in the neighborhood. On the other hand, Figure 4.17 indicates that the messaging overhead in RIM significantly grows for a high actor density and long communication range because the number of recovery participants increases in both cases. Although, the messaging overhead in PCR and DCR is almost similar, DCR mainly outperforms PCR in terms of coverage and connectivity. We will discuss this later in this section.

d. Percentage of coverage reduction: Figure 4.18 shows the impact on coverage, measured in terms of the percentage of coverage reduction relative to the pre-failure level, while changing the *N* and *r*. The action range is set to 50 m in these experiments. Overall, DCR limits the coverage loss and consistently outperforms baseline approaches. Although increasing the node density helps, DARA and RIM still do not make up for the coverage loss and definitely do not match DCR's performance. The advantage of DCR in terms of coverage is obviously due to the limited scope of node relocation, which causes a coverage loss at the network periphery. Moreover, DCR engages strongly connected nodes in recovery that have more overlapping coverage with their neighbors as illustrated above in Section 4.2.3. Hence, moving those actors only reduces the overlapping coverage. Figure 4.18 (b) indicates that the performance of DCR and PCR in terms of coverage reduction is not much affected with increasing the communication range. On the other hand, the performance of RIM significantly worsens when growing the communication range. With the increased value of *r*, the network becomes more connected and the number of neighbors of *F* grows. RIM moves nodes inwards making the area around *F* more crowded while leaving uncovered parts at the network periphery and thus causing a significant loss of coverage.



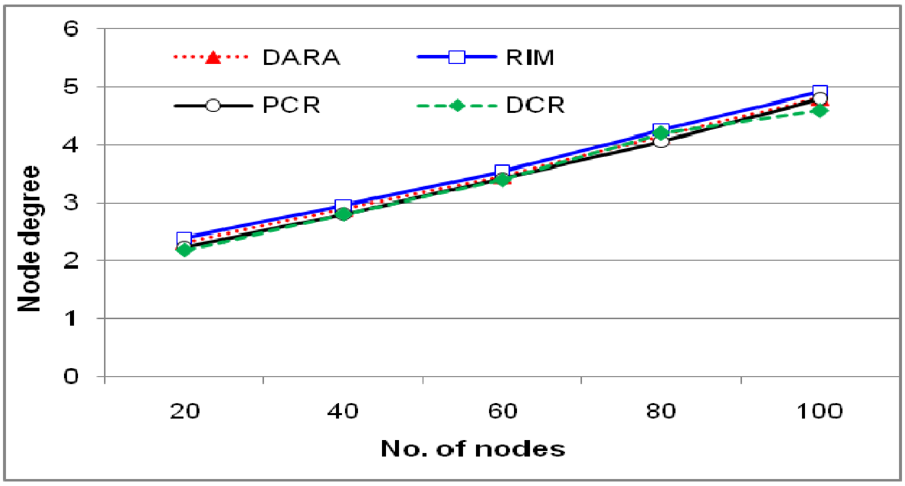
(a)



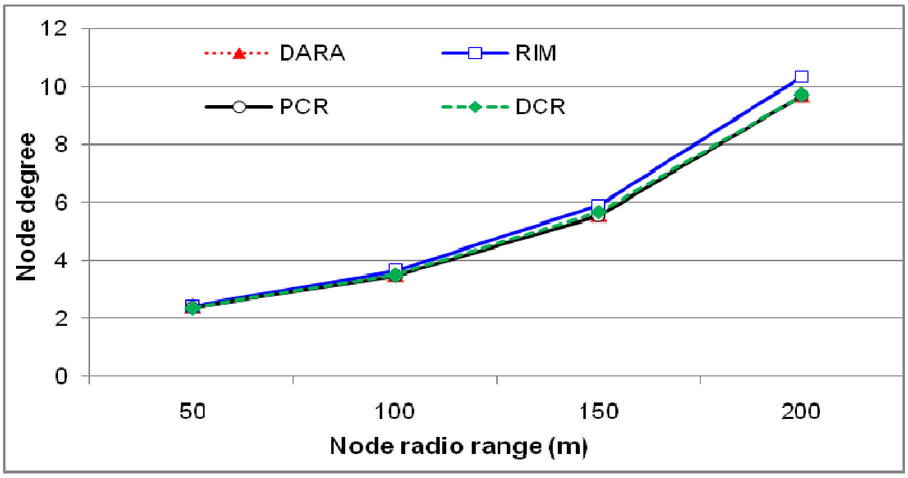
(b)

Figure 4.18: The impact of recovery on coverage area, as a function of *N* in (a) and *r* in (b).

e. <u>Average node degree:</u> Figure 4.19 shows the level of connectivity maintained by all approaches after the recovery is completed. As both figures indicate, DCR consistently maintains the same level of connectivity as of other approaches, despite the fact that DCR is not only factoring connectivity like the other baseline approaches. This is due to moving high degree non-critical nodes and limiting the scope of relocation. Figures 4.15-4.18 confirms that DCR strikes a balance between the various objectives.



(a)



(b)

Figure 4.19: The level of inter-actor connectivity after recovery, as a function of N in (a) and r in (b).

4.3 Recovery from failure of multiple actors

This section presents the application scenario, problem definition and the proposed <u>R</u>ecovery <u>A</u>lgorithm for <u>M</u>ultiple (RAM) node failures.

4.3.1 Application scenario and problem definition

The above mentioned proposed approaches so far assume single critical actor failure at a time and no other node fails during the recovery process. In other words, they handle sequential actor failures. Although the probability for multiple node failure is small, the hazardous application environment or exhaustion of onboard energy may cause the failure of more than one adjacent actor. In general, the recovery from simultaneous failure of multiple nodes is very challenging. DCR and other recovery schemes for a single node failure are not guaranteed to converge. For DCR, simple two non-critical backup nodes may move causing a network partitioning in other parts of the networks. Consider, for example, the topology of Figure 4.20. When node P_i and P_j fail, moving their backup nodes B_i and B_j will cause the network to partition although neither of these backups are critical nodes.

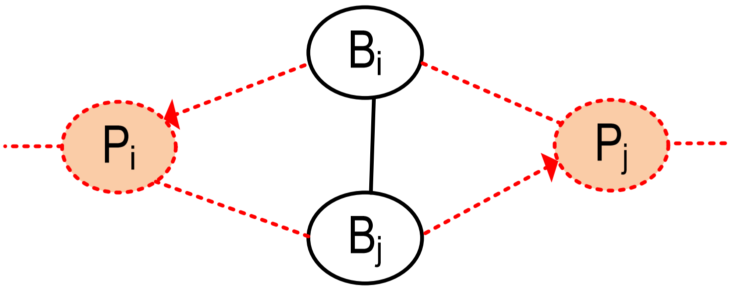


Figure 4.20: Illustrating the challenges in handling multiple simultaneous failures, where moving two non-critical partitions the network.

4.3.2 RAM overview

To handle only a special class of multi-node failure which we believe is more likely to happen. In scenarios in which the failure is caused by external factors such as explosions, multiple nodes may get damaged. For this scenario, we propose a novel <u>R</u>ecovery <u>A</u>lgorithm for <u>M</u>ultiple node failures (RAM). Like DCR, RAM is also a hybrid approach but the criteria for backup selection and recovery are different. RAM identifies critical actors using the same procedure explained earlier. Once the critical actors are identified, they choose appropriate backups that will handle their failure. The details of the backup selection and recovery procedure are described in the following of this section.

4.3.2.1 Backup selection

Once the critical actors (primary) are identified, they appoint appropriate backups to handle their failure. Like DCR, RAM also maintains minimum network state information (i.e. 1-hop) to avoid extra messaging overhead. However, RAM imposes additional constraints while choosing a backup in order to ensure convergence and avoid the creation of another network partitioning. The selection of a backup among 1-hop neighbors is based on the following ordered criteria:

Neighbor Criticality and Availability (NCA): Each actor maintains a state whether it is already engaged (as a backup) by some node or not beside its position in the network topology (i.e. critical/non-critical). Unlike DCR, RAM put the following constraints while choosing a backup based on the position and state of a node, i.e., candidate backup:

a. When a critical actor chooses a backup, it prefers a non-critical node that is not serving another primary. In other words, a non-critical node cannot have more than one primary as long as another free non-critical node is available in the neighborhood. A critical node is restrained from choosing a non-critical node as backup that is already designated for another adjacent actor. This is to ensure recovery in case two adjacent actors fail simultaneously.
b. A critical node can be chosen as a backup only if it is not already appointed by some other node. Moreover, two adjacent critical actors cannot serve each other as backup simultaneously. This will ensure that there will be some backup node to recover incase adjacent actors fail at the same time.
c. If a critical actor (i.e. primary) "*A*" picks a non-critical neighbor "*B*" as a backup, RAM requires "*B*" to also pick a backup "*C*" among its neighbors using the same criteria mentioned above. However, node "*B*" status is not changed to critical.

This condition enables recovery when the primary and backup both fail at the same time. In addition, it prevents the scenario of Figure 4.20. The other criteria are similar to DCR.

4.3.2.2 Failure detection and recovery

Like DCR, actors periodically exchange heartbeat messages with neighbors in order to update their status. The backup actor determines the failure of the primary through missing successive heartbeats. Once the failure is detected, the backup node (s) initiates a recovery process that depends on the NCA. Since our backup selection criteria strive to ensure that critical actors have distinct backups, the recovery procedure is executed concurrently on the various backups. If the backup (s) are non-critical nodes then they simply replace the corresponding primary and the recovery is complete. For example, failure of adjacent primary actors P_i and P_j is detected by their designated backups B_i and B_j, respectively. Both B_i and B_j will execute recovery concurrently. Figure 4.21(a) demonstrates that replacing non-critical B_i and B_j restores the connectivity lost due to failure of P_i and P_j and does not require cascaded

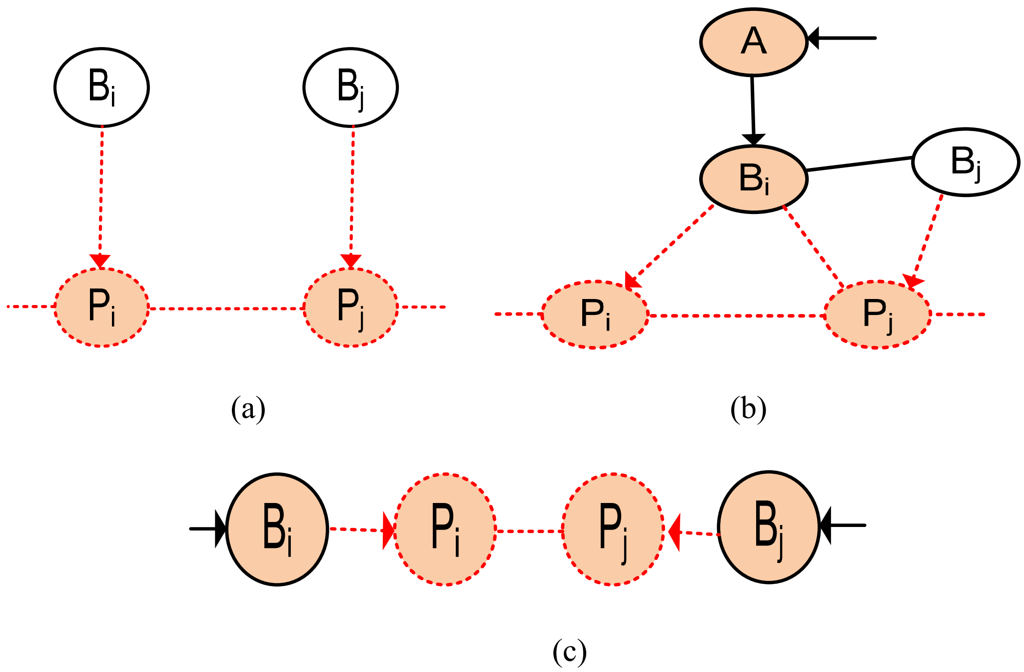


Figure 4.21: The recovery process when there is no risk in repartitioning the network.

relocations. On the other hand, if the backup is a critical actor, moving that node will further trigger cascaded relocations until a non-critical node replaces the primary. For instance, Figure 4.21(b) demonstrates the scenario where a critical backup B_i sends a movement notification message to its own backup and moves to the place of failed primary P_i. Moving critical node B_i initiates a shifted relocation (Li et al. 2007) where each backup replaces the primary. Whereas, a non-critical backup B_j simply moves to the location of primary P_j and the recovery is complete. Figure 4.21 (c) shows the recovery process when both the backup (s) B_i and B_j are critical nodes. They will send a message to their backups and replace their primary P_i and P_j, respectively.

Failure of adjacent actors: The presented recovery process of RAM will successfully restore the connectivity except for the following case for which the topology may not get repaired. A critical actor A_i may choose an adjacent critical node A_j as backup, while A_j designates another node A_k as a backup and A_k happens to be a neighbor of A_i. RAM can partially recover from failure of adjacent critical actors since one of the designated backups has also failed and none of the other nodes is responsible for the recovery. An example is depicted in Figure 4.22(a). Actor *A2* was a backup of *A1* and *A13* was a backup of *A2*. In the case of *A1* and *A2* failing, although, the backup *A13* detects the failure of *A2* and executes the recovery procedure described earlier, none of the surviving nodes is responsible to recover from the failure of *A1*. Figure 4.22(b) clearly indicates that RAM cannot restore connectivity despite the failure of *A1* being detected by its neighbors. The obvious reason is that RAM designates a backup to each critical actor that is only responsible to replace its primary.

To handle this case, we introduce a variant of RAM's recovery procedure that imposes a slightly extra recovery overhead. The idea is to let the backup know about the grand primary as well (i.e. primary of the primary). In the case of failure of adjacent critical actors, the designated backup coordinates the recovery. For instance, in Figure 4.22(a), *A2* makes *A13* aware that it is a backup for *A1* before failure. In case of failure of *A1* and *A2*, *A13* will replace *A2* and find that *A1* is also lost as shown in Figure 4.22(b). *A13* appoints a new backup i.e. *A9*, sends a notification message and moves to replace *A1*. The newly appointed backup follows the primary and cascaded relocations are performed as shown in Figure 4.22(c). This special case can be generalized to a ring of critical nodes in which A_2 serves as a backup to A_1, A_3

is backup of A_2, …, A_n is a backup of A_{n-1}, and, A_n is a neighbor of A_1. Since nodes A_1, A_2, …, and A_{n-1} are critical, if they fail, A_n needs to replace A_{n-1}.

RAM will simply recursively send the primary-backup relationship of a series of reachable critical nodes on the ring. This can be achieved simply by making a primary *C* to inform its backup node *B* about whether *C* also serves as a backup for another primary *A*. If *B* has a link to *A*, *B* will apply this procedure. Otherwise if B is a critical node, *A* will keep on informing its backup about *B* and *C* and so on.

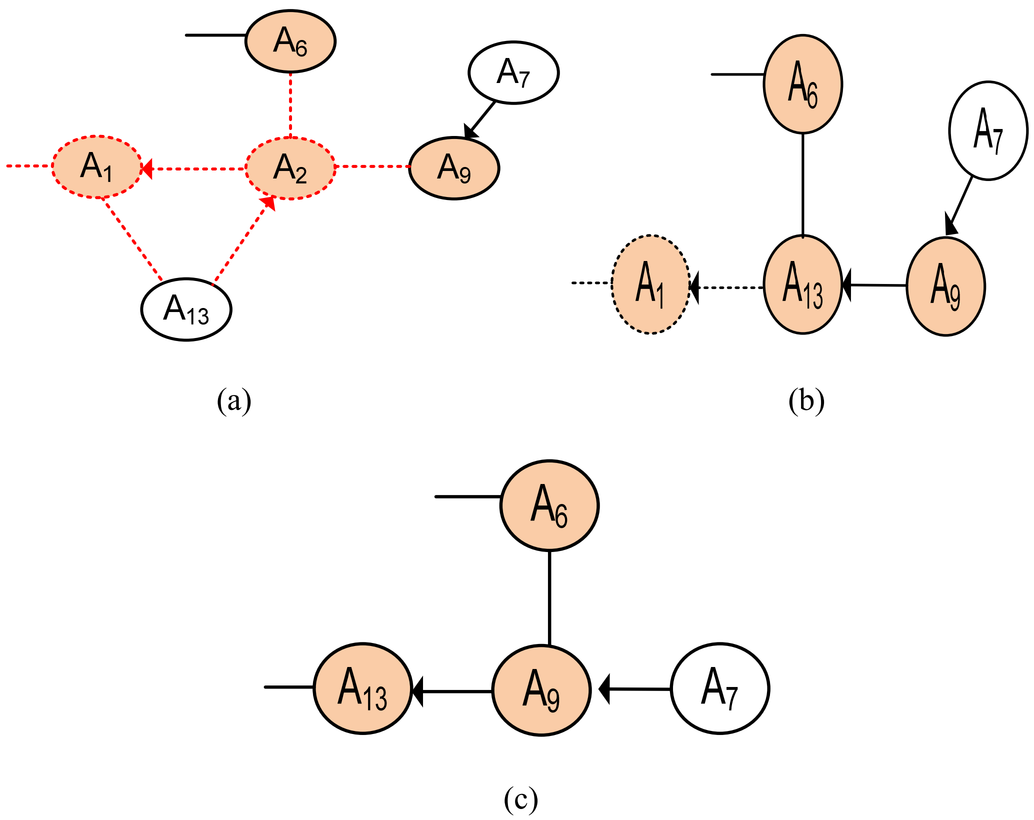
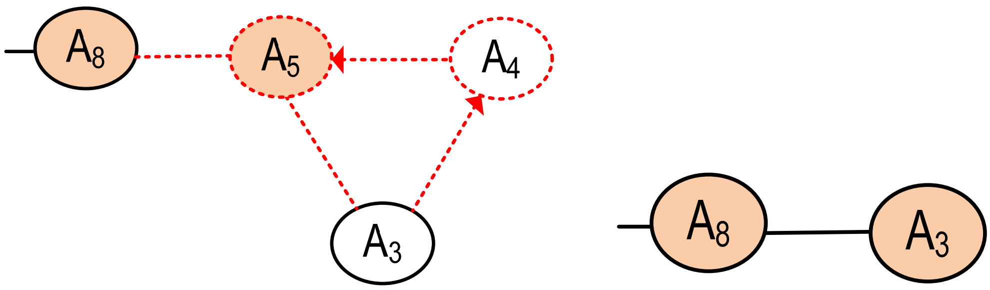


Figure 4.22: Special case of failure detection and recovery; (a) *A13* detects failure of *A1* and *A2* (b) *A13* replaces *A2* and appoints *A9* as backup (c) *A13* moves to the place of *A1*, *A9* and *A7* follow it.

Figure 4.23(a) illustrates a slightly different scenario. A_3 detects the failure of non-critical primary A_4 and finds that A_5 (grand primary) has also failed. A_3 ignores the failure of A_4 (since it is non-critical) and moves to the position of A_5 as shown in Figure 4.23(b).



(a) (b)

Figure 4.23: Special case of failure detection and recovery; (a) *A3* detects failure of critical actor A_5 and non-critical A_4 (b) *A3* directly replaces *A5* and ignores *A4*.

4.3.3 RAM analysis

In this subsection, we analyze the performance of the RAM algorithm in successfully restoring the connectivity of the network lost due to simultaneous failure of multiple actors. We introduce the following theorems:

Theorem 4.6: *RAM successfully rejuvenates the connectivity broken due to failure of adjacent actors.*

<u>Proof:</u> In essence, RAM carefully picks the backups so that DCR can be used without repartitioning the network, and thus the convergence of RAM is implicitly guaranteed by Theorem 4.1 if the backup selection process is shown to make the recovery from the failure of *m* actors as *m* or fewer independent invocations of DCR. RAM strives to designate non-critical (either intermediate or leaf) nodes as backups. RAM assigns distinct backups for adjacent critical actors and prevents two actors from mutually serving as backup for each other. Therefore, DCR can be applied to each failed actor independently. In addition, RAM requires a node to inform its backup about its own primary if any. This will allow DCR to select the right position to move to in case a critical actor A_i chooses an adjacent critical node A_j as backup, while A_j designates another node A_k as a backup and A_k happens to be a neighbor of A_i. For this case, A_k will replace A_i when both A_i and A_j fail. Doing that for a sequence of primary-backup

critical nodes can be viewed as a multi-application of DCR enabled by the increased portion of the network state that the primaries and backups share.

Theorem 4.7: *The time it takes RAM to converge while reestablishing the inter-actor connectivity is O(r) where r is the radio range of actors.*

Proof: As stated in Theorem 4.6, RAM appoints distinct backups for adjacent actors and then applies the DCR recovery procedure for each failed node independently. In DCR, the maximum time it takes the backup to replace the failed primary is proportional to *r*, as proved in Theorem 4.2. If the number of failed adjacent primary actors is *f* and moving a critical backup further triggers *c* relocations for each failed primary, the total recovery time will be proportional to $(f(c) + f) *r$. The relocations will be sequential for each failed node but in parallel for the failed primary (s). Thus, RAM convergence time to restore connectivity in the worst case is $(f (c) +f) *r$ which is $O(r)$.

Theorem 4.8: *The recovery process of RAM does not introduce additional cut-vertices in the repaired network topology.*

Proof: RAM strives to restore connectivity through multiple and independent invocations of DCR. Furthermore, when a series of critical nodes are engaged in a primary-backup relationship as a part of a ring, RAM handles this as an optimized implementation of a multiple-application of DCR and would not thus introduce a cut-vertex. Thus, based on Theorem 4.3, it can be concluded that RAM does not introduce new critical actors during the recovery process.

Theorem 4.9: *The recovery process of RAM incurs messaging overhead of* $O(N^2)$ *where N is the number of actors in WSAN.*

Proof: Like DCR, RAM also maintains 1-hop neighbor information to restore inter-actor connectivity; this requires 1 message for each actor. In addition, as a node *B* that is picked as a backup by node *A*, *B* will need to inform its own backup *C* about *A*. In the worst case when the topology is a ring, this will involve *N* more messages per node, i.e., total of N^2. Furthermore, every critical actor involved in recovery has to send 1 movement notification message to its corresponding backup. If the number of adjacent primary actors failing is *f*, RAM moves each critical node only once. Thus, in the worst case, when the entire critical nodes $(C-f)$ move, the total number of messages will be $(N^2+N+C\text{-}f)$. Therefore, the messaging complexity overhead in RAM is $O(N^2)$.

4.3.4 RAM pseudo code

Figure 4.24 shows the pseudo code of RAM that each actor "*B*" would execute. The pre-failure steps resemble DCR. During the network bootstrapping phase, each actor (either critical or engaged as a backup) will appoint an appropriate backup among neighbor actors using the AppointDistinctBackup() procedure (lines 1-3). If actor *B*

```
RAM (B)
1   if (IsCritical(B) == true || B.State == Engaged) then
2      AppointDistinctBackup (B)
3   end if
4   if (B detects the failure of primary actor F or receives a
    Movement notification message from F) then
5      FailureRecovery (B, F)
6   end if
AppointBackUp (B)
7   //Appoint a distinct backup for actor B based on the
     Criteria mentioned in Section-IV(C).
FailureRecovery(B, F)
8   if (IsCritical (B) == true) then
9     NotifyBackUp(B)
10  end if
11  if (IsCritical (F)) then
12     MoveToLocation(B, F)
13  end if
14  if (B detects failure of its primary's primary G) then
15     FailureRecovery (B, G)
16  end if
```

Figure 4.24: Pseudo code of RAM for backup selection and failure recovery.

either detects the failure of primary *F* or receives a movement notification message from *F*, node *B* triggers a recovery procedure FailureRecovery() to recover from *F* (lines 4-6).

The AppointDistinctBackup() procedure is slightly different from its counterpart "AssignBackup" in DCR. The AppointDistinctBackup() procedure ensures that the picked backup node does not serve another primary and bases the selection on the criteria mentioned in Section 4.3.2.1 (line 7). The procedure FailureRecovery() is also different from the "Recovery" in DCR since in RAM two adjacent actors are not allowed to choose each other as backup at the same time. If the backup *B* is a critical actor, it notifies its backup so that the connectivity can be maintained (lines 8-10). Since backup *B* is aware of the status of the failed primary *F*, it checks whether the failed primary was critical or not. If the failed node *F* was critical *B* moves to replace *F* (lines 11-13). Otherwise, there is no need to replace it since it was non-critical. In other words, *B* will directly move to the location of grand primary *G* as shown in Figure 4.23 and which will be discussed in the following lines.

If the backup node *B* also detects the failure of its grand primary *G* (i.e. primary of the primary), then *B* executes the recovery procedure FailureRecovery() to recover from grand primary as mentioned in Figure 4.22 earlier (lines 14-16).

4.3.5 Performance evaluation

The performance of RAM is validated through extensive simulations. This section describes the simulation setup, performance metrics, results and analysis.

4.3.5.1 Simulation setup and performance metrics

In the experiments, we have created inter-actor topologies that consist of a varying number of nodes (20-100). Nodes are randomly placed in an area of 1000 m × 600 m. We have varied the transmission range of the actors (50-125) so that the topology becomes strongly connected. The performance is assessed using the metrics described in Table 4.3.

Table 4.3: Performance metrics for RAM algorithm

Performance Metrics	Description
Total distance moved	The *total distance moved* by all nodes involved in the recovery: This gauges the efficiency of RAM in terms of energy efficiency and overhead involved in the recovery.
Number of nodes	The *number of nodes* moved during the recovery: This metric reflects the scope of the recovery process.
Number of messages	The *number of messages exchanged* among nodes: Again this metric indicates the energy dissipation and recovery overhead.
Impact of recovery on coverage	The *Total coverage* relative to the actual pre-failure coverage: Although connectivity is the main objective of RAM, actor coverage is important for many setups. The loss of a node usually has a negative impact on coverage. This metric assesses the effectiveness of RAM in terms of coverage reduction.

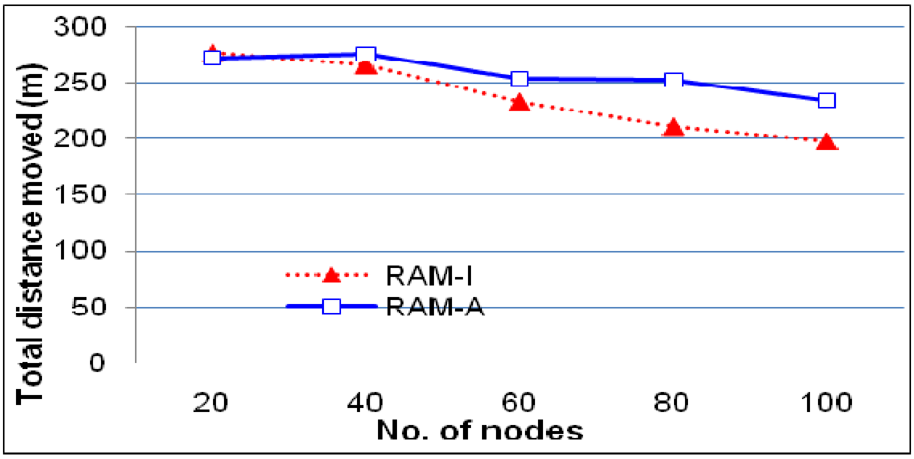
The following parameters were used to vary the WSAN configuration in the experiments:

- The *number of deployed nodes (N)* in the network affects the node density and the inter-actor connectivity.
- The *node communication range (r)* influences the network connectivity and highly affects the recovery overhead in terms of distance traveled and the number of actors involved.

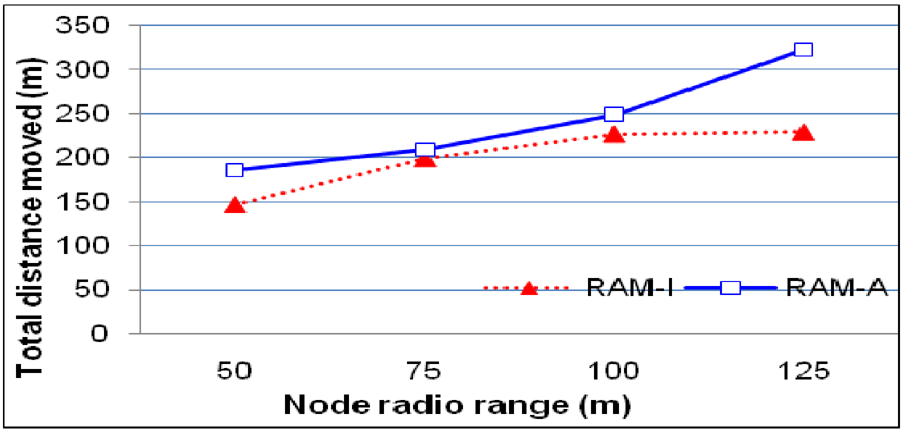
4.3.5.2 Results and Analysis

We use the same experiment setup of DCR to evaluate the performance of RAM. We use a similar procedure of DCR to identify critical actors and choose two adjacent cut-vertices at random to be failed simultaneously. For RAM-I, the failed nodes have backup independent of each other, and thus it is like running DCR twice. As we have seen in the previous section, DCR significantly outperforms contemporary schemes found in literature; therefore, the validation of RAM is based on DCR. The RAM-A curve reflects the results when one of the designated backups also fails along with its adjacent primary as illustrated above in failure of adjacent actors. For RAM-I and RAM-A, we have performed experiments with 15 different topologies. The goal of comparing the performance of RAM-I and RAM-A is to capture the effect of failure scenarios, for which a node *C* has to deal with the failure of its primary *B* as well as node *A* that *B* serves as a backup.

a. **Total distance moved:** Figure 4.25 shows the total distance moved by all the nodes involved in the recovery. Both the graphs shows that RAM-A moves a slight distance more than RAM-I. This is due to engaging additional nodes to recover from failure of an adjacent node. Moreover, RAM-I has independent pre-designated backups for the failed actors that do not have to travel an additional distance to recover from failure of the grand primary. Figure 4.25(a) indicates that the performance of both the algorithms improves with the increased actor density. Increasing the number of actors boosts the level of connectivity and consequently boosts the number of non-critical nodes. The availability of non-critical nodes reduces the scope of cascaded relocations. On the other hand, Figure 4.25(b) shows that the travelled distance grows with the increase in transmission range. This is because nodes have to travel longer distances in order to restore connectivity.



(a)



(b)

Figure 4.25: The effect of changing *N* (a) and *r* (b) on the total distance travelled by all nodes during the recovery.

b. <u>Number of moved nodes:</u> Figure 4.26 reports on the number of nodes that get involved in the recovery. Again, both the performance graphs indicate that RAM-I marginally performs better than RAM-A in terms of the scope of recovery. This is because independent execution of recovery in RAM offers more non-critical nodes on the recovery paths that prevent unnecessary cascaded relocations. The performance of both variants of RAM improves with the increased actor density and higher transmission range due to the increased degree of connectivity and the availability of non-critical nodes in the neighborhood. This limits the scope of recovery.

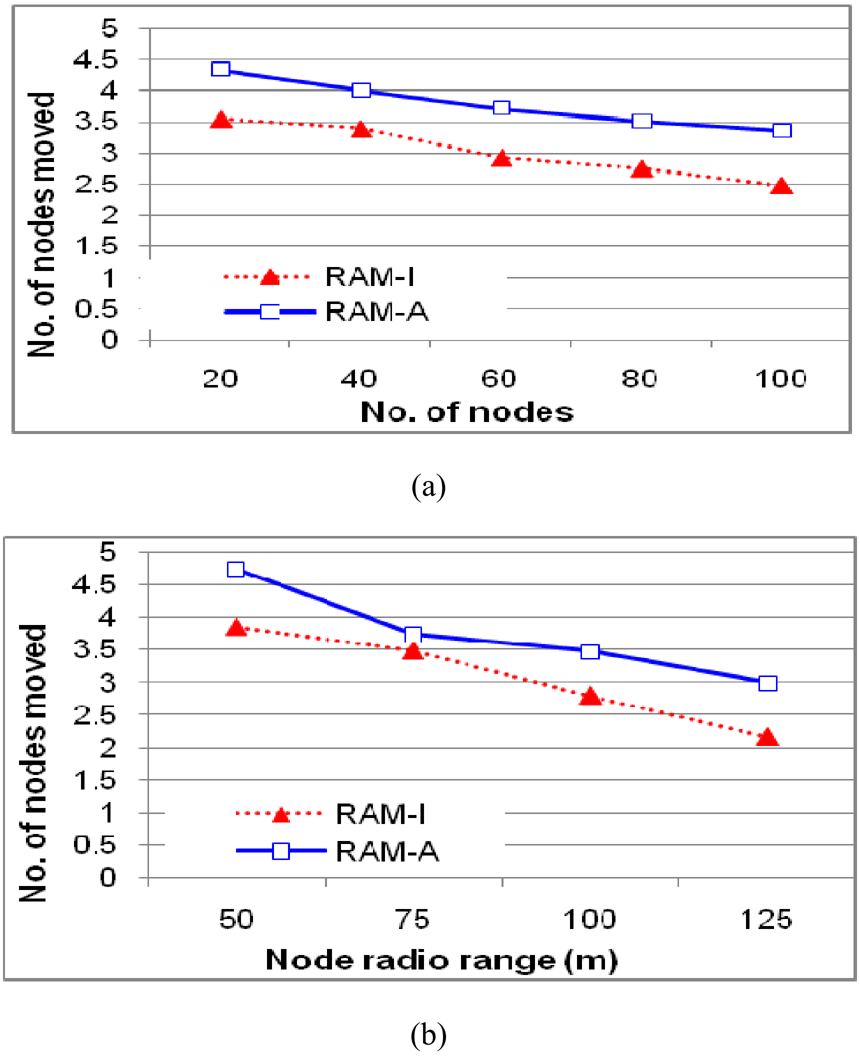
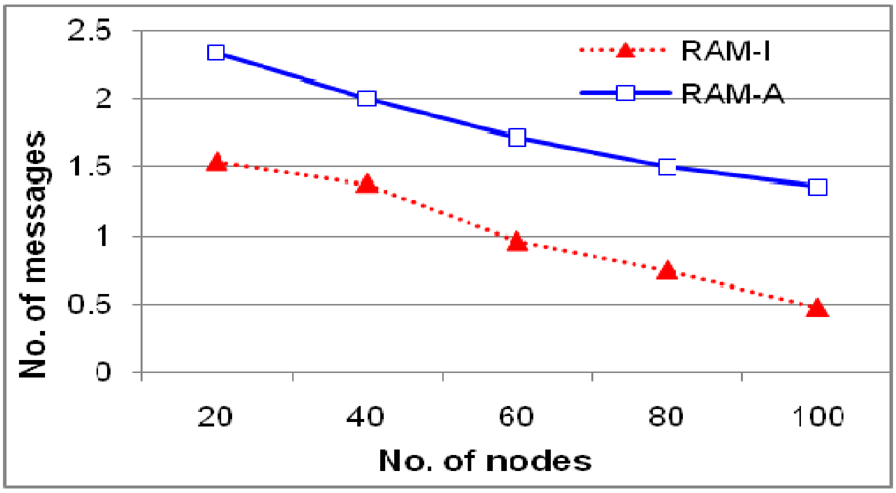
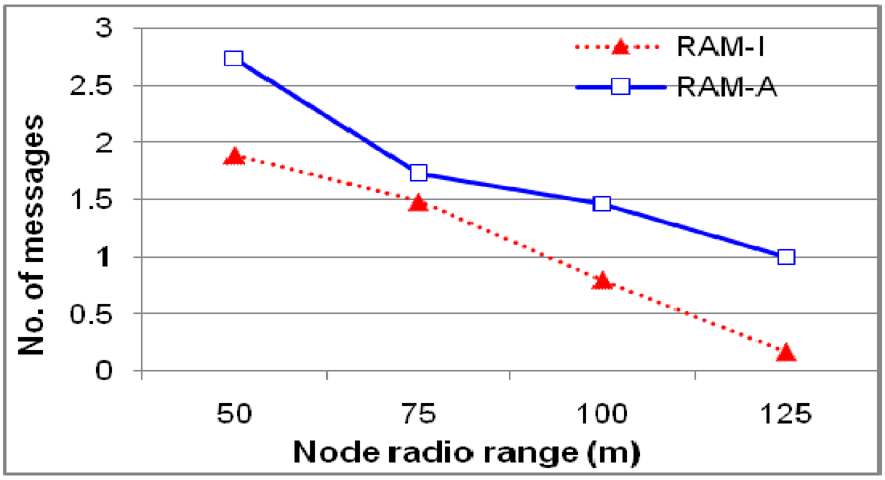


Figure 4.26: The total number of nodes moved to restore connectivity, as a function of N in (a) and r in (b).

c. **Number of messages exchanged:** Figure 4.27 shows the messaging overhead as a function of the network size and radio range. As expected, RAM-A incurs slightly more messaging overhead than RAM-I. The obvious reason is because RAM-A sends extra recovery coordination messages. Both the figures suggest that messaging overhead reduces with the higher density and radio range. This is because network connectivity improves in both cases. This ensures the availability of more non-critical nodes that do not require sending movement notification messages.



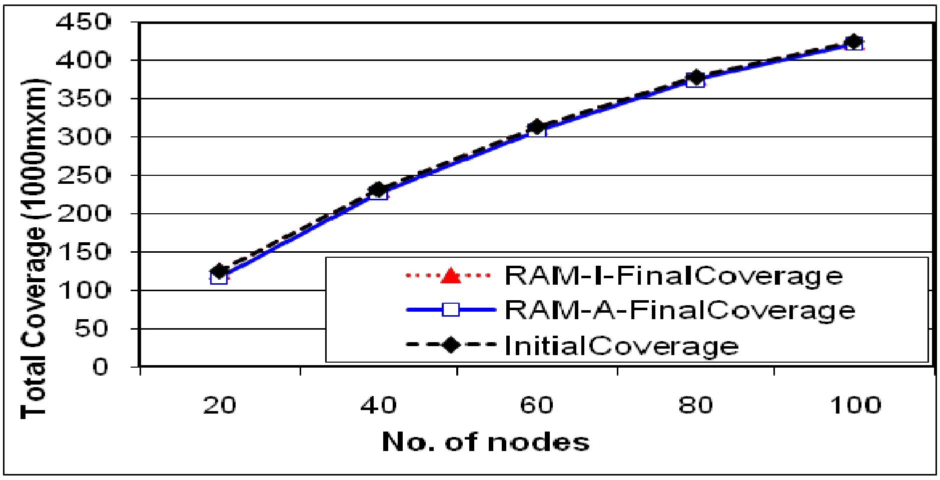
(a)



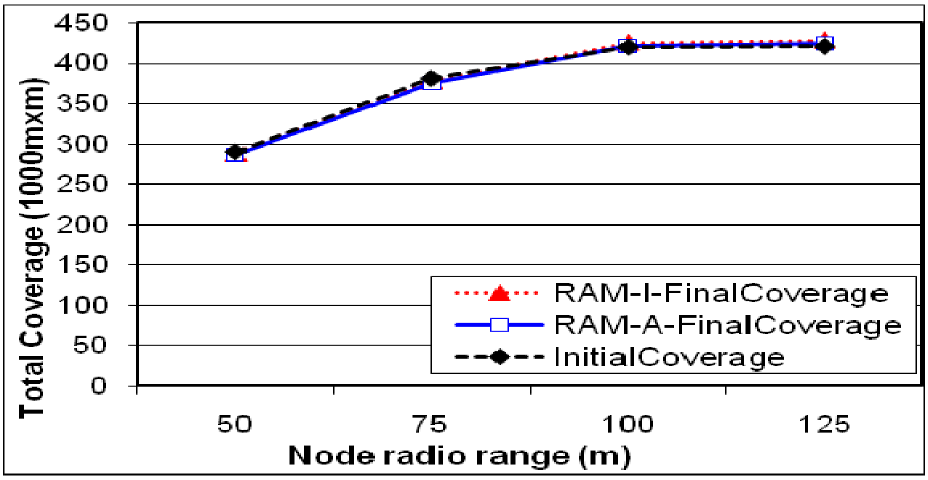
(b)

Figure 4.27: The total number of messages exchanged during recovery, while varying the node density (a) and radio range (b).

d. **Impact of recovery on coverage:** Figure 4.28 shows the impact of node relocations on coverage relative to the pre-failure level, while changing the *N* and *r*. Both the performance graphs clearly indicate that the affect of node relocations on coverage during connectivity restoration is nominal. Obviously, this is due to limiting the scope of cascaded relocations by involving non-critical nodes during recovery. Moreover, moving non-critical actors with overlapping coverage further preserves the loss of coverage.



(a)



(b)

Figure 4.28: Impact of node relocations on the coverage during connectivity restoration, as a function of *N* in (a) and *r* in (b).

4.4 Chapter summary

In this chapter, we have investigated the problem of restoring inter-actor connectivity lost due to failure of one or multiple actors. We have proposed PCR; a distributed algorithm for Partitioning detection and Connectivity Restoration. Unlike reactive approaches published in literature which does not assess the impact of actor failure on coverage and connectivity, PCR pursues a pre-failure planning to avoid overreacting and, thereby increase the efficiency of the recovery. PCR uses a localized scheme to identify critical actors and designate backups for them. The backup actor detects the failure of the primary and pursues node relocation to repair the partitioned network topology. We have presented an optimization to the backup selection procedure of PCR; a partitioning Detection and Connectivity Restoration (DCR) algorithm. Unlike PCR, DCR prefers to choose high degree actors in order to minimize the recovery overhead and the impact on coverage and connectivity.

DCR assumes a single critical actor failure at a time and no other node fails during the recovery process. In order to handle a special case of simultaneous multi-actor failure, we have proposed RAM. RAM handles failure scenarios in which two adjacent nodes simultaneously fail. Like DCR, RAM is also a distributed hybrid approach that identifies critical actors and assigns for them backups. However, RAM assigns distinct backups for each critical actor. The designated backups detect failure of their primaries and move to replace them. In addition, RAM extends the primary-backup relationship in some cases in order for the recovery to converge when a primary and its backup fail at the same time and when the relocation of two backups causes the network to partition. The performance of the proposed approaches has been analyzed and validated through simulation. The simulation results have confirmed the effectiveness of the proposed approaches in terms of messaging and movement overhead while minimizing the scope of recovery and the impact on coverage.

Most of the existing connectivity restoration algorithms do not consider application-level constraints on mobility of actors while recovering from node failure. In the next chapter, we present an Application-centric Connectivity Restoration (ACR) algorithm that factor in application-level interests while recovering from an actor failure.

CHAPTER 5

APPLICATION-CENTRIC CONNECTIVITY RESTORATION ALGORITHM FOR WSANS

As stated in Section 2.2.3, most of the existing movement control connectivity restoration approaches do not consider application-level constraints on mobility of actors while recovering from a node failure. This chapter presents a novel hybrid Application-centric Connectivity Restoration (ACR) algorithm. The main idea is to identify critical nodes and appoint appropriate backup for them, preferably among the non-critical nodes. A backup is picked among the 1-hop neighbors that cause a minimum disturbance to the application and has the least impact on coverage and connectivity. Each pre-assigned backup starts monitoring its primary and then initiates a recovery process once the failure of the primary is detected. Simulation results validate the performance of ACR compared to contemporary recovery schemes.

The rest of the chapter is organized as follows. Section 5.1 details the system model, application scenario and problem definition. The design of the proposed ACR algorithm is presented in Section 5.2. Section 5.3 details the performance evaluation and results. The summary of all the approaches is presented in Section 5.4. The chapter is finally concluded in Section 5.5.

5.1 System model, application scenario and problem definition

We have adopted the same system model described in Section 3.1. As mentioned earlier, actors need to collaborate with each other on planning an optimal coordinated response and synchronize their operations. For instance, in disaster management applications, sensors and actors are placed in an area that got hit by a natural disaster such as fire, earthquake, blizzard, etc. The sensors detect the presence of survivors in the vicinity and report it to the responsible actors for that area. The actors equipped with the necessary equipment receive the sensors data, process it and share it with peer actors to determine the most suitable set of actors that will participate in the operation. The designated actors are responsible for approaching and rescuing survivors, extinguishing fires, lifting rubble, etc. Therefore, actors should collaboratively identify the most appropriate set of actors that will conduct the operation. This requires that actors should be able to frequently update each other's state e.g. actor position, capabilities, criticality of the current task in execution, responsibilities, etc. Therefore, actors establish and maintain inter-actor topology in order to enable such communication. Since each actor is equipped with limited resources and capabilities for executing a critical task, the role of an actor is extremely crucial for a timely response to detected events and to prevent serious consequences.

Nonetheless, the harsh environment that WSAN operates in makes actors susceptible to physical damage and component malfunction. Failure of a critical actor, i.e., a cut-vertex node, may split the inter-actor network into disjointed segments while leaving an uncovered region. Consequently, an inter-actor interaction may cease and the network may become incapable of delivering a timely response to a serious event that may cause a major failure to the application. Therefore, a timely recovery from an actor failure is of the utmost importance. Since WSANs operate autonomously in unattended setups, replacing the failed actor is often infeasible and the recovery should be a self-healing and agile process that involves reconfiguring the inter-actor topology. Moving an incompetent actor or an actor executing a critical task to replace the failed one may cause a major failure at the application-level. In other words, compromising on the application-level interest to achieve a minimum recovery overhead may not be practical in many scenarios. Moreover, the criticality of the applications and the resource constrained nature of networks necessitate an application-centric recovery scheme with a low restoration time and reduced overhead.

An actor failure may cause degraded task execution, a drop in coverage and severed connectivity. Actor capabilities and its current task may determine the significance of an actor from an application and coverage perspective. Similarly, the position of an actor significantly affects the inter-actor connectivity. For example, losing a leaf/non-critical node, such as A_5 in Figure 5.1, does not affect inter-actor connectivity. Meanwhile, the failure of a critical actor such as A_2 partitions the network into disjointed segments. ACR pursues actor relocation to recover from critical node failures. We consider one failure at a time and assume that no node fails during the recovery of another.

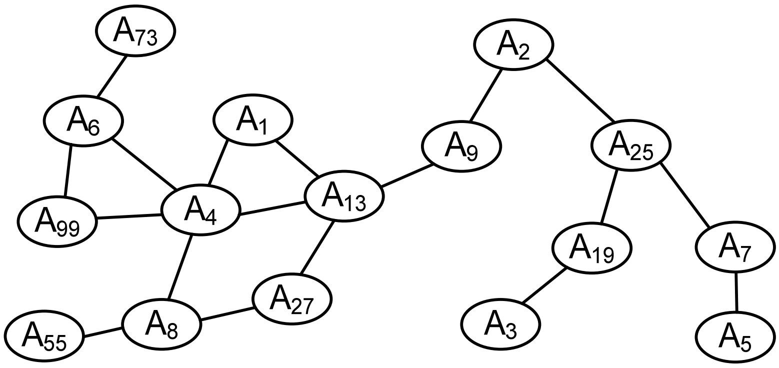


Figure 5.1: A connected inter-actor topology showing a mix of critical/non-critical actors.

We associate two application-level parameters to each actor i.e. Actor Capabilities (AC) and Task Criticality Index (TCI). Each actor would maintain the value of AC in the range [0-1]. The value of AC determines the application aspect, i.e., what an actor is expected to do. The lower bound 0 is interpreted as the actor's inability to respond to an event, whereas, 1 means an actor can fully respond to an event in the area covered by the actor. Moreover, each actor would maintain TCI that refers to the priority of the current task being executed by the actor. The value of TCI ranges from 0 to 1, where 1 means the actor is executing an extremely important task. A noticeable point is that AC has a higher priority than TCI since it reflects an application-level, multi-task-based, aspect. In addition to these two values, actors periodically exchange ID, location and degree with their 1-hop neighbors.

5.2 Design of application-centric connectivity restoration algorithm

As stated in the previous section, moving an incompetent actor and/or an actor executing a crucial task may cause major failure at the application-level. Moreover, criticality of applications and the resource-constrained nature of WSANs necessitate a minimum recovery time and overhead. Unlike contemporary schemes found in the literature, the proposed ACR algorithm factors in application level concerns besides minimizing recovery time and overhead while repairing the damaged network topology. Like PCR, ACR determines the critical actors (primary) and designates for them backup nodes as part of pre-failure planning. Each critical actor handpicks a suitable backup that can satisfy application level constraints. While choosing a backup, a primary actor strives to find a nearby non-critical backup node in order to limit the scope of recovery and reduce the overhead.

Moreover, ACR strives to minimize the affect of actor failure on coverage and connectivity by engaging strongly connected nodes with overlapping coverage. The pre-assigned backup pursues controlled and coordinated motion to reach the position of a failed primary. Since moving a critical backup actor may further break the inter-connectivity, ACR is recursively applied until all actors become connected. To the best of our knowledge, this is the first hybrid algorithm that considers application-level interests while reducing the recovery overhead besides minimizing the impact of recovery on coverage and connectivity. The detailed algorithm is described in the following subsections.

5.2.1 Determining cut-vertex (critical) actors

As discussed in the previous chapter, the failure of a critical actor divides the inter-actor network into disjointed segments in addition to leaving a coverage hole. Therefore, ACR determines critical nodes as part of pre-failure planning and designates for them backup actors to tolerate node failures. A critical node in this context acts as a cut-vertex in the network topology, and when it fails, it causes the network to partition into multiple disjointed connected components. Like PCR, ACR employs a simple localized cut-vertex detection procedure that only requires 1-hop positional information to detect critical nodes. The procedure is based on

(Jorgic et al. 2004) and runs on each node in a distributed manner to determine locally whether a node is critical or not.

Each actor determines locally whether it is critical or not based on its neighbor's position information. It calculates the distance between the neighbors based on their positions. If the distance is less than their communication range than the actor is non-critical because neighbors stay connected. On the other hand, if the 1-hop neighbors of an actor can be partitioned into more than one segment, then the actor is 1-hop critical. Figure 5.2 shows the critical (shaded circles) and non-critical nodes. For instance, Figure 5.2 also shows a localized view of critical actor A_2 (dotted line) and non-critical node A_{99} (solid line). Node A_2 is 1-hop positional critical since its 1-hop neighbors A_9 and A_{25} become disconnected without A_2. Hence, this network segment will be divided into two sub networks. On the other hand, neighbors of A_{99} i.e. A_4 and A_6 remain connected without it. Therefore, A_{99} is a 1-hop positional non-critical node. Moreover, leaf nodes such as A_3, A_5, A_{73}, etc. are detected as non-critical, since there failure does not affect inter-actor connectivity.

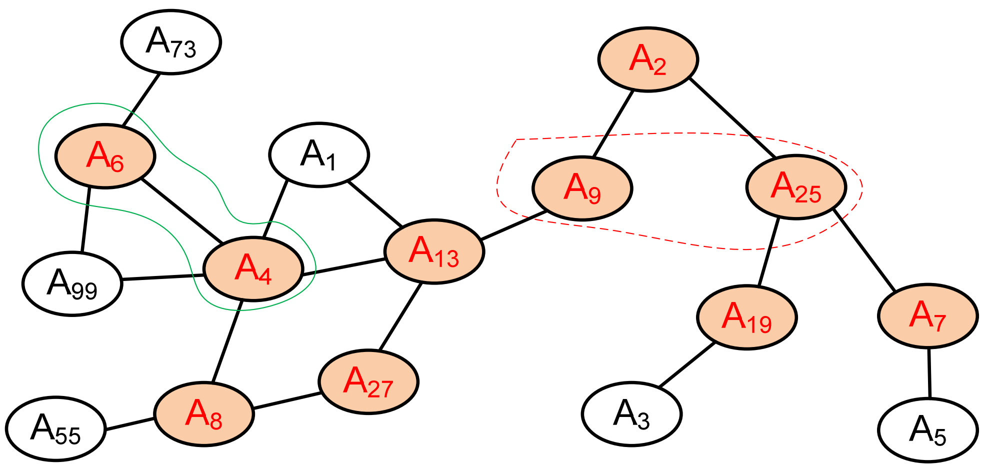


Figure 5.2: A connected inter-actor network with critical and non-critical actors.

5.2.2 Backup selection

Once a critical actor is identified, it chooses an appropriate backup to handle its failure. Each primary preferably picks a non-critical healthy backup among 1-hop neighbors based on its impact on the application, coverage and connectivity. The purpose of the pre-nomination of backup nodes is to instantaneously react to the failure, avoid serious consequences and immediately recover from failure.

Selection of a backup: The actors maintain the minimum state information (i.e. 1-hop neighbors) to avoid excessive messaging overhead. Since with 1-hop information, neighbors of the failed critical actor become disconnected and cannot coordinate, backup actors are determined and notified before a failure of critical nodes takes place. Consider the inter-actor topology presented in Figure 5.3 and assume the parameter values in Table 5.1 to have a better understanding of the procedure. The selection of a backup among 1-hop neighbors is based on the following ordered criteria:

a. Neighbor position (NP): As discussed above, each actor determines whether it is critical or non-critical depending on the position of that node in the topology. A non-critical neighbor actor is preferred to serve as backup because it will limit the scope of recovery that ultimately reduces the implication on an application, coverage and connectivity. For example, critical actor A_8 prefers to appoint non-critical node A_{55} as backup instead of critical actors A_4 and A_{27} as shown in Figure 5.3. The arrow head points towards the primary (critical) nodes.

b. Application-level interests: A non-critical neighbor with the most appropriate actor capabilities (AC) and/or executing a non-essential task (least TCI) is a more suitable candidate for backup. Choosing an unsuitable node will be a futile effort because it cannot respond to an event as expected. Moreover, moving an actor executing the least TCI will have a minimum implication on an application-level task. Unlike PCR (Imran et al. 2010c), ACR prefers to choose as a backup a non-critical node among the 1-hop neighbors with a similar AC and the least TCI . As shown in Figure 5.3, node A_2 picks A_{25} as backup because of a higher AC than A_9. Similarly, actor A_{27} chooses node A_8 as backup due to a lower TCI than A_{13}.

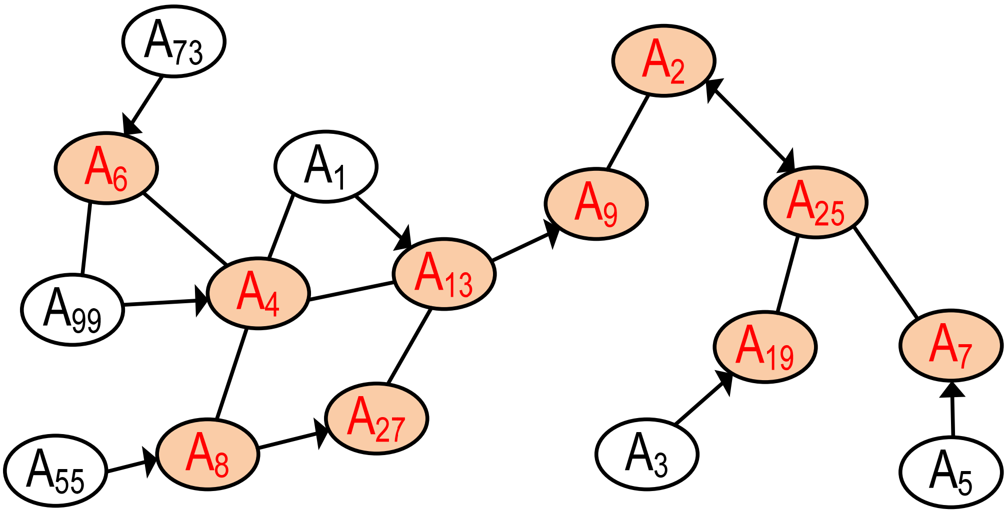


Figure 5.3: Critical actors designate their backup based on the criteria specified.

Table 5.1: Associated parameter values of actors

ID	NP	AC	TCI	Degree
A1	N	4	2	2
A2	C	4	2	2
A3	N	4	3	1
A4	C	4	4	5
A5	N	2	3	1
A6	C	1	2	3
A7	C	4	3	2
A8	C	4	1	3
A9	C	3	3	2
A13	C	4	2	4
A19	C	4	4	2
A25	C	4	3	3
A27	C	3	2	2
A55	N	4	4	1
A73	N	4	2	1
A99	N	4	1	2

c. Connectivity: An actor that causes minimum disturbance to application tasks while having strong connectivity is a better choice to serve as a backup. Moreover, a strongly connected node most probably has non-critical actors in the neighborhood. Furthermore, moving such a node will improve the overall connectivity of the network in addition to limiting the scope of recovery. On the other hand, moving weakly connected nodes may trigger successive cascaded relocations that significantly increase the movement overhead. In contrast to DARA (Abbasi et al. 2007) and PCR (Imran et al. 2010c), ACR appoints higher degree nodes. For example, a cut-vertex A_9 prefers to designate actor A_{13} as backup over A_2 due to a higher degree as shown in Figure 5.3.

d. Overlapping coverage: A strongly connected node has more neighbors that increase the possibility of having actors with more overlapping coverage. A high degree with more overlapping coverage is preferred to serve as a backup. Moving a node with more overlapping coverage will mitigate the effect of the lost actor without major degradation of the coverage in other parts of the network.

It is to be noticed that ACR pursues localized greedy heuristics that may not always lead to an optimal solution. For instance, choosing actor A_{13} as backup for node A_{27} instead of A_8 would result in the overall least TCI. However, it would have required more network state information that is not feasible to maintain. Nonetheless, simulation results have shown that ACR significantly outperforms DARA although it maintains more network state information as will be discussed in Section 5.3.

5.2.3 Primary monitoring and failure detection

Neighbor actors exchange heartbeat messages as part of their network operation to update their status. The chosen backup actors are notified via these messages. A primary may choose a new backup when the existing backup either fails or moves outside the range of the primary that can be detected through missing successive heartbeats. A primary can have only one backup at a time while a backup node can have more than one primary. Once an actor receives BACKUP notification, it starts monitoring the primary through heartbeats. Missing a number of consecutive heartbeats is perceived by the backup as failure of the primary. For instance, a backup node A_{25} detects the failure of primary A_2 shown in Figure 5.4 and initiates a recovery process as detailed in the following section.

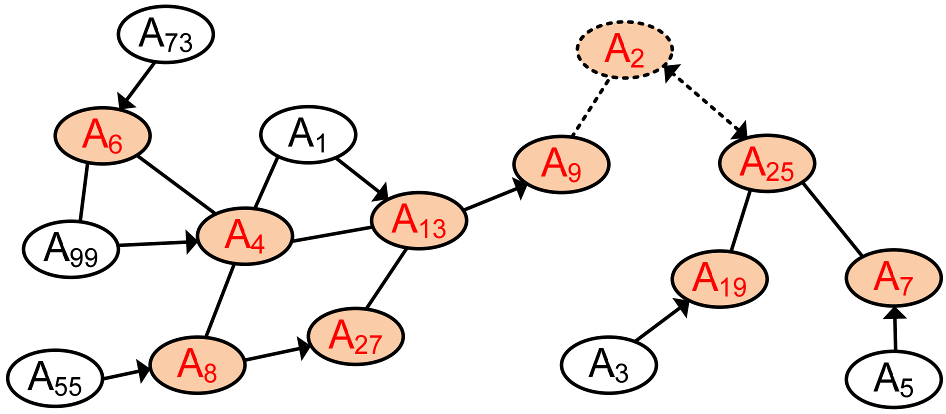


Figure 5.5: Failure of primary actor A_2 is detected by the pre-designated backup node A_{25}

5.2.4 Failure recovery

The pre-designated backup actor immediately initiates a recovery process once it detects the failure of the primary. Three scenarios may be encountered. First, if the backup actor is critical then it checks whether the failed node was also its backup or not, i.e., the two nodes are backup for each other. In Figure 5.3, nodes A_{25} and A_2 are serving as each other's backup. Now the backup actor chooses and appoints another backup using the same criteria as specified in the preceding section. For example, Figure 5.5 shows that actor A_{25} designates node A_7 as its new backup. A_{25} sends a movement notification message to its newly appointed backup so that it can maintain

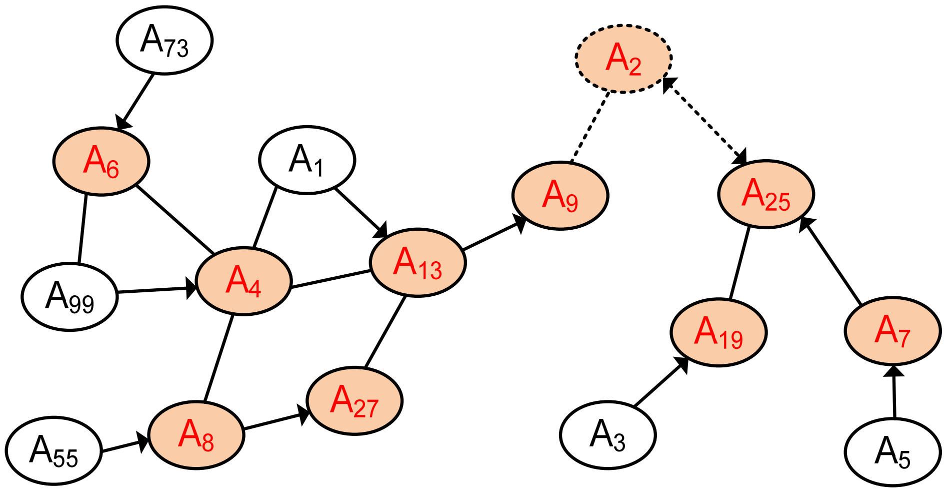


Figure 5.4: Backup A_{25} chooses another backup (since failed node was its backup)

its connectivity with the primary. This is referred to in the literature as coordinated multi-actor relocation (English et al. 2006). Once the backup is notified, the primary moves to the location of the failed node and starts exchanging heartbeat messages with its new neighbors as shown in Figure 5.6. However, moving a critical node further partitions the network, therefore, the algorithm is recursively executed on the notified backup until a non-critical node is reached. Figure 5.6 shows that moving the critical actor A_{25} further partitions the network and the algorithm is recursively applied until the connectivity is restored or the network periphery is reached. The backup nodes successively replace their primary in a cascaded manner. The recovery process is similar for both cases whether the primary node fails or moves as part of recovery.

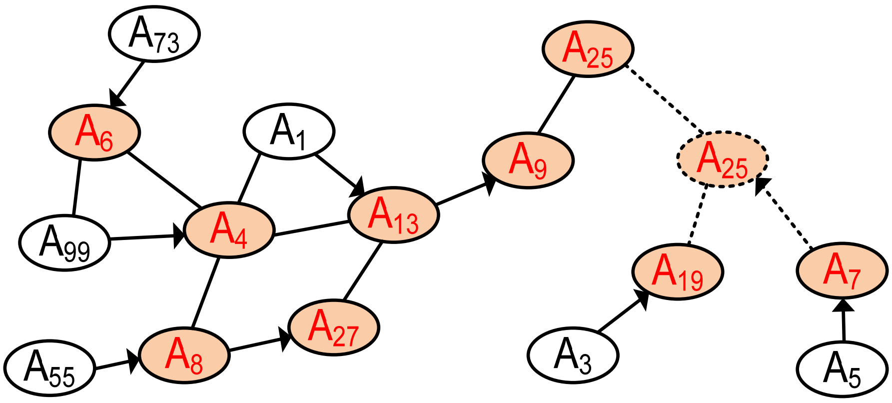


Figure 5.6: The backup node A_7 replaces the primary A_{25}, whereas, non-critical backup A_5 replaces the primary A_7 to complete the recovery.

Second, if the pre-designated backup actor is critical and its backup is alive, then it just sends a movement notification message to its backup and moves to the location of its failed or moved actor as shown in Figure 5.7. Third, if the backup is non-critical then it simply replaces the primary and the recovery is complete as shown in Figure 5.7.

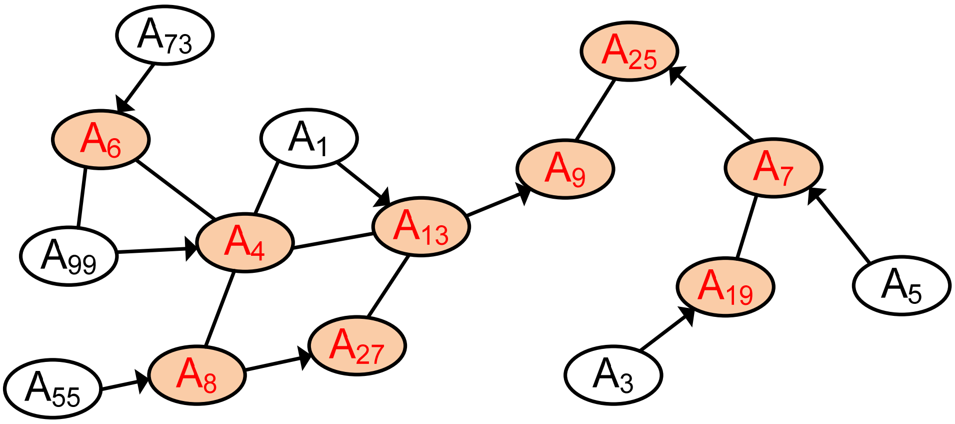


Figure 5.7: *A25* sends movement notification message to newly appointed backup and moves to location of A_2.

5.2.5 ACR algorithm analysis

We analyze the performance of the proposed ACR algorithm in this subsection. The analysis shows that the presented ACR algorithm successfully restores the connectivity severed due to failure of a cut-vertex actor. We introduce the following theorems to prove the correctness and analyze the performance of the ACR algorithm:

Theorem 5.1: *ACR guarantees to rejuvenate the broken connectivity due to failure of a cut-vertex and converge in a maximum of N-3 relocations, where N is the number of actors.*

Proof: Since, ACR relocates a pre-designated backup node to the place of the failed primary (moves primary in case of cascaded relocation). This restores all the communication links broken due to either failure or movement of the primary(s) because ACR assumes a symmetric communication range for all the actors. The recovery process in ACR is guaranteed to terminate once a non-critical backup node replaces the critical primary and no node moves more than once. A connected network of N actors (where $N \geq 2$) has at least $N-2$ critical nodes. In such network topology, each actor has a degree of 2 as shown in Figure 5.8. In the worst case, ACR relocates $N-3$ actors for such topologies. □

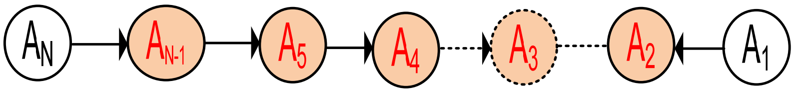


Figure 5.8: The worst case topology for ACR in which it relocates *N-3* actors.

Theorem 5.2: *The maximum travel overhead imposed by ACR on each backup involved in recovery is r, where r is the communication range of the actors.*

Proof: As mentioned earlier, ACR is a localized approach that only maintains 1-hop neighbour information at each actor. The maximum distance between the neighbour actors is equivalent to their communication range (i.e., *r*) because we assume a free space propagation model. Each critical actor appoints a backup from among its neighbours that replaces the primary in case it fails or moves to recover from another node failure. ACR moves each backup only once, therefore, the maximum distance to move an actor is equal to *r*. In other words, ACR pursues cascaded or shifted relocation in order to avoid moving an actor for a long distance. □

Theorem 5.3: *The worst-case time complexity to repair inter-actor connectivity in ACR is O (r/m + t (N-3)), where r is the distance, m is the movement speed and t is the time to receive the message.*

Proof: Since ACR performs pre-failure planning (i.e., designates an appropriate neighbour as backup before the failure), therefore, the worst-case time complexity is proportional to the time required to detect the failure, sends a movement notification to the backup and perform relocation. Similarly, the backup receiving the notification message repeats the same steps and moves to replace the primary. The relocations can be performed in parallel in such a way that as soon as a backup receives a notification, the maximum time it takes to replace the primary will be r/m, where *r* is the distance represented by the communication range and *m* is the movement speed of an actor. The time it takes to detect a failure and send a message to the backup is *t*. The maximum number of relocations to perform in ACR is *N-3* as proven in Theorem 5.1. Therefore, the total recovery time in the worst case will be equivalent to r/m + *t* (*N-3*) which is *O* (r/m + *t* (*N-3*)). □

Theorem 5.4: *ACR may introduce a new cut-vertex as a consequence of the recovery process.*

Proof: As proven in Theorem 5.1, ACR restores all the communication links broken due to failure of a cut-vertex node and terminates when a non-critical backup node replaces the critical primary. ACR only introduces a new cut-vertex as a result of recovery when an intermediate non-critical backup node replaces the primary. Figure 5.9 shows the scenario in which moving an intermediate non-critical backup A_2 causes another intermediate non-critical node A_4 to become critical. □

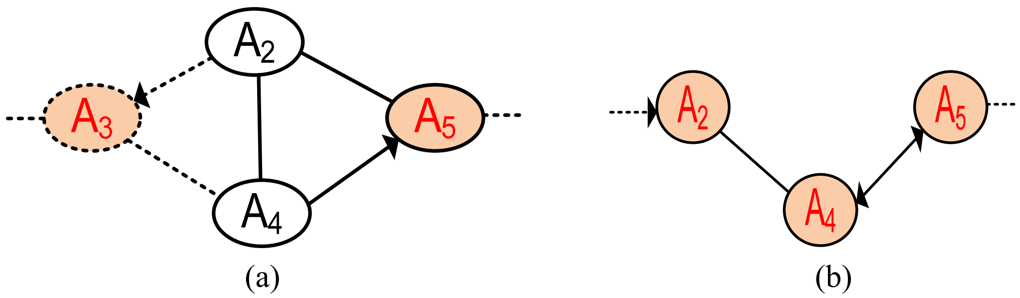


Figure 5.9: Impact of moving an intermediate non-critical node; (a) A_2 detects the failure of primary A_3 (b) A_4 becomes critical as a result of moving A_2.

Theorem 5.5: *The maximum number of messages required in ACR for connectivity restoration is O(N) where N is the number of actors in WSAN.*

Proof: ACR is a localized approach that only maintains 1-hop neighbour information to recover from an actor failure. This costs 1 message for each actor. Moreover, each critical backup involved in the recovery has to send 1 movement notification message to its backup in order to avoid being perceived as faulty. Recovery announcements at the new location are considered as part of regular status updates for maintaining the 1-hop table. In the worst case, when all the critical actors *(c -1)* move, the total number of messages will be *(N+c-1)*. Hence, the worst case message complexity of ACR is *O (N)*. □

5.2.6 Pseudo code of ACR algorithm

Figure 5.10 show the high level pseudo code of procedures in ACR for identifying 1-hop positional critical nodes and finding the appropriate backup. Each actor executes the procedure *Find1HopPosCritical ( )* in a distributed manner to determine whether it is critical or not. If it is critical, it looks for an appropriate backup among the1-hop neighbors using the procedure *FindBackup()* (lines 1-5). A critical actor *F* looks among its neighbors for a candidate backup. If there is an actor *A* that is non-critical then *F* marks it as the best candidate (i.e., BC) and assumes that A has the highest AC and the least distance to F (lines 6-10). If there is a non-critical actor in the neighborhood of F then it looks for an appropriate actor with a similar AC. The procedure *FindBestAC(F)* is used to preferably find a neighbor with a similar AC. If such a neighbor is not found then it looks for an actor with the minimum higher AC. A neighbor node with the maximum lower AC is marked as the best candidate, if none of them have an equal or higher AC (lines 11-17). If there is more than one non-critical node having similar AC, *F* marks the node with the least TCI as the best candidate (lines 18-25). Again, if multiple non-critical nodes have equal AC and TCI, the actor with the highest node degree among the neighbors of *F* is marked as the best candidate (lines 26-36). Otherwise, to break the tie among multiple actors having the same AC, TCI and node degree, an actor that has the highest overlapping coverage with *F* is marked as the best candidate (lines 37-46). On the other hand, if a non-critical actor among the neighbors of F is not available, then *F* looks for the best candidate among the critical nodes based on the same criteria specified earlier (lines 47-49).

Find1HopPosCritical (*F*)

```
IsCritical(F) ← false
IF (A divides 1-hop neighbors to more than one segment) then
  IsCritical (F) ← True
  FindBackup(F)
ENDIF
```

FindBackup(*F*)

```
∀ N(F) IF ∃A ∈ N(F) ∧ IsCritical(A) == false then
BC ← A
BestAC = GetAC(A)
MinOC  = FindOverlapCoverage (A, F)
IsNonCriticalNeigh ← true
IF (IsNonCriticalNeigh = = true) then
   ∀ N(F) IF ∃B ∈ N(F) ∧ IsCritical(B) == false then
   IF ( BestAC = = FindBestAC (F)) then
      BC ← B
      BestAC = GetAC(B)
      LeastTCI = GetTCI(B)
   ENDIF
   ∀ N(F) IF ∃B ∈ N(F) ∧ IsCritical(B) == false then
   IF ( BestAC = = GetAC (B))
      IF ( LeastTCI > GetTCI(B)) then
    BC ← B
    LeastTCI = GetTCI(B)
    HighDegree = GetNodeDegree(B)
      ENDIF
   ENDIF
```

```
26      ∀ N(F) IF ∃B ∈ N(F) ∧ IsCritical(B) == false then
27      IF ( BestAC = = GetAC (B)
28         IF ( LeastTCI = = GetTCI(B))
29            IF (HighDegree < GetNodeDegree(B)) then
30          BC←B
31          LeastTCI = GetTCI(B)
32          HighDegree = GetNodeDegree(B)
33          LeastDistance = Distance (B, F)
34       ENDIF
35         ENDIF
36      ENDIF
37      ∀ N(F) IF ∃B ∈ N(F) ∧ IsCritical(B) == false then
38      IF ( BestAC = = GetAC (B)
39         IF ( LeastTCI = = GetTCI(B))
40       IF (HighDegree = = GetNodeDegree(B))
41          IF (MinOC < FindOverlapCoverage (A, F) then
42             BC←B
43          ENDIF
44       ENDIF
44         ENDIF
45      ENDIF
46   ELSE
47      If a non-critical node is not present among the neighbors of F
48       then, it looks for the best candidate using the same criteria
49       as specified in lines 12-46.
```

Figure 5.10: Pseudo code of identifying critical nodes and finding a backup.

Figure 5.11 shows if a backup actor *A* detects successive missing heartbeats from the primary *F*, it initiates a recovery process (lines 1-3). The same recovery procedure is applied when the backup node *A* either detects failure of the primary F or receives a movement notification message from *F* (lines 4-5). If the pre-designated backup actor A is critical, it checks whether the failed node was its backup or not. In other words, both A and F were the primary for each other. If the failed primary actor F was also the backup of A (i.e. *SimPrimaryBackup*) then A chooses another backup using *FindBackup()*.

```
FailureDetection (A, F)
1   IF (backup A detects consecutive missing heartbeats from F) then
2     PerformRecovery(A, F)
3   ENDIF
PerformRecovery(A, F)
4   IF (A detects Primary failure F or receives movement notification
5   message from F ) then
6     IF(IsCritical(A) == true) then
7       IF(SimPrimaryBackup(F, A) == true) then
8         FindBackup (A)
9         NotifyBackup(A)
10        MoveToLocation(F, A)
11      ELSE
12        NotifyBackup (A)
13        MoveToLocation(F, A)
14      ENDIF
15    ELSE
16      MoveToLocation(F, A)
17    ENDIF
18  ENDIF
MoveToLocation (F, A)
19      Move A to the location of F
```

Figure 5.11: Pseudo code of failure detection and recovery procedure.

Once the new backup is selected, *A* sends a movement notification message and moves to the location of the failed node (lines 6-10). On the other hand, if the pre-assigned backup A is simply a critical node then it sends a movement notification message and moves to replace the primary (lines 11-14). Otherwise, if the backup node A is non-critical, it moves to replace the primary node F and the recovery is complete (lines 15-17).

5.3 Performance evaluation

The performance of ACR is validated through extensive simulations. This section describes the simulation environment, performance metrics and experimental results.

5.3.1 Simulation setup and performance metrics

We have developed the simulation environment in Visual C++. In the simulation experiments, connected inter-actor network topologies are created that consist of varying the number of actors (20-100). Nodes are placed in an area of 1000 m × 600 m using random uniform distribution. Experiments were performed while varying the transmission range of the actors (50-125). The values of AC and TCI are randomly assigned to actors using discrete uniform distribution in the range [0, 5]. The performance of ACR is assessed using the performance metrics defined in Table 5.2.

Table 5.2: Performance metrics for DCR algorithm

Performance Metrics	Description
Max change in AC	This metric captures the variations in the AC caused by the swapping of actor positions. It reports the maximum change in the AC when a node replaces another node. This includes the backup and the failed node as well as the motion triggered by the subsequent relocation until the recovery algorithm terminates. This metric in essence indicates the readiness of the network to handle serious events in the vicinity of a replaced actor given the capabilities of the node that moved in. ACR strives to move more capable actors with respect to the failed one so that the on-going network operation should be sustained effectively.
Average TCI	Measures the average TCI of all the nodes participating in the recovery. This metric reflects the level of disturbance caused to critical tasks being carried out by actors.
Total distance moved	*Total movement distance:* reports the total distance moved by all actors during recovery: This gauges the efficiency of the ACR algorithm in terms of energy efficiency, recovery time and overhead.
Number of nodes	*Number of actors* moved during the recovery: This metric reflects the scope of the recovery which indicates the level of disturbance to the network operation.
Number of messages	*Number of coordination messages exchanged:* Again this metric indicates the energy consumption and recovery overhead in terms of communication.
Percentage of coverage reduction	*Percentage of Area coverage reduction* relative to the pre-failure level: assess how effectively ACR limits the coverage loss while appointing backup actors.
Average node degree	*Average node degree:* measures the level of inter-actor connectivity and availability of alternative paths.

The following parameters were used to vary the WSAN configuration in the simulation experiments and study the impact on the performance of ACR:

- *Number of placed actors (N):* This parameter affects the actor density and the inter-actor connectivity. Boosting the actor density increases the number of non-critical nodes in addition to growing the area coverage.
- *Actor communication range (r):* The communication range influences the inter-actor connectivity and highly affects the recovery overhead in terms of the travelled distance and the number of involved actors.

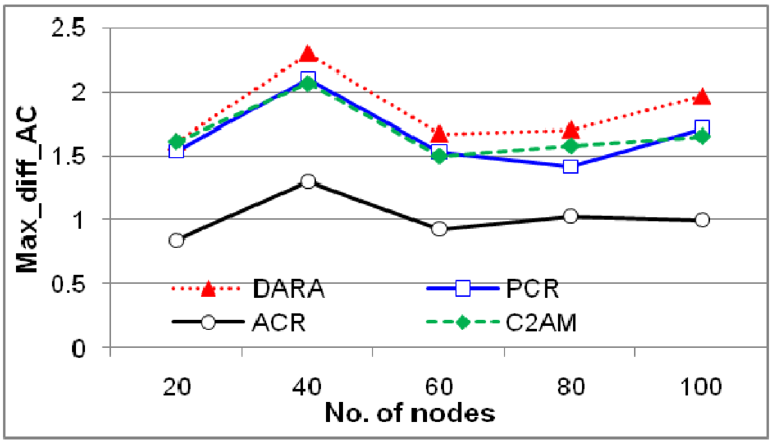
5.3.2 Baseline approaches

The performance of ACR is compared to DARA (Abbasi et al. 2007; Abbasi et al. 2009b), PCR (Imran et al. 2010c) and C^2AM (Abbasi et al. 2009a). Like ACR, all three algorithms are distributed and exploit actor mobility to recover from node failures. However, their approach is different. Unlike ACR, DARA and C^2AM are reactive approaches that replace a failed node *F* with one of its suitable neighbors and continue successive relocations until connectivity is restored or the network periphery is reached. DARA does not factor in the application-level interest at all; whereas, C^2AM only considers the importance of the currently-executed task. Neither DARA nor C^2AM consider the actor capability and coverage. On the other hand, PCR is a hybrid approach that moves a pre-designated backup to recover from a primary failure. PCR is recursively applied only when the backup is also a critical node. PCR does not consider application-level constraints while recovering from node failures.

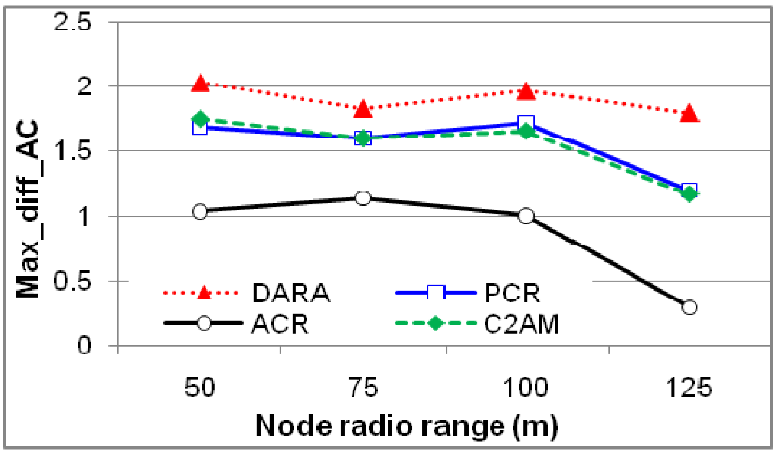
5.3.3 Results and analysis

The experiments involve randomly generated topologies with a varying actor count and communication range. The number of actors has been set to 20, 40, 60, 80 and 100. The communication range of actors is changed among 50, 75, 100 and 125. When changing the node count, "*r*" is fixed at 100 m and "*N*" is set to 100 while varying the communication range. The results of individual experiments are averaged over 30 trials. All results are subject to 90% confidence interval analysis and stay within 10% of the sample mean.

a. **<u>Max change in AC:</u>** Figure 5.12 confirm the effectiveness of ACR over other contemporary schemes in terms of considering application-level concerns. Basically, ACR avoids making major changes in the acting capabilities in a particular region. ACR strives to pick a backup with as close an AC value as possible to the primary node. Both graphs indicate that ACR consistently outperforms application-oblivious schemes in terms of caring for application interests while varying the actor density and communication range. This is because increasing the number of actors leads to stronger inter-actor connectivity and hence, more candidate actors would be available around the failed node. This allows ACR to designate a nearby actor with suitable capabilities. Figure 5.12(b)



(a)

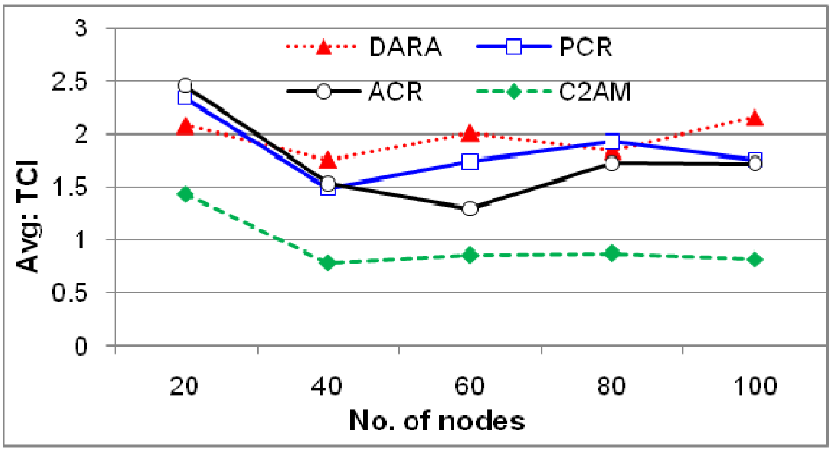


(b)

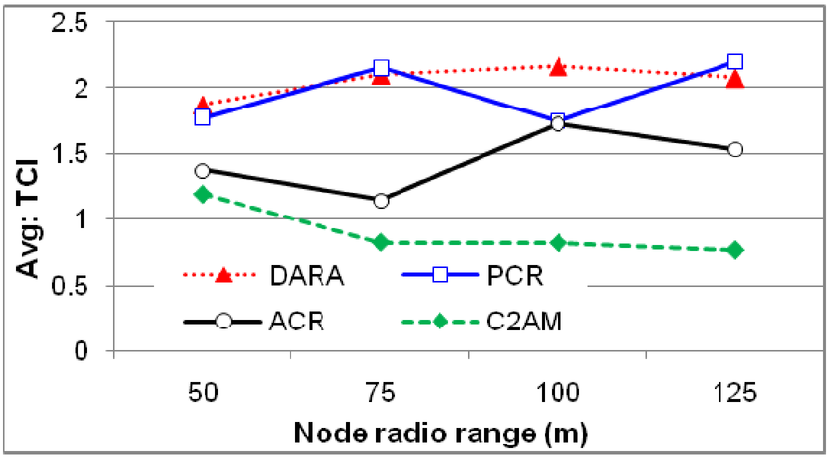
Figure 5.12: The change in the actor capabilities profile in the network due to the recovery, as a function of *N* in (a) and *r* in (b).

further confirms our inference. It indicates that increasing the communication range mostly sustains the action capabilities profile by minimizing the difference between the AC value of the primary and backup nodes. Again, this is due to increased inter-actor connectivity.

b. **Average TCI:** Figure 5.13 reports the average TCI for the actors involved in the recovery. The plot in essence indicates the application-level disturbance caused due to moving nodes during the recovery. Since the primary concern of C^2AM is to minimize disruption to an on-going operation, it performs better than the other schemes. Figure 5.13(a) suggests that the performance of ACR surpasses that of DARA and PCR especially for a larger value of N. The obvious reason is ACRs



(a)

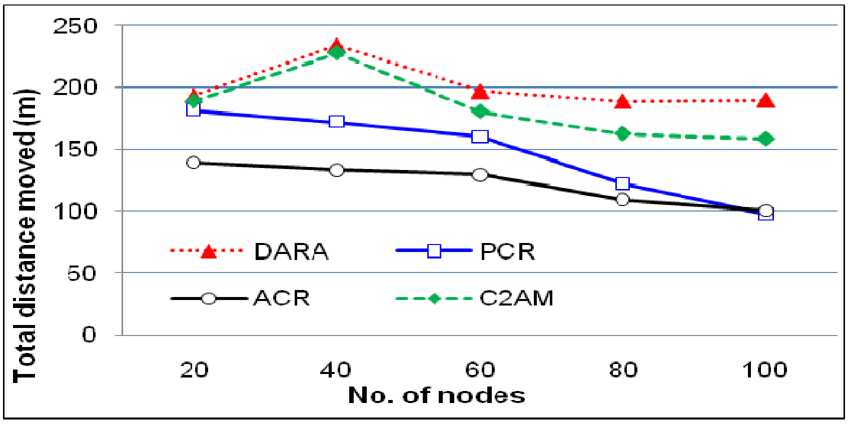


(b)

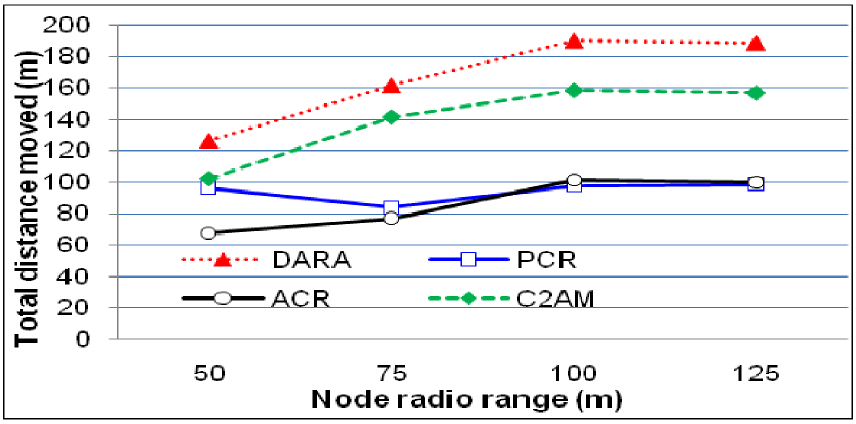
Figure 5.13: The impact on critical tasks due to the recovery, as a function of *N* in (a) and *r* in (b).

priority of moving non-critical nodes with appropriate actor capabilities. In other words, ACR strives to balance between the application interests and recovery overhead while minimizing the impact on coverage and connectivity. We will come back to this point later in this section. Figure 5.13(b) confirms the effectiveness of ACR over other application-unaware schemes in terms of interrupting critical tasks while varying the transmission range. This is due to the high node density that increases the number of neighbors. Moreover, increasing '*r*' further boosts the node degree, i.e., the number of neighbors of a node, which enhances the prospect for picking a more suitable backup. The figure might give an impression that the performance of ACR becomes worse with the increased transmission range. In fact, this is due to ACRs preference of limiting the recovery scope and balancing the utilization of actor capability as can be observed from Figure 5.12 and later in Figure 5.15.

c. **Total movement distance:** Figure 5.14 shows the total distance moved by all nodes until the connectivity is restored. As the Figure 5.14(a) indicates ACR consistently outperforms the baseline schemes, especially for sparse networks. This is because ACR designates high-degree nodes as backup which increases the probability of having non-critical nodes in the neighborhood. Thus, it strives to avoid successive cascaded relocations. Figure 5.14(a) suggests that despite considering application constraints, the performance of ACR scales very well and is not affected by the node density because of choosing non-critical nodes as backup. While varying the transmission range, ACR incurs far less overhead than DARA and C^2AM due to limiting the scope of cascaded relocations by choosing non-critical actors as shown in Figure 5.14(b). As the figure indicates, the performance of ACR is marginally better than PCR for low communication ranges. This is because of designating strongly connected actors as backup. Moreover, the performance of ACR is not much affected by increasing the communication range despite ACR's concern about application-level constraints in addition to minimizing the number of nodes involved in recovery as will be later discussed. The performance of DARA and C^2AM worsens with the growth in the transmission range because of the increased distance between nodes. Another important observation that can be made from both the figures is that the performance gap between ACR and PCR starts to decrease while increasing the actor density and communication range. This is because of the availability of more non-critical nodes, since, inter-actor connectivity is improved in both cases.



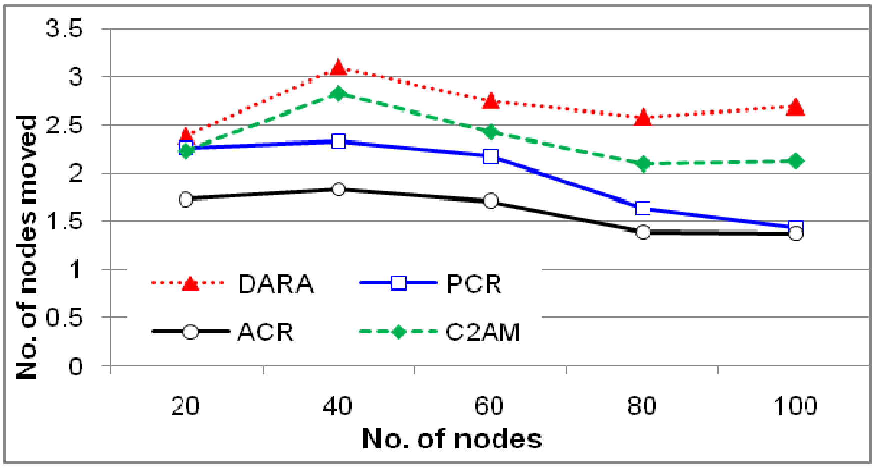
(a)



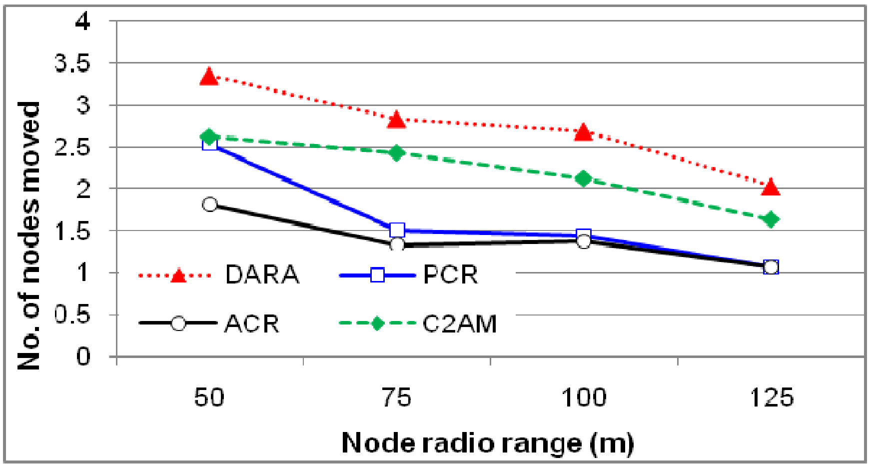
(b)

Figure 5.14: The total distance moved by all nodes involved in recovery, as a function of N in (a) and r in (b).

d. **Number of nodes moved:** Figure 5.15 shows the number of recovery participants when ACR and the baseline approaches are applied. The performance graphs confirm the advantage of ACR which moves fewer actors than all the other approaches. This is because ACR limits the scope of the recovery and avoids successive cascaded relocations by choosing non-critical nodes as backup. Moreover, ACR designates high degree nodes as backup in contrast to other approaches. This further helps ACR because strongly connected nodes have more probability to have non-critical nodes in the neighborhood. Furthermore, the performance of ACR improves with the high actor density and communication range that indicates great scalability. In both the cases, the number of neighbors increases due to the stronger inter-actor connectivity. This offers better choices for candidate backups.



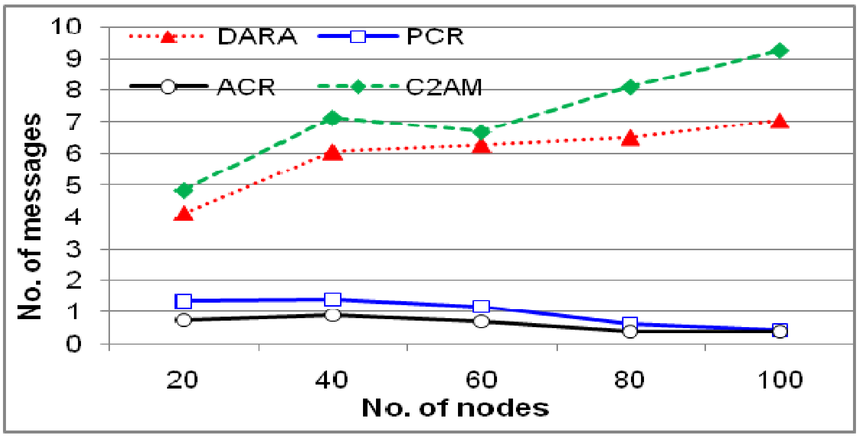
(a)



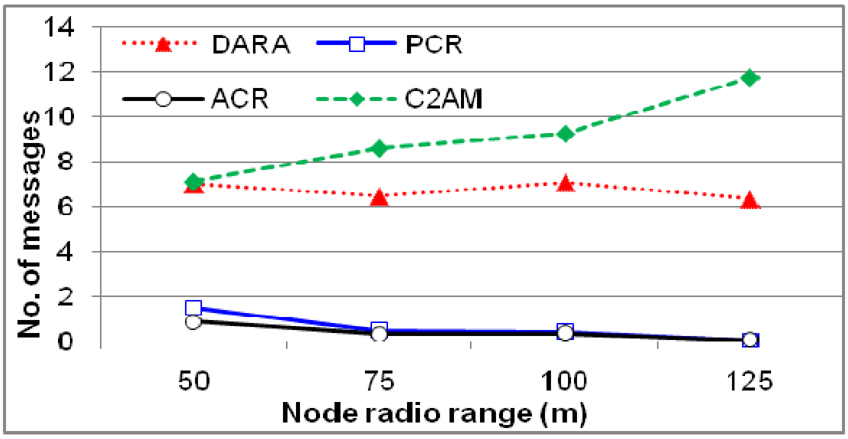
(b)

Figure 5.15: The effect of changing *N* (a) and *r* (b) on the total number of nodes involved in the recovery.

e. <u>Number of coordination messages:</u> Figure 5.16 reports on the coordination messaging overhead as a function of the network size and radio range. As the figure indicates, ACR incurs far less messaging overhead than reactive approaches like DARA and C^2AM at the cost of high computation overhead in picking the backup. Like PCR, ACR often strives to engage non-critical nodes as backup because they do not require exchanging coordination messages. Moreover, ACR appoints highly connected nodes as backup which increases the possibility of having non-critical nodes in the neighborhood. This limits the cascaded relocations and thus reduces the number of coordination messages. Furthermore, ACR only requires maintaining 1-hop network state information; therefore, the messaging overhead is less. Unlike DARA and C^2AM, the performance of ACR is



(a)

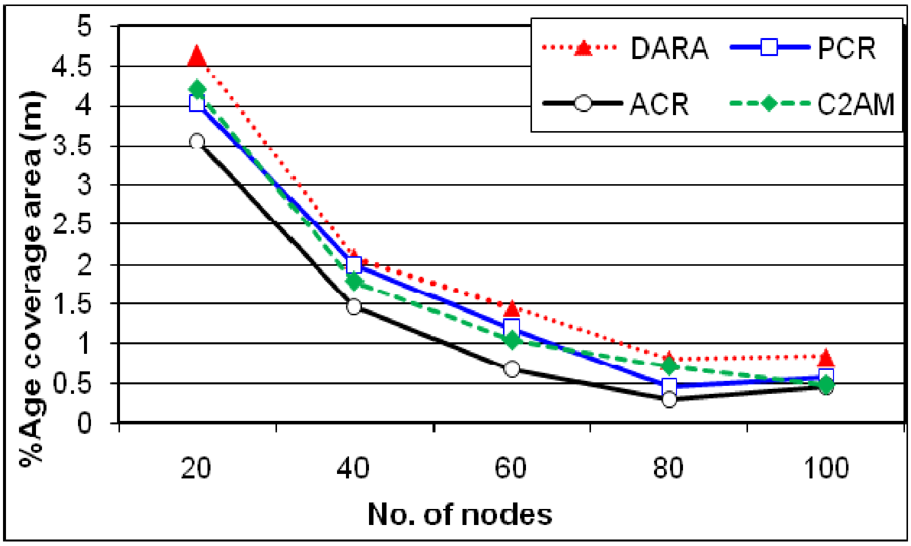


(b)

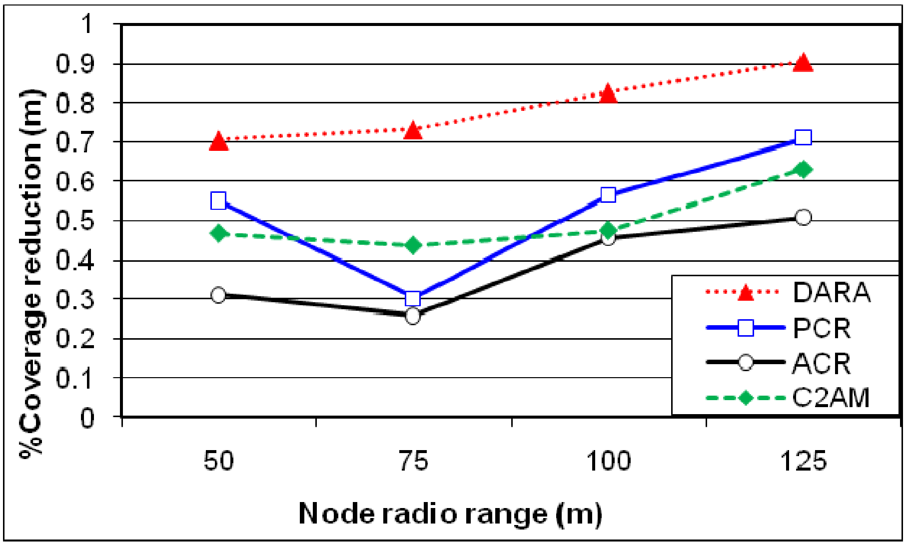
Figure 5.16: The total number of messages exchanged during recovery, while varying the node density (a) and radio range (b).

improved with the high actor density and communication range. Under ACR, the average number of coordination messages sent by a node in Figures 5.16(a) and (b) are 0.37-0.9 and 0.07-0.87, respectively. As both figures indicate, C^2AM incurs slightly more messaging overhead than DARA. At first instance, it seems surprising but the matter of fact is that C^2AM strives to look for the best candidates having the least TCI which are often found in the very active parts of the network. Therefore, it requires more messaging as the number of neighbors increases.

f. Percentage of coverage reduction: Figure 5.17 show the impact on coverage, measured in terms of the percentage of coverage reduction relative to the pre-failure level, while changing the *N* and *r*. The action range is set to 50 m in these experiments. Overall, ACR limits the coverage loss and consistently outperforms other approaches. Although increasing the node density helps, PCR and C^2AM still do not make up for the coverage loss and definitely do not match ACR's performance. The advantage of ACR in terms of coverage is obviously due to moving high degree non-critical nodes, which most probably have a high overlapping coverage. Moreover, the limited scope of node relocation also limits the coverage loss at the network periphery. Figure 5.17(b) indicates that the performance of ACR in terms of coverage reduction is not much affected with increasing the communication range. On the other hand, the performance of DARA significantly worsens when growing the communication range. With the increased value of *r*, the network becomes more connected and the number of neighbors of *F* grows. DARA moves nodes with the least degree that may be critical nodes and do not have overlapping coverage. Furthermore, moving critical nodes triggers successive relocations that cause significant coverage loss at the network periphery.



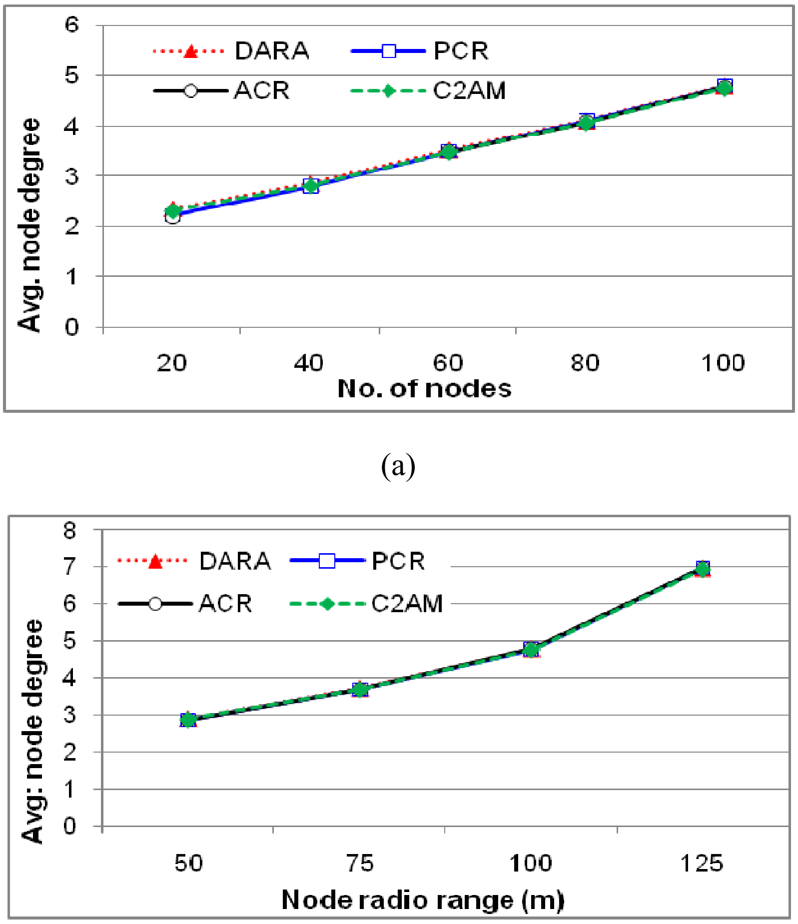
(a)



(b)

Figure 5.17: The impact of recovery on coverage, while varying the node density (a) and radio range (b).

g. Average node degree: Figure 5.18 show the level of connectivity maintained by all approaches after the recovery is completed. As both figures indicate that ACR consistently maintains the same level of connectivity of the other approaches, despite the fact that ACR is not factoring connectivity only like the other baseline approaches. This is due to moving high degree non-critical nodes and limiting the scope of relocation. Figures 5.12-5.18 confirm that that ACR strikes a balance between the various objectives.



(a)

(b)

Figure 5.18: The degree of inter-actor connectivity after recovery, while varying the network size (a) and radio range (b).

5.4 Summary of the proposed algorithms

Table 5.3 summarizes the proposed movement control connectivity restoration algorithms presented in chapters 3-5.

Table 5.3: Summary of the proposed algorithms

		VCR	PCR	DCR	RAM	ACR
Primary objective		Connectivity restoration	Connectivity restoration	Connectivity restoration	Connectivity restoration	Application-centric Connectivity restoration
Connectivity Restoration Method		Increase partially utilized transmission power and relocate internal actors	Relocate internal actors	Relocate internal actors	Relocate internal actors	Relocate internal actors
Relocation approach		Cascaded or shifted	Cascaded or shifted	Cascaded or shifted	Cascaded or shifted	Cascaded or shifted
Network state information		1-Hop neighbor list	1-Hop neighbor list	1-Hop neighbor list	1-Hop neighbor list	1-Hop neighbor list
Approach		Reactive	Hybrid	Hybrid	Hybrid	Hybrid
Failure monitoring		All 1-hop neighbors	Only backup	Only backup	Backup/Grand backup	Only backup
Handle failure of		Any actor	Critical	Critical	Critical/Non-critical	Critical
Whether assess impact of failure		No	Yes	Yes	Yes	Yes
Handle failure of one or multiple actors		One	One	One	Multiple	One
Recovery Strategy	**Execution**	Distributed	Distributed	Distributed	Distributed	Distributed
		Sequential	Sequential	Sequential	Parallel	Sequential
	Recovery Process	Volunteer actors move towards F until become connected	Preferably a non-critical neighbor replaces F	Pre-designated backup replaces primary	Pre-designated backup (s) replace its/their respective primary	Pre-designated backup replaces primary
		Algorithm is recursively applied if receives a movement notification message from parent	Algorithm is recursively applied if receives a movement notification message from parent	Algorithm is recursively applied if the moved backup is critical i.e. Cascaded relocation	Algorithm is recursively applied if the moved backup (s) is/are critical i.e. Cascaded relocation	Algorithm is recursively applied if the moved backup is critical i.e. Cascaded relocation

5.5 Chapter summary

In this chapter, we have presented a novel hybrid Application-centric Connectivity Restoration (ACR) algorithm that factors in application-level concerns in addition to resource optimization while recovering from critical node failures. The proposed ACR algorithm identifies critical actors (i.e. primaries) based on localized information and designates for them backup actors as part of pre-failure planning to minimize recovery time. ACR factors in application-level constraints such as actor action capability and a critical task index while appointing backup nodes to avoid application level failures. It strives to reduce the scope of recovery and incurred overhead by choosing nearby non-critical neighbors as backups. ACR designates highly connected backup nodes with overlapping coverage in order to minimize the impact of critical node failure on coverage and connectivity. In post-failure recovery, it pursues controlled and coordinated actor relocation in order to reorganize the topology and regain the pre-failure strong connectivity. The simulation results have confirmed the effectiveness of ACR in terms of minimizing recovery time, satisfying application requirements and reducing recovery overhead. The results have also shown that ACR outperforms contemporary recovery schemes and limits the impact of the node failure on the network coverage and connectivity.

Moreover, the chapter concludes with the summary of our proposed reactive and hybrid connectivity restoration algorithms presented in chapters 3, 4 and 5 respectively. The next chapter presents the summary of contributions and briefly discuss the future directions.

CHAPTER 6

CONCLUSION AND FUTURE WORK

This chapter concludes the dissertation with a summary of achievements in this research and provides some future directions. First, we present the summary of this dissertation. The second section highlights the contributions made in order to restore inter-actor connectivity with minimum recovery overhead while considering the application-level constraints besides minimizing the impact on coverage. The last section gives a hint of our follow-up-work in order to address the emerging problems.

6.1 Localized movement control connectivity restoration algorithms

This dissertation addressed the problem of restoring inter-actor connectivity lost due to failure of one or a special case of multi–actor failure. The main objective of the connectivity restoration process is to minimize the recovery time and overhead, and reduce the impact of recovery on actor coverage. We have proposed localized and distributed (execution) approaches to connectivity restoration. The proposed VCR (Imran et al. 2010b), PCR (Imran et al. 2010c), DCR, RAM and ACR (Imran et al. 2011) only require each actor to maintain minimal network state information (i.e. 1-Hop neighbors). The main idea is to pursue controlled and coordinated relocation of existing (internal) actors in order to recover from a node failure. Moreover, we cared for application-level constraints on mobility of actors while recovering from an actor failure. We have analyzed the performance of proposed approaches and validated them through extensive simulations. Simulation results confirm the supremacy of the proposed algorithms compared to the best contemporary approaches available in literature that address the same problem.

6.2 Summary of the contributions

This section summarizes the contributions and highlights the distinct features of the proposed solutions:

6.2.1 A new way of handling single actor failure: This dissertation has presented a novel reactive VCR algorithm for connectivity restoration that exploits the partially utilized transmission range of neighboring actors and moves them towards the failed node in chapter 3. The performance of the proposed algorithm is validated through extensive simulations. Simulations results confirm the effectiveness of the proposed algorithm against contemporary recovery schemes. Moreover, we have proposed a novel hybrid PCR algorithm for connectivity restoration in chapter 4 in order to minimize the recovery time and unnecessary overhead. The PCR identifies cut-vertices as part of pre-failure planning and designates for them backup nodes. The pre-assigned backup nodes detect the failure and execute a recovery procedure that may involve successive relocations. We analyze the convergence of the PCR, proving its correctness and confirming its efficiency. The PCR outperforms other published recovery schemes. Furthermore, a variant of the PCR; the DCR algorithm is also presented and validated through simulations in the same chapter.

6.2.2 A novel approach to handling a special case of multiple actor failure: We have presented a novel hybrid recovery algorithm RAM in Section 4.2 to handle a special case of multi-actor failure. The RAM identifies critical actors (i.e. cut-vertices) in advance and designates for them distinct backups. The designated backups execute recovery in parallel once the failure of the primary actor (s) is detected. The performance of the proposed algorithm is analyzed and validated through simulation.

6.2.3 A novel approach to satisfying application-level interests during connectivity restoration: This dissertation has presented a novel hybrid application-centric connectivity restoration algorithm (ACR). In order to avoid application-level failures, the ACR designates a backup node that best matches the actor capabilities of the primary and is executing the least critical task besides minimizing the recovery overhead and impact on coverage and connectivity. To the best of our knowledge, the ACR is the first hybrid algorithm that considers application-level constraints on mobility of actors while recovering from an actor failure. Simulation results confirmed the effectiveness of the ACR in terms of meeting the connectivity restoration and/or application-level goals while minimizing the recovery overhead and impact of the recovery on coverage and connectivity.

6.2.4 Maintaining inter-actor connectivity and coverage: The proposed approaches presented in chapter 3, 4 and 5 strive to minimize the scope of recovery and preserve actor coverage during recovery. The average number of neighbors indicates the level of connectivity. On the other hand, coverage is computed as measuring the total area covered by the deployed actors. The analysis and simulation results confirmed that the proposed approaches keep most of the topology intact and minimize the coverage loss as compared to contemporary recovery schemes.

6.2.5 Localized self-diagnosis and self-healing: The prime advantage of this work is the ability to self-diagnose and self-heal from actor failure while only maintaining 1-hop neighbor information. All the proposed approaches in chapter 3-5 require each actor to maintain only a list of direct neighbors. This not only improves the scalability of the network but significantly reduces the communication overhead during recovery. The neighbors of the failed actor diagnose the failure and initiate a recovery process that does not require external intervention.

6.3 Future work

Our future plan is to handle simultaneous failure of multiple (more than two) actors while maintaining localized network state information. For instance, number of actors either collocated or a distance apart from each other may fail at once causing the network to be partitioned into multiple disjointed segments. We plan to assess the impact of actor failures and either employ the mobility of existing actors or deploy additional actors to recover from the damage. It will be based on the level of damage occurred and the particular application. Moreover, we intend to consider coverage and connectivity in an integrated manner while recovering from an actor failure.

REFERENCES

Abbasi AA, Akkaya K, and Younis M. 2007. A Distributed Connectivity Restoration Algorithm in Wireless Sensor and Actor Networks. 32nd IEEE Conference on Local Computer Networks (LCN' 07), 2007. Dublin, Ireland. pp. 496-503.

Abbasi AA, Baroudi U, Younis M, and Akkaya K. 2009a. C^2AM: an algorithm for application-aware movement-assisted recovery in wireless sensor and actor networks. Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly. Leipzig, Germany: ACM. pp. 655-659.

Abbasi AA, Younis M, and Akkaya K. 2009b. Movement-Assisted Connectivity Restoration in Wireless Sensor and Actor Networks. IEEE Transactions on Parallel and Distributed Systems 20(9):1366-1379.

Ahmadi M, and Stone P. 2006. Keeping in touch: maintaining biconnected structure by homogeneous robots. Proceedings of the 21st national conference on Artificial intelligence - Volume 1. Boston, Massachusetts: AAAI Press.

Akkaya K, and Janapala S. 2008. Maximizing connected coverage via controlled actor relocation in wireless sensor and actor networks. Journal Computer Networks 52(14):2779-2796.

Akkaya K, and Senel F. 2009. Detecting and connecting disjoint sub-networks in wireless sensor and actor networks. Journal Ad Hoc Networks 7(7): 1330-1346.

Akkaya K, Senel F, and McLaughlan B. 2009. Clustering of wireless sensor and actor networks based on sensor distribution and connectivity. Journal of Parallel and Distributed Computing 69(6):573-587.

Akkaya K, Senel F, Thimmapuram A, and Uludag S. 2008. Distributed Recovery of Actor Failures in Wireless Sensor and Actor Networks. IEEE Wireless Communications and Networking Conference (WCNC 2008), 2008 pp.2480-2485.

Akkaya K, Senel F, Thimmapuram A, and Uludag S. 2010. Distributed Recovery from Network Partitioning in Movable Sensor/Actor Networks via Controlled Mobility. IEEE Transactions on Computers 59(2):258-271.

Akkaya K, and Younis M. 2006. COLA: A Coverage and Latency Aware Actor Placement for Wireless Sensor and Actor Networks. IEEE 64th Vehicular Technology Conference (VTC-2006), 2006 Montreal, Canada. pp. 1-5.

Akkaya K, and Younis M. 2007. C^2AP: Coverage-aware and Connectivity-constrained Actor Positioning in Wireless Sensor and Actor Networks. IEEE International Performance Computing, and Communications Conference (IPCCC' 07), 2007 New Orleans, LA pp.281-288.

Akkaya K, and Younis M. 2008. Coverage and latency aware actor placement mechanisms in WSANs. International Journal of Sensor Networks 3(3): 152-164.

Akkaya K, Younis M, and Bangad M. 2005. Sink repositioning for enhanced performance in wireless sensor networks. Journal Computer Networks 49(4):512-534.

Akyildiz IF, and Kasimoglu IH. 2004. Wireless sensor and actor networks: research challenges. Journal Ad Hoc Networks 2(4):351-367.

Akyildiz IF, Su W, Sankarasubramaniam Y, and Cayirci E. 2002. Wireless sensor networks: a survey. Journal Computer Networks 38(4):393-422.

Almasaeid HM, and Kamal AE. 2007. Data delivery in fragmented wireless sensor networks using mobile agents. Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems. Chania, Crete Island, Greece: ACM.

Andersen JB, Rappaport TS, and Yoshida S. 1995. Propagation measurements and models for wireless communications channels. IEEE Communications Magazine 33(1):42-49.

Atwood B, Warneke B, and Pister KSJ. 2000. Preliminary circuits for Smart Dust. Southwest Symposium on Mixed-Signal Design, 2000 (SSMSD 2000). pp. 87-92.

Bao L, and Garcia-Luna-Aceves JJ. 2003. Topology management in ad hoc networks. Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and Computing. Annapolis, Maryland, USA: ACM. pp. 129-140.

Basu P, and Redi J. 2004. Movement control algorithms for realization of fault-tolerant ad hoc robot networks. Journal IEEE Network 18(4):36-44.

Batalin MA, and Sukhatme GS. 2002a. Sensor coverage using mobile robots and stationary nodes. Scalability and Traffic Control in IP Networks II. 1 ed. Boston, MA, USA: SPIE. pp. 269-276.

Batalin MA, and Sukhatme GS. 2002b. Spreading out: A local approach to multi-robot coverage. International Symposium on Distributed Autonomous Robotic Systems. Fukuoka, Japan. pp. 373-382.

Batalin MA, and Sukhatme GS. 2003. Coverage, exploration, and deployment by a mobile robot and communication network. Proceedings of the 2nd international conference on Information processing in sensor networks. Palo Alto, CA, USA: Springer-Verlag.

Batalin MA, and Sukhatme GS. 2005. The Analysis of an Efficient Algorithm for Robot Coverage and Exploration based on Sensor Network Deployment. Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA 2005) Barcelona, Spain. pp. 3478-3485.

Borriello G, Farkas KI, Reynolds F, and Zhao F. 2007. Guest Editors' Introduction: Building a Sensor-Rich World. IEEE Pervasive Computing 6(2):16-19.

Bulusu N, Heidemann J, and Estrin D. 2000. GPS-less low-cost outdoor localization for very small devices. , IEEE Personal Communications 7(5):28-34.

Butterfield J, Dantu K, Gerkey B, Jenkins OC, and Sukhatme GS. 2008. Autonomous biconnected networks of mobile robots. 6th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks and Workshops (WiOPT 2008) Berlin, Germany. pp. 640-646.

Casper J, and Murphy RR. 2003. Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 33(3):367-385.

Chatzigiannakis I, Kinalis A, and Nikoletseas S. 2006. Sink mobility protocols for data collection in wireless sensor networks. Proceedings of the 4th ACM international workshop on Mobility management and wireless access. Terromolinos, Spain. pp. 52–59.

Chee-Yee C, and Kumar SP. 2003. Sensor networks: evolution, opportunities, and challenges. Proceedings of the IEEE 91(8):1247-1256.

Correia LHA, Macedo DF, dos Santos AL, Loureiro AAF, and Nogueira JMS. 2007. Transmission power control techniques for wireless sensor networks. Computer Networks 51(17):4765-4779.

Dantu K, Rahimi M, Shah H, Babel S, Dhariwal A, and Sukhatme GS. 2005. Robomote: enabling mobility in sensor networks. 4th International Symposium on Information Processing in Sensor Networks (IPSN' 05), 2005 Los Angeles, California, USA. pp. 404-409.

Das S, Liu H, Kamath A, Nayak A, and Stojmenovic I. 2007. Localized Movement Control for Fault Tolerance of Mobile Robot Networks. In: Orozco-Barbosa L, Olivares T, Casado R, and Bermúdez A, editors. Wireless Sensor and Actor Networks: Springer Boston. pp. 1-12.

Datta A, Gradinariu M, Ammari H, and Das S. 2006. Coverage, Connectivity, and Fault Tolerance Measures of Wireless Sensor Networks. Stabilization, Safety, and Security of Distributed Systems: Springer Berlin / Heidelberg. pp. 35-49.

Demirbas M. 2004. Scalable design for fault-tolerance for wireless sensor networks. Columbus, Ph. D thesis, OH: The Ohio State University.

Duque-Anton M, Bruyaux F, and Semal P. 2000. Measuring the Survivability of a Network: Connectivity and Rest-Connectivity. European Transactions on Telecommunications 11(2):149-159.

English J, Wiacek M, and Younis M. 2006. CORE: Coordinated Relocation of Sink Nodes in Wireless Sensor Networks. in the Proceedings of the 23rd IEEE Biennial Symposium on Communications, 2006. Kingston, Ontario, Canada. pp. 320-323.

Errol LL, and Guoliang X. 2007. Relay Node Placement in Wireless Sensor Networks. IEEE Transactions on Computers 56(1):134-138.

Garcia-Hernandez CF, Ibarguengoytia-Gonzalez PH, Garcia-Hernandez J, and Perez-Diaz JA. 2007. Wireless Sensor Networks and Applications: a Survey International Journal of Computer Science and Network Security 7(3):264-273.

Ghosh A, and Das SK. 2008. Coverage and connectivity issues in wireless sensor networks: A survey. Journal of Pervasive and Mobile Computing 4(3): 303-334.

Goyal D, and J. Caffery J. 2002. Partitioning Avoidance in Mobile Ad Hoc Networks Using Network Survivability Concepts. Proceedings of the Seventh International Symposium on Computers and Communications (ISCC'02), Taormina, Italy.

Guiling W, Guohong C, and La Porta T. 2004. Proxy-based sensor deployment for mobile sensor networks. IEEE International Conference on Mobile Ad-hoc and Sensor Systems, Florida, USA. pp. 493-502.

Gupta G, and Younis M. 2003. Load-balanced clustering of wireless sensor networks. IEEE International Conference on Communications, Anchorage, Alaska. pp. 1848-1852 vol.1843.

Heo N, and Varshney PK. 2003. A distributed self spreading algorithm for mobile wireless sensor networks. IEEE Wireless Communications and Networking (WCNC' 03), 2003 New Orleans, LA, USA pp.1597-1602

Howard A, Mataric MJ, and Sukhatme GS. 2002. Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. 6th International symposium on Distributed Autonomous Robotic Systems. Fukuoka, Japan. pp. 299-308.

Huang C-F, and Tseng Y-C. 2003. The coverage problem in a wireless sensor network. Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications. San Diego, CA, USA: ACM.

Imran M, Said AM, and Hasbullah H. 2008. Personality Wireless Sensor Networks. 7th Annual Seminar on Science and Technology. University Malaysia Sabah.

Imran M, Said AM, and Hasbullah H. 2010a. A survey of simulators, emulators and testbeds for wireless sensor networks. International Symposium on Information Technology (ITSim), 2010 Kuala Lumpur, Malaysia. pp. 897-902.

Imran M, Said AM, Younis M, and Hasbullah H. 2011. Application-centric Connectivity Restoration algorithm for Wireless Sensor and Actor Networks. 6th International Conference on Grid and Pervasive Computing (GPC' 11). Oulu, Finland.

Imran M, Younis M, Md Said A, and Hasbullah H. 2010b. Volunteer-instigated connectivity restoration algorithm for Wireless Sensor and Actor Networks. 2010 IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS). Beijing, China. pp. 679-683.

Imran M, Younis M, Said AM, and Hasbullah H. 2010c. Partitioning Detection and Connectivity Restoration Algorithm for Wireless Sensor and Actor Networks. The 8th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC-2010). Hong Kong, China. pp. 200-207.

Ishizuka M, and Aida M. Performance study of node placement in sensor networks. 24th International Conference on Distributed Computing Systems Workshops, 2004.pp. 598-603.

Jie W, and Shuhui Y. 2006. Optimal movement-assisted sensor deployment and its extensions in wireless sensor networks. 12th International Conference on Parallel and Distributed Systems, 2006 (ICPADS 2006) pp.8.

Jie W, Shuhui Y, and Cardei M. 2008. On Maintaining Sensor-Actor Connectivity in Wireless Sensor and Actor Networks. The 27th IEEE Conference on Computer Communications (INFOCOM 2008) Phoenix, AZ pp. 286-290.

Jorgic M, Stojmenovic I, Hauspie M, and Simplot-ryl D. 2004. Localized algorithms for detection of critical nodes and links for connectivity in ad hoc networks. in the Proceedings of the 3rd Annual IFIP Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net). Bodrum, Turkey. pp. 360-371.

Jun L, and Hubaux JP. 2005. Joint mobility and routing for lifetime elongation in wireless sensor networks. Proceedings of the IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2005 Miami, FL, USA. pp. 1735-1746 vol. 1733.

Kansal A, Rahimi M, Estrin D, Kaiser WJ, Pottie GJ, and Srivastava MB. 2004a. Controlled mobility for sustainable wireless sensor networks. First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON' 04), 2004. Santa Clara, California. pp. 1-6.

Kansal A, Somasundara AA, Jea DD, Srivastava MB, and Estrin D. 2004b. Intelligent fluid infrastructure for embedded networks. Proceedings of the 2nd international conference on Mobile systems, applications, and services (MobiSys'04). Boston, MA, USA.

Kansal A, Yuen E, Kaiser WJ, Pottie GJ, and Srivastava MB. 2004c. Sensing uncertainty reduction using low complexity actuation. 3rd International Symposium on Information Processing in Sensor Networks (IPSN' 04), 2004. Berkeley, California, USA. pp. 388-395.

Koushanfar F, Potkonjak M, and Sangiovanni-Vincentelli A. 2004. Fault Tolerance in Wireless Sensor Networks. in Handbook of Sensor Networks: CRC Press.

Krishnamachari B. 2007. A wireless sensor networks bibliography. Autonomous Networks Research Group, [Online], Available: http://ceng.usc.edu/~anrg/SensorNetBib.html.

Lee S. 2010. Optimized Relay Node Placement for Federating Wireless Sensor Sub-Networks: Ph. D Thesis, University of Maryland, Baltimore County.

Lee S, and Younis M. 2010a. EQAR: Effective QoS-Aware Relay Node Placement Algorithm for Connecting Disjoint Wireless Sensor Sub-networks. IEEE Transactions on Computers (PrePrint).

Lee S, and Younis M. 2010b. Optimized relay placement to federate segments in wireless sensor networks. IEEE Journal on Selected Areas in Communications 28(5):742-752.

Lee S, and Younis M. 2010c. Recovery from multiple simultaneous failures in wireless sensor networks using minimum Steiner tree. Journal of Parallel and Distributed Computing 70(5):525-536.

Lee S, and Younis M. 2010d. Recovery from multiple simultaneous failures in wireless sensor networks using minimum Steiner tree. Journal of Parallel and Distributed Computing 70(5):525-536.

Li N, and Hou JC. 2004. Topology control in heterogeneous wireless networks: problems and solutions; IEEE INFOCOM, Hong Kong, China. pp. 232–243.

Li X, Santoro N, and Stojmenovic I. 2007. Mesh-based Sensor Relocation for Coverage Maintenance in Mobile Sensor Networks. Proceedings of the International Conference Ubiquitous Intelligence and Computing (UIC). Hong Kong. pp. 696-708.

Lin S, Zhang J, Zhou G, Gu L, Stankovic JA, and He T. 2006. ATPC: adaptive transmission power control for wireless sensor networks. Proceedings of the

4th international conference on Embedded networked sensor systems. Boulder, Colorado, USA: ACM.

Liu B, Brass P, Dousse O, Nain P, and Towsley D. 2005. Mobility improves coverage of sensor networks. Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing. Urbana-Champaign, IL, USA: ACM.

McLaughlan B, and Akkaya K. 2007. Coverage-based Clustering of Wireless Sensor and Actor Networks. IEEE International Conference on Pervasive Services. Istanbul, Turkey. pp. 45-54.

McMickell MB, Goodwine B, and Montestruque LA. MICAbot: a robotic platform for large-scale distributed robotics; 2003. pp. 1600-1605 vol.1602.

Melodia T, Pompili D, and Akyildiz IF. 2006. A Communication Architecture for Mobile Wireless Sensor and Actor Networks. 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks, 2006 SECON '06 2006 Reston, VA pp.109-118.

Melodia T, Pompili D, Vehbi CG, and Akyildiz IF. 2007. Communication and Coordination in Wireless Sensor and Actor Networks. IEEE Transactions on Mobile Computing 6(10):1116-1129.

Misra SC, Woungang I, Misra S, Liu H, Nayak A, and Stojmenovic I. 2009. Fault-Tolerant Algorithms/Protocols in Wireless Sensor Networks. Guide to Wireless Sensor Networks: Springer London. pp. 265-295.

Nachman L, Kling R, Adler R, Huang J, and Hummel V. 2005. The Intel(R) Mote platform: a Bluetooth-based sensor network for industrial monitoring. Proceedings of the 4th international symposium on Information processing in sensor networks. Los Angeles, California: IEEE Press.

Nojeong H, and Varshney PK. 2005. Energy-efficient deployment of Intelligent Mobile sensor networks. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 35(1):78-92.

Orozco-Barbosa L, Olivares T, Casado R, Bermúdez A, Das S, Liu H, Kamath A, Nayak A, and Stojmenović I. 2007. Localized Movement Control for Fault Tolerance of Mobile Robot Networks. Wireless Sensor and Actor Networks: Springer Boston. pp. 1-12.

Ozaki K, Watanabe K, Itaya S, Hayashibara N, Enokido T, and Takizawa M. 2006. A fault-tolerant model for wireless sensor-actor system. 20th International Conference on Advanced Information Networking and Applications Volume 2 (AINA 2006), 2006. Vienna, Austria pp. 303-307.

Pandya A, Kansal A, and Pottie G. 2008. Goodput and Delay in Networks with Controlled Mobility. IEEE Aerospace Conference, 2008 Montana, USA. pp. 1-8.

Rajkumar R, and Lee I. 2006. Workshop On Cyber-Physical Systems [Online], Available: http://varma.ece.cmu.edu/cps/.

Ramanathan R, and Rosales-Hain R. 2000. Topology control of multihop wireless networks using transmit power adjustment. Proceedings of IEEE 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2000) Tel Aviv , Israel. pp. 404-413.

Schmid S, and Wattenhofer R. 2006. Algorithmic models for sensor networks. In 14th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS). Rhodes Island, Greece. pp. 11.

Senel F, Akkaya K, and Younis M. An Efficient Mechanism for Establishing Connectivity in Wireless Sensor and Actor Networks; 2007 26-30 Nov. 2007. pp. 1129-1133.

Sinopoli B, Sharp C, Schenato L, Schaffert S, and Sastry SS. 2003. Distributed control applications within sensor networks. Proceedings of the IEEE 91(8):1235-1246.

Space and Naval Warfare Systems Center (SPAWAR) [Online], Available: http://www.spawar.navy.mil/robots/.

Stankovic JA. 2008. When Sensor and Actuator Networks Cover the World. ETRI Journal 30(5):7.

Suet-Fei L. 2006. Wireless sensor actuator network for light monitoring and control application. 3rd IEEE Consumer Communications and Networking Conference, 2006 CCNC 2006 pp. 974-978.

Tamboli N, and Younis M. 2009. Coverage-Aware Connectivity Restoration in Mobile Sensor Networks. IEEE International Conference on Communications, (ICC '09) Dresden, Germany. pp. 1-5.

Tamboli N, and Younis M. 2010. Coverage-aware connectivity restoration in mobile sensor networks. Journal of Network and Computer Applications 33(4): 363-374.

Thrun S, Diel M, and Hahnel D. 2003. Scan Alignment and 3D Surface Modeling with a Helicopter Platform. in Proceedings of the 4th International Conference on Field and Service Robotics.

Wang G, Cao G, and Porta TL. 2004. Movement-assisted sensor deployment. 23rd International Annual Joint conference of the IEEE Computer and Communications Societies (INFOCOM'04). Hong Kong. pp. 2469-2479.

Wang G, Cao G, Porta TL, and Zhang W. 2005a. Sensor relocation in mobile sensor networks. Proceedings of the IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005) Miami, FL, USA pp. 2302-2312.

Wang W, Srinivasan V, and Chua K-C. 2005b. Using mobile relays to prolong the lifetime of wireless sensor networks. Proceedings of the 11th annual international conference on Mobile computing and networking. Cologne, Germany: ACM.

Wang X, Xing G, Zhang Y, Lu C, Pless R, and Gill C. 2003. Integrated coverage and connectivity configuration in wireless sensor networks. Proceedings of the 1st international conference on Embedded networked sensor systems. Los Angeles, California, USA: ACM. pp. 28-39.

Wang ZM, Basagni S, Melachrinoudis E, and Petrioli C. 2005c. Exploiting Sink Mobility for Maximizing Sensor Networks Lifetime. Proceedings of the 38th Annual Hawaii International Conference on System Sciences. Big Island, HI, USA: IEEE Computer Society.

Xia F. 2008. QoS Challenges and Opportunities in Wireless Sensor/Actuator Networks. Sensors 8(2):1099-1110.

Xu L, and Santoro N. 2006. ZONER: A ZONE-based Sensor Relocation Protocol for Mobile Sensor Networks. Proceedings of the 31st IEEE Conference on Local Computer Networks. pp. 923-930.

Xu N. 2003. A Survey of Sensor Network Applications. University of Southern California, [Online], Available: http://enl.usc.edu/~ningxu/papers/survey.pdf.

Yick J, Mukherjee B, and Ghosal D. 2008. Wireless sensor network survey. Journal Computer Networks 52(12):2292-2330.

Younis M, and Akkaya K. 2008. Strategies and techniques for node placement in wireless sensor networks: A survey. Journal Ad Hoc Networks 6(4):621-655.

Younis M, Lee S, and Abbasi AA. 2010. A Localized Algorithm for Restoring Inter-node Connectivity in Networks of Moveable Sensors. IEEE Transactions on Computers 59(12):1669 – 1682.

Younis M, Lee S, Gupta S, and Fisher K. 2008. A Localized Self-Healing Algorithm for Networks of Moveable Sensor Nodes. IEEE Global Telecommunications Conference (GLOBECOM' 08), 2008. New Orleans, LA, USA. pp. 1-5.

Younis M, Youssef M, and Arisha K. 2002. Energy-aware routing in cluster-based sensor networks. Proceedings of the 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS' 02), 2002 Fort Worth, Texas pp.129-136.

Youssef A, Ashok A, and Younis M. 2005. Accurate anchor-free node localization in wireless sensor networks. 24th IEEE International Performance, Computing, and Communications Conference (IPCCC' 05), 2005 Phoenix, Arizona. pp. 465-470.

Youssef W, and Younis M. 2008. A cognitive scheme for gateway protection in wireless sensor network. Journal of Applied Intelligence 29(3):216-227.

Youssef W, Younis M, and Akkaya K. 2006. An Intelligent Safety-Aware Gateway Relocation Scheme for Wireless Sensor Networks. IEEE International Conference on Communications (ICC '06), 2006 Istanbul, Turkey. pp. 3396-3401.

Zamanifar A, Sharifi M, and Kashefi O. 2009. A Hybrid Approach to Actor-Actor Connectivity Restoration in Wireless Sensor and Actor Networks. 8th International Conference on Networks. Gosier, Guadeloupe, France pp. 76-81.

Zhang H, and Hou JC. 2005. Maintaining Sensing Coverage and Connectivity in Large Sensor Networks. International Journal of Ad Hoc & Sensor Wireless Networks 1(1-2):89-123.