Object Mobility in Radio Frequency Identification Systems and Underwater Sensor Networks

by

Youssef Nasser Altherwy

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

We investigate two wireless networking problems related to object mobility in the two fields of Radio-Frequency Identification (RFID) Systems, and Underwater Sensor Networks (UWSNs). RFID is an automatic identification technology where inexpensive information storage devices, called tags, are attached to objects for identification purposes. Nowadays, RFID is pervasive with a number of deployed tags fast approaching millions of tags. UWSNs, on the other hand, have attracted attention for their use in scientific studies of marine life, as well as industrial applications such as monitoring underwater oil pipelines. For RFID systems, we investigate the problem of identifying a stream of tags on a moving conveyor belt. For UWSNs, we investigate the problem of assessing the likelihood that two given nodes of an UWSN can reach each other using a path of bounded length or delay. For each problem, we devise a solution method, obtain analytical results on the quality of the derived solution, and present simulation results to verify and gain insight into our findings.

To my mom

Thanks for buying me my first laptop.

Acknowledgements

I would like to express my deepest appreciation to my supervisor, Professor Ehab S. Elmallah, for his continuous and invaluable guidance, support, and mentorship throughout my thesis work. I would also like to extend my special thanks to my Examining Committee members, Professor Ioanis Nikolaidis and Professor Janelle Harms, for their valuable suggestions and constructive feedback.

I would like to thank my family for their unconditional love and support throughout my whole life. Finally, a special thanks goes to my wife for being always at my side and helping me during my study.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter we give an overview of some important examples involving user and object mobility in networking research. We then identify and motivate specific mobility problems on Radio-Frequency Identification (RFID) systems and Underwater Sensor Networks (UWSNs). We conclude by outlining thesis organization and contributions.

1.1 Introduction

Recent advances in wireless communication and networking have opened the door for providing numerous important real time services involving mobile users and objects. The scope of such networking context and services is broad and spans, for example, the following areas.

- The development of *wireless local area networks* (WLANs), which support communication on the scale of a few hundred meters, has allowed the realization of multi-hop mobile adhoc networks (MANETs) to connect a population of mobile users. Since no networking infrastructure needs to exist before deploying a MANET, applications of MANETs are broad and includes, for example, search and rescue operations in disaster recover missions [47, 15].

- The development of cellular wireless networks, which support communication distances on the scale of a few kilometres in each cell, has enabled

mobile users to enjoy voice and data services that are currently an integral part of our day to day life [57].

- The development of low power low data rate WLANs, which support communication distances on the scale of a few tens of meters together with advances in manufacturing miniaturized sensing and computing devices, have enabled the design of *wireless sensor networks* (WSNs). Such networks enable the detection and tracking of mobile objects; a fundamental service in human security and wild life monitoring applications[55].

- The development of Radio-Frequency Identification (RFID) systems, which support communication distances up to a few meters, has enabled the monitoring and tracking of goods and products in supply-chain management applications, as well as monitoring individuals and wild life in controlled spaces [30].

- The development of Underwater Sensor Networks (UWSNs), which support communication distances on the scale of tens of kilometres, has enabled the collection of marine data over vast areas in rivers and oceans [19].

In all of the above cases, information provided to (or about) mobile entities is an indispensable feature of the deployed services. Providing the intended services, however, faces difficulties and challenges caused by the necessity to deal with the unpredictable mobility aspects in each scenario. By the way of example, we identify the following challenges that have received much attention in the literature.

- In MANETs, uncontrolled user mobility causes multi-hop routes to break frequently. Research in this area has introduced both proactive and reactive routing algorithms to cope with the problem.

- In cellular networks, supporting mobility as users move closer to, or away from, a cellular base station (BS) requires careful adjustment of both BS to user, and user to BS, transmission powers to fully utilize the available

forward channel and reverse channel bandwidths. In addition, maintaining un-interrupted voice or data service to a mobile user crossing the boundary of multiple cells requires the development of effective hand-off schemes. Furthermore, providing reliable connection oriented datagram service to a user requires the adaptation of the TCP protocol to work with the wireless connection.

- For WSNs, detection and tracking of mobile objects while preserving sensor energy consumption requires the development of special protocols to adaptively reduce energy consumption of nodes that do not contribute to the detection or the tracking process.

- For RFID systems, the identification of a population of static and mobile tags (each tag is attached to an object, and stores identification information of that object) requires the development of protools that deal with the interference generated by co-located tags.

- Many UWSNs are inherently mobile where nodes travel long distances in an unpredictable fashion with water currents. Thus, network connectivity is frequently interrupted.

Our work in the thesis is motivated by the importance of supporting mobility in modern systems and application scenarios. To this end, we identify specific problems in the two broad areas of RFID systems, and UWSNs, where mobility is the prime source of difficulty.

These two areas have fundamental differences. On the one hand, RFID systems have short communication ranges (no more than a few meters), and their typical applications are within confined spaces in environments that are often under human control. On the other hand, UWSNs have long communication ranges (e.g., dozens of kilometres), and their typical use is in rivers, lakes, and oceans. However, we are encouraged by the existence of many results on many mobility related problems. The availability of such results encourages

the development of convergent methodologies that can be adapted to problems across different types of networks.

1.2 Thesis Scope

In this section, we highlight the specific mobility problems dealt with in the thesis. The first part of the thesis (Chapters 2 and 3) deals with RFID systems. As explained in Chapter 2, the main components of an RFID system are called *readers* (or *interrogators*), and *tags* (or *transponders*). Tags are information storage devices while readers are used to obtain information stored in tags.

Many different types of RFID systems are currently in use. Such systems differ in their working principles, capabilities, and potential areas of applications. RFID systems serve the purpose of providing automatic identification to objects. They offer significant advantages over comparable technologies such as barcode systems, and optical character recognition systems. For example, tags can store significantly larger amounts of information (e.g., several Kbytes) than what is typically encoded in a barcode. In addition, the stored information can be read without a line-of-sight communication (i.e., the tag can be hidden in a box). Furthermore, the information stored in many tags can be retrieved in a short period of time (in the order of milliseconds).

In Chapter 3, we investigate the problem of identifying moving objects placed on a conveyor belt. Conveyor belt scenarios have wide applications in supply-chain management. In fact, according to [56], part of the RFID mandates the ability to read products passing through a portal with 6 inch spacing between products, while the carrying conveyor moves at a speed of 540 to 600 feet per minute. In such applications, many parameters interact to determine system's performance (e.g., the conveyor belt speed, the length of the active reading zone, the spacing between tags, etc.). Despite the problem's importance, however, not many results on relating the above parameters to the obtained system's performance appear to exist.

Of the available results, we mention that [56] conducted several tests on this topic. The authors describe the tests done, however, to be less repeatable than other (non conveyor belt) tests. The difficulty in repeating the tests arises since belt mobility introduces variables that are difficult to constrain (e.g., products may shift in cases over time, or differ slightly from case to case). In addition, the repetitions necessary for statistical confidence are found to be cost prohibitive. The lack of adequate analytical results provides us with a motivation to investigate the problem further.

The second part of the thesis (Chapters 3 and 4) concerns UWSNs. The class of UWSNs has received attention recently (see, e.g., [23, 39]) for their potential applications in oceanographic scientific explorations, military surveillance tasks, and industrial applications. For some of such applications, it is economical and beneficial to deploy mobile nodes that move with water currents during a monitoring mission of several days. The nodes are collected at the end of a mission.

Such mobile deployments have been used, for example, in the experimental work reported in [12]. When nodes are required to communicate with each other to form a network, connectivity of the network can be disrupted due to possible large movement of such free floating nodes in open water spaces. To investigate the resulting effects, the work of [17] has adopted a kinematic model for node mobility. Using the adopted model, the authors use simulation to analyze important performance measures such as the fraction of the total covered area, and the area covered by the largest connected component of the network, during a given interval of time. In [43, 42] the authors use a probabilistic mobility model to develop algorithms for estimating the probability that a network is wholly or partially connected

In Chapter 5, we tackle the following problems:

- **P1:** The problem of estimating the likelihood that two given nodes in an UWSN can reach each other with a path of length at most a given constant D.

- **P2:** Extension of the above problem to the case where each link has an associated delay, and we want to estimate the likelihood the end-to-end delay is at most a given constant D.

Both problems are motivated by applications that require bounded communication delays. To cope with the above problems, effective algorithms need to be developed. Effectiveness of an algorithm is measured both by its running time, and the accuracy of the obtained results. The thesis pursues this direction in Chapter 5.

1.3 Thesis Organization and Contribution

The organization and contributions of the thesis are as follows.

- In Chapter 2, we give a broad overview of RFID systems with emphasis on basic definitions and classifications, anti-collision protocols, important aspects of the EPCglobal Class 1 Gen 2 standard, and a brief look at some of the experimental results on measuring performance of RFID systems.

- In Chapter 3, we tackle the problem of identifying objects placed on a moving conveyor belt. We adopt a framed slotted-Aloha based anti-collision approach. In such approach, each tag utilizes a slot counter that is loaded with an integer value in the range $[0, K-1]$ (or, $[1, K]$) where the setting of the parameter K determines the performance of the system. We devise two identification schemes based on using the slotted-Aloha approach to handle the problem. We also derive bounds on the parameter K that enable each scheme to achieve good performance. Our analysis considers scenarios where consecutive tags passing through the interrogation zone of a reader have consecutive slot counter values

(modulo K). To the best of our knowledge, the work in Chapter 3 is novel.

- In Chapter 4, we give pointers to some networking research work done on UWSNs. The topics discussed include medium access control (MAC) protocols, routing algorithms, connectivity and coverage algorithms, and localization methods. We also review some work done on mobility modelling in scenarios where uncontrollable node mobility is of prime concern.

- In Chapter 5, we tackle the problems P1 and P2 above. We adopt a probabilistic mobility model similar to the model used in [43]. Using an approximation of the subgraph connecting the two specified end nodes using a set of node-disjoint paths, we obtain an efficient algorithm to compute lower bounds on the exact solution. We note that the definition of such a set of node-disjoint paths takes into consideration probabilistic node connectivity. To the best of our knowledge, the work in Chapter 5 is novel.

1.4 Summary

Support of mobility is an important requirement for the majority of information processing and networking systems being used and sought nowadays. Currently, a vast amount of research work exists on mobility support in various important classes of wireless networks: e.g., MANETs, cellular networks, WSNs, UWSNs, and RFID systems. In this chapter, we have identified two types of problems on RFID systems and UWSNs that are worthwhile investigating in the rest of the thesis.

Chapter 2

Literature Review on RFID Systems

In this chapter, we give background information on RFID systems to set a context for the problem dealt with in the next chapter. The main topics discussed in this chapter include RFID systems characteristics and classification, current networking research issues in the field, an overview of basic results in the area of RFID anti-collision protocols, highlights of the anti-collision protocol used in the EPCglobal Class 1 Generation 2 protocol, and summary of some experimental research work in the area.

2.1 Introduction

The term RFID system is used to refer to a number of information and communication technologies that provide means of automatic identification of objects, locations, and individuals to computing systems. The core components of any such system are called *readers* (or *interrogators*), and *tags* (or *transponders*). Readers can perform both reading and/or writing operations to tags, while tags are identification information storing devices.

RFID systems discussed in the literature work according to different physical principles, engineered using multiple architectures, and manufactured using divers technologies. Nevertheless, they share the following common aspects: the energy required for the operation of a tag is transmitted by a reader wire-

lessly, and the tag employs a mechanism to communicate with the reader by modulating the reader's transmitted signal (rather than generating its own signal).

In general, RFID technologies enable reading and/or writing of tag information with no line of sight between the two devices. Thus, providing an important improvement over the ubiquitous barcode technology. Additionally, many RFID tag technologies allow storage, retrieval, and modification of substantially larger volumes of data than barcode information.

RFID systems are widely used in various applications including supply chain management, health administration, animal and human tagging, electronic payment, and asset management. The US Department of Defence (DoD) was among the first to embed RFID chips on containers for supply chain management purposes in 1990s [30]. Other examples are Wal-Mart in the United States, Tesco in the United Kingdom, and Metro in Germany which they all use RFID technology to track their products [30].

RFID systems have also gained popularity in the health industry where tags are employed to track assets in healthcare applications for different purposes including the control of bio-samples [59]. Another wide spread use of RFID tags is in the financial sector where RFID-based micropayment are made possible by storing electronic cash information on RFID chips attached to credit cards [59].

RFID systems are classified according to a number of properties. The most important property is the *coupling* method between readers and tags. The coupling method determines the practical radio frequency range that can be used in the system, and the maximum possible separation distance between readers and tags. Thus, by implication, the coupling type method determines the type of applications most suitable for such systems. Currently, inductive coupling and capacitive coupling dominate the RFID literature.

Inductive Coupling: RFID systems in this category utilize the magnetic field generated by the electromagnetic wave emitted by a reader. Thus, power transmission from a reader to a tag utilizes magnetic induction. Tags communicate by changing the load resistance connected to their antenna in a process called *load modulation*. Note that inductive coupling works if a tag is in close proximity of a reader. This mechanism allows the realization of the following types of RFID systems:

- Close-coupling systems that work for up to 1 cm. Here, tags are either inserted into the reader, or placed on a reader's surface. Such systems are primarily used in security applications such as electronic door locking systems.

- Remote-coupling systems that provide read and write ranges of up to 1 meter. Applications of such systems include contactless smart cards, animal identification, and industrial automation.

Inductive coupling systems operating in the high frequency (HF) band (between 3 MHz and 30 MHz) are commercially available.

Examples of standards that deal with this type of RFID systems include the ISO 14443 and ISO 15693 (see,[59], Chapter 2). Both standards concern systems operating in the 13.56 MHz frequency. The ISO 14443 specifies RFID proximity tags (e.g., for ticketing applications) that have a typical range of a few dozen centimeters. The ISO 15693 standard specifies RFID vicinity tags (e.g., tags attached to luggage) that have a typical range between 1 and 1.5 meters.

Capacitive Coupling: RFID systems in this category utilize the electromagnetic field emitted by the reader. Tags communicate by reflecting back a small part of the radio wave emitted by the reader in a process called *backscattering modulation*. Compared to inductive coupling, capacitive coupling allows greater distances between tags and readers. Therefore, the mechanism allows the realization of the following types of RFID systems:

- Remote-coupling systems (up to 1 meter separation distances), as explained above.

- Long-range systems that provide separation distances of 3 meters using passive backscatter tags, and 15 meters and above using active backscatter tags.

Capacitive coupling systems utilizes either an ultra high frequency (UHF) band (between 300 MHz and 3 GHz), or a microwave ($>$ 3 GHz) band.

Examples of standards that deal with this class of RFID systems include the EPCglobal Class 1 Generation 2 standard (see,[59], Chapter 3). The standard concerns systems operating in the UHF band from 860 MHz to 960 MHz. Typical uses include supply-chain applications (e.g., item tracking in distribution centres and warehouses) where a reader manages a tag population through selection, inventory, and access operations. We provide more information on this standard in Section 2.3. Tags operating in this class of systems that are equipped with RISC processors and non-volatile memory exist commercially.

RFID systems can also be classified according to the following properties:
Passive versus Active Tags: A passive tag has no source of power supply except the magnetic or electromagnetic field received by the tag's antenna. A semi-passive tag has a battery that supplies power to the tag's chip, but not for data transmission. An active tag has a battery to power both its chip and data transmission. In general, RFID systems have gained popularity because of the low manufacturing cost of passive tags.
Tag Information Storage and Processing Capabilities: The memory capacity of a tag can range from a single bit to a few Kbits of static random access memory or non-volatile memory. 1-bit tags are used extensively for Electronic Article Surveillance (EAS) applications, e.g., to detect the presence or absence of a tag passing though a security gate. Tags with higher memory capacity can store more information, e.g., a 96-bit object Electronic Product Code (EPC).

2.2 RFID Medium Access Control Protocols

A typical UHF RFID system contains one, or more, readers that perform various operations on a possibly large number of tags. When communications share a given wireless band, contention between different transmissions occurs. Thus, as with other types of wireless networks, designing efficient medium access control (MAC) protocols arise as an important topic. In this section, we give a brief overview of the work done in this area. Our presentation follows the surveys of [4], [44], and [75]. To start, we note that collisions in RFID systems are typically classified as either tag collisions, or reader collisions, as explained below.

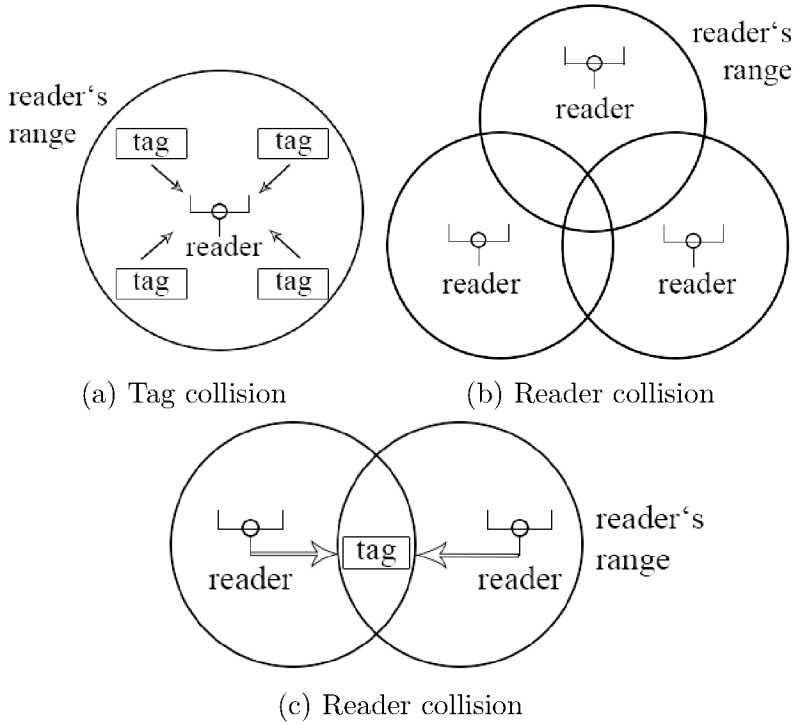
- **Tag collisions** (includes *tag-to-reader* collisions): This type of collisions occurs when multiple tags respond simultaneously to a reader's initiated operation. Such collisions cause different information transmitted by tags to be lost. MAC protocols that aim at resolving such collisions are known as anti-collision protocols.

- **Reader collisions** (includes *reader-to-tag* collisions and *reader-to-reader* collisions): This type of collisions occurs when two, or more, readers transmit at the same time. In reader-to-tag collisions, a tag receives transmissions from two different readers; this causes the two transmissions to be corrupted. In reader-to-reader collisions, two transmitting readers are within the interrogation range of each other, and one (or both) may lose information that may be propagating from a tag to that particular reader.

Below, we give more details on the class of tag anti-collision protocols.

2.2.1 Tag Anti-Collision Protocols

Protocols to resolve collisions among different tags responding to a reader typically use Time Division Multiple Access (TDMA) that are driven by the readers (also called reader-talk first protocols). These protocols are broadly



(a) Tag collision (b) Reader collision

(c) Reader collision

Figure 2.1: RFID collision types

classified as either *tree* based protocols, *Aloha* based protocols, or *hybrid* protocols.

We now discuss the above protocols in the context where the reader seeks to identify all tags present in the interrogation zone (i.e., read some label stored in each tag) where the tag population size is not known in advance.

Tree Based Protocols: The use of a tree based algorithm to deal with a multi-access contention problem dates back to the work of [16]. The work of [41] is an early work that applied such ideas to the RFID tag collision problem. Subsequent work in this direction has resulted in a large number of approaches. The survey work in [44] classifies the approaches into the following classes.

- **Tree Splitting Algorithms** (about 3 approaches are surveyed in [44]). Algorithms in this class operate by splitting responding tags into multiple subsets using a random number generator. The subsets become increasingly smaller until each subset contains one tag. Each tag maintains a counter to record its position in the resulting tree. An example

of such an algorithm is the work in [48].

- **Query Tree Algorithms** (about 6 approaches are surveyed in [44]). Algorithms in this class do not require tags to use a random number generator or use counters. Instead, they rely on storing tree construction information at the reader, and tags only need to have a prefix matching circuit. An example of a query tree algorithm is the work done in [49].

- **Binary Search Algorithms** (about 4 approaches are surveyed in [44]). Algorithms in this class involve the reader transmitting a binary string which the tags receive and compare against their stored IDs. Tags with ID equal to or less than the received binary string respond. The reader monitors tags reply bit by bit and splits tags into subsets on collided bits. The *Enhanced BS Algorithm* (EBSA) of [68] is an example algorithm in this class.

- **Bitwise Arbitration Algorithms** (about 5 approaches are surveyed in [44]). Algorithms in this class operate by requesting tags to respond bit by bit (from the most significant to the least significant bit) of their ID. Such protocols work in RFID systems where tag responses are synchronized at the bit level. In such systems, multiple tag responses carrying the same value of some bit allow the reader to recover this shared value. On the other hand, multiple tag responses carrying different values of some bit allow the reader to detect collision on that particular bit. The work done in [29] is an example of such an algorithm.

Aloha Based Protocols: Anti-collision protocols in this category include Slotted Aloha (SA) protocols and Framed Slotted Aloha (FSA) protocols (see, e.g. [30, 4, 44, 75]).

Slotted Aloha (SA) anti-collision protocols assume that channel time is divided into fixed length slots, each slot is long enough for a tag to send the requested information (e.g., a tag ID). In RFID systems, reader-controlled syn-

chronization establishes slot boundaries. A collision occurs at slot boundary. On collision, tags retransmit after a random delay.

Several variants of the basic SA protocol has been investigated (see, e.g.,[44]). The variants include the use of the following ideas:

- SA with muting: here, a tag is silenced after its successful identification, and
- SA with early end: if no transmission is detected at the beginning of a slot (i.e., and idle slot is detected), the reader closes the slot early, and starts a new slot.

In a basic Framed Slotted Aloha (FSA) an inventory interrogation cycle is composed of one, or more, reading rounds. A reading round uses a frame with a specified number of slots. At the beginning of each round, each tag chooses a slot at random to transmit its ID. Each tag is allowed one transmission attempt in each frame. Variants of the basic FSA protocol with tag muting after a successful read, and early end of idle slots have been discussed in the literature (see, e.g., references in [44]). Examples of work on FSA protocols include [9], and [64].

The class of Dynamic Framed Slotted Aloha (DFSA) protocols is an important variant of FSA protocols. Here, a reader potentially uses different frame sizes in different reading rounds within an interrogation cycle. Changes in the frame size are made to better serve the decreasing population of unread tags after each reading round. DFSA protocols typically use a method to estimate the number of tags to be identified. Examples of such methods include the work in [63, 9, 31, 21].

Hybrid Protocols: Aloha based protocols are faster than tree based protocols. However, Aloha based protocols suffer from tag starvation syndrome. Hybrid protocols combine tree based and Aloha based algorithms. For example, a QT protocol can be used to divide tags into smaller groups. Each group

can then be read using a FSA protocol. Examples of hybrid protocols include the work of [11, 60, 50, 70].

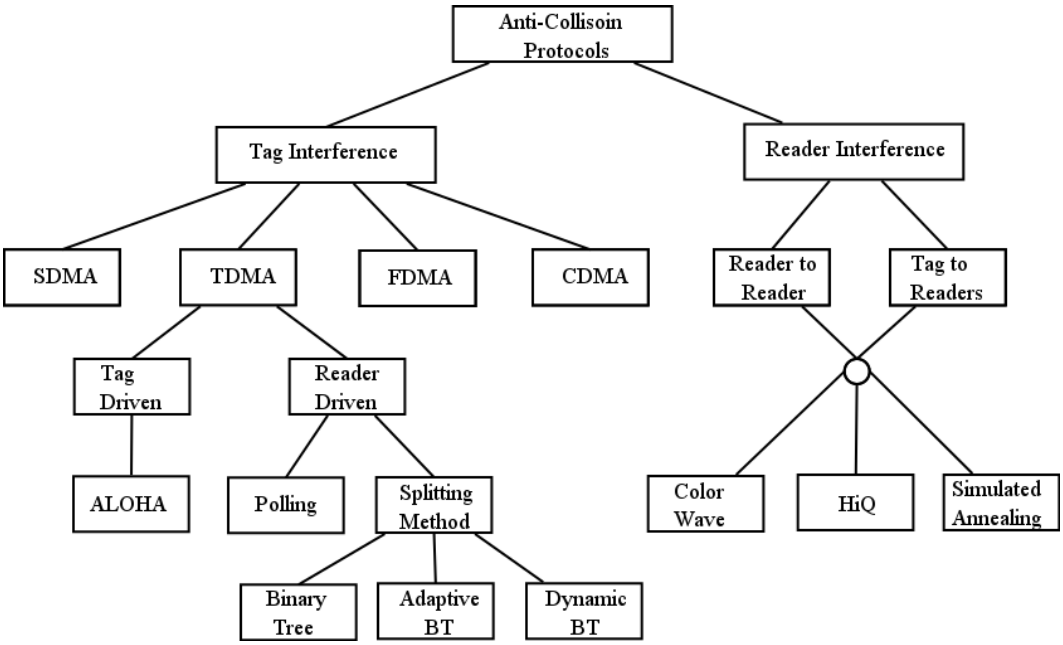


Figure 2.2: Anti-collision protocols (adapted from [61])

2.3 Highlights of the EPCglobal Class 1 Generation 2 Standard

In this section, we review some basic aspects of the EPCglobal Class 1 Generation 2 (abbreviated Gen 2 in this section) standard with emphasis on its anti-collision protocol (the Q protocol) [1]. Gen 2 concerns UHF readers (*interrogators*) and passive tags operating in the 860 MHz - 960 MHz frequency range, and provides packetized, reader-talks-first protocol.

In general, the standard defines **(a)** the physical interaction (signalling layer), and **(b)** logical operating procedures, and low level commands between readers and tags. Next, we present the following protocol aspects, following the explanation in [1] and ([25], Chapter 8).

2.3.1 Power-up Sequence and Tag States

Tags within a reader's reading zone are powered up by receiving a continuous wave (CW) transmitted by a reader for at least 2500 μs. When powered up, a tag is in the *Ready* state. Other states a tag can be in include: *Arbitrate*, *Reply*, *Acknowledged*, *Open*, *Secured*, and *Killed*. Note that a power is lost if a reader hops to another frequency, or a tag passes through a fading zone.

2.3.2 Forward Channel (Reader-to-Tag) Symbols and Data Rates

Binary data transmitted by readers are pulse-interval-encoded (PIE) symbols, and each symbol encodes one bit as follows:

- Binary '0': consists of a power-on interval followed by a power-off interval of equal length. The total length is a reference interval, denoted *Tari*. The pulse width (PW) is the length of the power-on (or power-off) interval. So, PW = $\frac{Tari}{2}$.
- Binary '1': consists of a power-on interval of length 2 to 3 times longer than the power-off interval.

Typical values of *Tari* are 6.25, 12.5, and 25 μs corresponding to binary 0 symbol rates of 160, 80, and 40 Kbps, respectively. The reading timing information (such as the *Tari* parameter) are communicated to tags in a *preamble* (or *frame sync*) field transmitted as part of issuing a command (such as the Query command). In particular, the reader sends binary '0' (whose length is *Tari*), and a special reader-to-tag calibration symbol, denoted *RTcal*, whose length is binary '1'.

2.3.3 Reverse Channel (Tag-to-Reader) Symbols and Data Rates

Tags support 4 symbols encoding modes, denoted FM0, Miller-2, Miller-4, and Miller-6. In each mode, a symbol encodes one bit, and 0s and 1s have equal length.

- **FM0:** The length of each symbol is denoted *Tpri*. The backscatter line frequency (BLF) is defined as BLF = $\frac{1}{Tpri}$.

- **Miller-Modulated-Subcarrier (MMS) with index M = 2,4, or 6:** These encodings utilize longer time periods of each symbol. For a given Miller index M, the length of a symbol is M times longer than that of FM0 (so, the resulting data rate is BLF/M).

Commands such as the Query command are used to set the *Tpri* parameter, and also the backscatter line frequency (BLF) used by a tag. This is done as follows:

- The preamble part of the Query command contains a special tag-to-reader calibration symbol, denoted *TRcal*, whose length is measured by tags.

- In addition, the preamble specifies two other parameters, called the divide-ratio (DR) that can either take the value 8 or 64/3, and the Miller index M.

- The length of the *TRcal* symbol, and the values of parameters DR and M, define the tag-to-reader used data rate.

2.3.4 Stored Tag Flags

Each tag keeps 5 flags, and each flag takes one of two possible values: either A or B. All flags are initialized to A when tags are powered up, and readers have the ability to access and test each flag. The function and persistence of each flag is described below.

1. **The Select (SL) flag**. This flag facilitates selecting a subset of tags for further access or participation in an inventory operation.

2. **Session flags (S0 through S3)**. Taking inventory of a given tag population is an important operation in RFID systems. Gen 2 supports quasi-simultaneous inventory to be performed by different readers. The

state of an inventory operation taken by each reader constitutes a *session*. Four such sessions are supported by the standard, and each one of them utilizes a dedicated flag (S0 through S3). Persistence of the flags when tag power is on, or off differ as follows.

- S0 keeps its state while power is on, but loses value when power is lost.
- S1 keeps its state for an interval between 500 ms, and 5 sec when power is either on or off.
- S2, S3, and SL keep their state when power is on, and for at least 2 sec if power is lost.

2.3.5 Logical Operations and Commands

Readers manage tag populations using three basic operations:

- **Select:** reader chooses a subset of tags for subsequent inventory or access operations
- **Inventory:** reader retrieves unique identifiers associated with a subset of selected tags
- **Access:** reader performs core operations such as reading, writing, locking, and killing a tag

Each such operation is done by issuing one or more of several basic commands to the tags, e.g., Select, Query, QueryRep, QueryAdjust, ACK, NAK, etc. Table 2.1 illustrates the mandatory commands in each operation.

2.3.6 Medium Access Control (the Q Protocol)

Gen 2 utilizes a varient of the Slotted-Aloha medium access control (MAC) protocol to perform inventory operations. The algorithm utilizes a parameter, denoted Q that defines the range $[0, 2^Q - 1]$ from which a contending tag chooses a random slot number. The chosen number is stored in a specific *slot-counter* in a tag. The inventory operation is done in rounds, each of which has 2^Q slots. The Q protocol proceeds as follows.

Round	Command	Description
Select	Select	allows a reader to select a tag subpopulation based on a user-defined criteria
Select	Challenge	allows a reader to advise tag(s) to precompute and store a cryptographic value(s) for authentication purposes
Inventory	Query	allows a reader to set up an inventory round by setting up a session parameters
Inventory	QueryAdjust	allows a reader to change the Q value, thus, the number of slots in an inventory round without changing any other parameters
Inventory	QueryRep	allows a reader to instruct tags to decrement their slot counter by 1, change the inventoried flag value of identified tag, and ends the current slot
Inventory	ACK	allows a reader to acknowledge a single tag
Inventory	NAK	allows a reader to send a tag back to *Arbitrate* state due to a communication error. Tags in *Ready* and *Killed* states ignore this command
Access	Req_RN	allows a reader to advise a tag to backscatter a new RN16 (denoted handle)
Access	Read	allows a reader to read tag's memories
Access	Write	allows a reader to write a word in tag's memories
Access	Kill	allows a reader to permanently disable a tag
Access	Lock	allows a reader to lock various parameters and tag's memories to prevent future modifications

Table 2.1: EPCgloabl Gen 2 anti-collision protocol mandatory commands (adapted from [1])

1. The reader performing an inventory operation issues a Select command to prepare a subset of tags for participation in the inventory operation.

2. The Select command specifies a matching criterion. This criterion can involve values of the 5 tag flags, and the contents of tag memories. In addition, the command specifies updated values for the matching tags, and possibly different updated values for the non-matching tags who heard the command. The Select command carriers a CRC-16 word. (Note: the time between the Select command and the next command constitutes a command gap. Such a gap is filled with a continues wave (CW) for at least a specified period of time)

3. The reader then issues a Query command. The preamble of the command

specifies tag-to-reader timing information (e.g., *TRcal* symbol, the divide ratio, and Miller index M). In addition, the command specifies the state of the Select flag for the selected tags participating in the inventory operation, the session number to be used, and the state of the session flag for participating tags, the value of the Q parameter, $Q \in [0, 15]$, and a 5-bit CRC word.

4. Each of the selected tags generates a random number in the range $[0, 2^Q - 1]$ and stores the number in the tag's slot-counter.

5. A tag whose slot-counter = 0 generates a 16-bit random number, denoted RN16. The number is backscattered to the reader with no error-checking (i.e., no parity bits or CRC)

   (a) If there is no collision, the reader sends an ACK command that includes the received sequence (which the reader thinks it received from the responding tag). ACK commands have no error checking bits.

   (b) The tag verifies the received sequence. If correct, the tag responds by sending the Protocol Control (PC) bits, its stored EPC, and a computed CRC16 bits (such sequence is denoted PC+EPC+CRC16). The PC stores the length of the EPC as well as some optional information about the tag. Then, the tag moves to the Acknowledge state. On the other hand, if verification of the received RN16 fails, the tag does not respond.

   (c) If the PC+EPC+CRC16 is garbled, the reader sends a NAK command. The session's flag is not flipped. The tag does not choose a new slot value, and the tag waits until the next inventory round.

6. To continue the inventory round, the reader sends a QueryRep command. The command signals the end of a slot, and the tag that has just sent its EPC (now in the Acknowledge state) flips the flag corresponding to the session. Other participating tags decrement their slot counters.

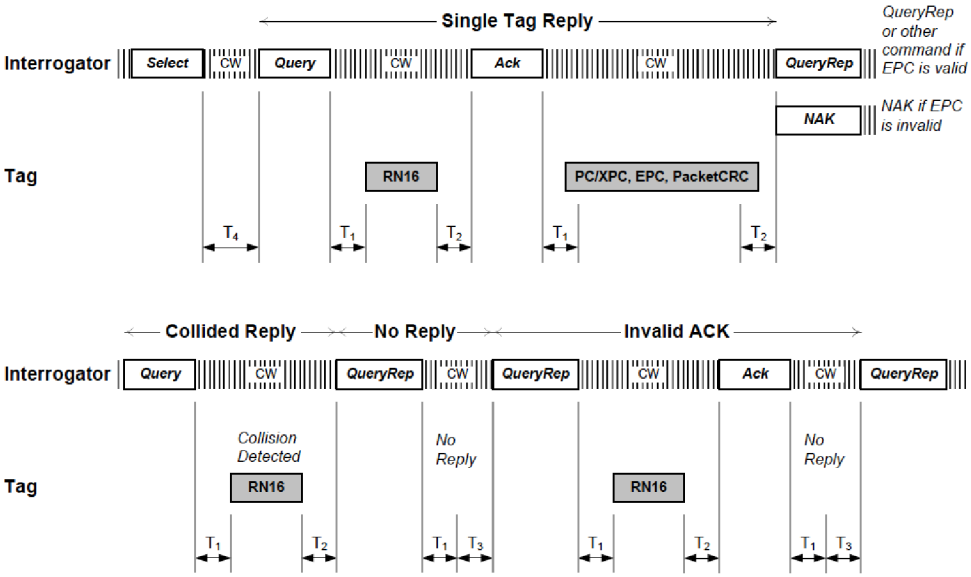


Figure 2.3: Some steps of the Q-protocol (adapted from [1])

7. If collision occurs, the reader is likely to receive the preamble backscattered by the responding tags. (All such preambles are identical and start at the same time.) The received random numbers, however, will be garbled. The reader can distinguish between a collision and no response (i.e., an empty slot). The reader issues a QueryRep to advance the inventory round to the next slot. The protocol goes to Step 5.

Figure 2.3 shows a communication process between a reader and tags under the Q protocol. In the figure, T_1 is the time from reader transmission to tag response, T_2 is the time from tag response to reader transmission, T_3 is the time reader waits, after T_1, before issuing another command, and T_4 refers to the minimum time between reader's commands.

2.4 Experimental Work on RFID Performance

In this section, we present some results of [5, 56, 14] on real life experimental evaluation of RFID systems. In particular, the authors of the above mentioned work noted that many timing and performance evaluation figures of RFID systems are not obtainable from manufactures' specification. This may be the results of the complexity of the underlying RFID systems and the use of

proprietary command sequences by a reader to perform high level operations such as inventory.

The Work of [56, 5]. In [56], the authors focused on the design of a set of benchmark experiments to measure the following (real life) aspects on UHF RFID tags.

- Effect of the reader-tag separation distance on the *response rate* of a given brand and model of tags. The response rate is defined as the ratio of successful tag reads per the number of read attempts.
- Sensitivity of tag reading to the orientation of tag's antenna surface with respect to the reader's antenna surface.
- Variance of tag performance among tags of the same model (e.g., using the response rate measure defined above).
- Evaluation of tag *read time*, and *time to first read*, when tags are processed in isolation, and in population.

The work in [56] explores, also, identifying objects moving on conveyor belts. (According to [56], part of the RFID mandates the ability to read products passing through a portal with 6 inch spacing between products, and the conveyor moving at speed of 540 to 600 feet per minute.) The testing in [56], however, used a lower speed of 200 feet per minute.

The experiments set up a portal across a conveyor section using three reader antennas (two on the sides and one above). Tags with a single dipole antenna, and dual dipole antennas, are used in the experiments. The tags are spaced about 2 feet apart. The products carrying the tags vary in both the containers' material (e.g., paper, foil-lined boxes, metal cans, and polypropylene containers) and contents (e.g., paper, water, soup, and other liquids). Results on the product read rate (the fraction of times a product is successfully read) are reported. Among other results, it has been found that the following combinations performed poorly: dipole antennas on foil-lined detergent containers,

and dual dipole on canned soup containers (that contains metal cylinders).

In a similar vein, the work of [5] conducts experimental work on commercially available ISO 18000-6c UHF RFID systems. The experiments use tags of different sizes (a large size implies a large antenna): 4×4 in, 1×4 in, 0.5×4 in, and 1.5×1.5 in. Measurements of read speeds when tags are placed near metal (e.g., directly attached on metal plate, or separated from a plate by cardboards of different thicknesses), and near water (e.g., directly attached to polypropylene containers filled with water) are reported.

The Work of [14]. Work in [14] concerns performance of EPC Class 1 Gen 2 systems. The work is motivated by the importance of such systems, the lack of tools for studying RFID protocols, and the finding that vendor white papers provide limited information that cannot be interpreted without details of the reader configuration. To enable a deeper investigation, the authors designed a monitoring platform that helps in reverse-engineering of some aspects of a protocol's implementation.

The platform utilizes a Universal Software Radio Peripheral (USRP), and GNURadio to monitor reader transmissions. As well, RFID readers that provide detailed results of each read cycle (e.g., inventory results for each read cycle) are used. The experimental setup utilizes a poster board with 16 tags arranged in a 4x4 grid where tags are spaced approximately 6 inches apart. Different experiments are conducted in two rooms of different dimensions while changing the reader to grid distance and antenna orientation. The two operations of tag selection and inventory are used. Measurements on performance measures such as single tag read rate, tag set read rate, inventory cycle duration, as well as hidden protocol values such as inter-cycle duration, are observed when a reader operates in one of three modes: high speed, standard, and dense reader mode. The obtained results are interpreted in light of the protocol details inferred by the monitoring platform.

2.5 Concluding Remarks

RFID systems are complex information and communication systems. They have gained significant attention in recent years due to their many important applications, and the low cost of manufacturing passive tags. The overview given in this chapter starts by presenting some fundamental concepts of such systems. Next, we have discussed some research results on anti-collision protocols, and highlighted some aspects of the EPCglobal Class 1 Gen 2 standard, as one of the important standards in the field. It has been noted in the literature that vendor white papers provide limited information that cannot be interpreted without details of the reader configuration. To gain more insight into the performance of UHF RFID systems, some experimental results have been reported in the literature. Our overview includes some information in this direction as well.

Chapter 3

Identification of Mobile Tags on Conveyor Belts

In this chapter we consider an RFID application where a continuous stream of objects is placed on a moving conveyor belt. Object information is retrieved by reading a key that can be stored in an RFID tag attached to an object. We wish to identify as many of the key tags as possible. To this end, we propose schemes and protocols that utilize low cost passive RFID tags to approach the problem. We identify important parameters that affect the performance of the devised schemes. We also develop relations among such parameters that are useful in analyzing performance. In addition, we also investigate the performance by simulation, present the obtained results, and draw some conclusions.

3.1 Introduction

In this section, we motivate research work on developing effective anti-collision schemes for solving the problem of identifying tags moving on a conveyor belt. To start, we recall from previous chapters that RFID mandates the ability to read tags placed on a conveyor belt moving at a high speed (e.g., 540 to 600 feet/minute) [56]. The work in [56] has conducted experiments to assess the obtained performance of using UHF RFID in such applications. Among the obtained findings, the authors have found that due to the many parameters that are hard to constrain in such experiments, there is a difficulty in repeating

the experiments. In addition, repetition of experiments to obtain statistical confidence in the obtained numerical results are found to be cost prohibitive.

We also recall that many researchers have reported on the difficulty of obtaining good characterization of UHF RFID performance in even less challenging environments. For example, in [14], the authors have found that vendor white papers provide limited information that cannot be interpreted without details of the reader configuration. In a similar vein, in [30], Section 7.2, the authors state that for reasons of competition, system manufacturers are not generally prepared to publish anti-collision procedures that they use.

The above findings motivate our work in this chapter on

1. identifying a subset of parameters (e.g., conveyor belt speed and inter-tag distances) that affect the identification process,
2. developing schemes for identifying a potentially infinite sequence of tags moving on a conveyor belt,
3. analyzing the schemes to develop recommendations on how to set the underlying design parameters so as to avoid degraded performance, and
4. conducting simulation experiments to investigate some performance aspects of the developed schemes.

Our work here relies on the use of RFID readers and passive tags only. We do not assume the use of sensor devices to detect the beginning or end of the containers placed on the conveyor belt. As well, we do not assume that the conveyor belt can be slowed down, stopped, or temporarily reverse its direction during the identification process.

3.2 System Model

We consider tags placed on a conveyor belt that moves at a constant speed, denoted s_{conv} , and tags are spaced apart from each other with a distance, denoted d_{inter}. Tag identification is done by placing one, or more, *identification*

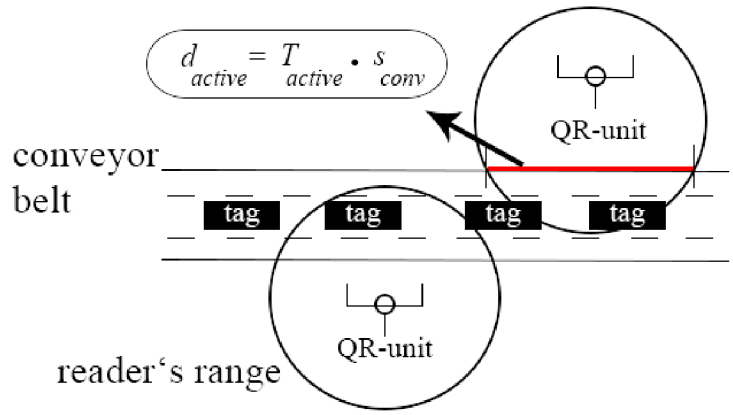
station along the belt. Each station may utilize one, or more, RFID reader(s). Multiple stations can then be placed along the belt to improve the reading accuracy. We approach the problem by proposing and investigating two schemes that utilize one, or two, reader(s) per identification station, as explained below.

The two devised schemes utilize the following common concepts.

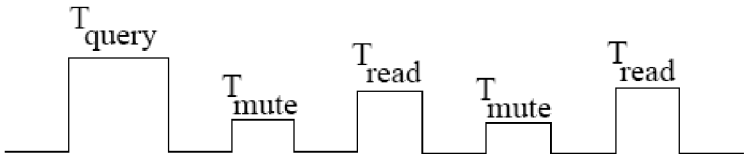
- Each scheme installs one, or two, RFID readers in close proximity of the conveyor belt. Depending on the location of the reader with respect to the belt, and the curvature of the belt in the neighbourhood of the reader, certain region of the belt becomes the interrogation zone of the reader. We denote the length of such an active region by d_{active}.
- Tags passing through the active region of the belt respond to query messages of the reader. Upon reading a response from a tag, the reader sends a mute command to that particular tag. We denote by T_{read} and T_{mute} the times allocated by the reader to read and mute the tags.
- Multiple tags passing through the active region of the belt may respond simultaneously to a query message of the reader. An anti-collision mechanism needs to be employed for effective operation. Similar to the Gen 2 standard, we choose to use a framed slotted Aloha based mechanism as our main methodology. In particular, we assume that each tag stores a slot number, denoted k, from a set of integers of cardinality, denoted K. In addition, each tag has a slot counter.
- Upon receiving a query message, each tag loads its slot counter with the stored slot number k.
- The slot counter is decremented by 1 when a tag receives a *read* command.

3.2.1 Single Reader (S-READER) Scheme

In this scheme, each station uses a single *query-response reader unit* (QR-unit), as illustrated in Figure 3.1a.



(a) Conveyor belt environment



(b) Reader T_{cycle}

Figure 3.1: Layouts for the S-READER scheme

The placement of the QR-unit at a given distance from the belt determines the length d_{active} of the interrogation zone.

The S-READER scheme operates under the following protocol:

1. Time is divided into fixed length cycles of length T_{cycle} each (see, Figure 3.1b).

2. At the beginning of each cycle, the reader issues a query command of length denoted T_{query}. The value of T_{query} is determined by the used RFID technology.

3. The remaining T_{cycle} time is subdivided into equal length *read-mute* periods. Each such period is of length $T_{read} + T_{mute}$ where T_{read} is the time the reader awaits a response from a tag. Upon a successful tag read operation, the reader sends a mute command (in T_{mute}) with the successfully read tag identifier.

4. As mentioned above, upon hearing the query command, each tag loads it slot counter with a pre-stored number k. The slot counter is decremented when a tag hears a read command. A tag sends its ID when the slot counter reaches zero. In this scheme, $k \in [0, K-1]$. So, tags with $k = 0$ send their IDs in the slot immediately following the query command.

5. A muted tag does not respond to subsequent queries by other identification stations that may be installed along the belt.

Interest in analyzing the S-READER scheme arises as follows.

- As can be seen, various parameters affect the probability of a successful identification of a tag: e.g., the conveyor belt speed s_{conv}, the distribution of tag inter-distances d_{inter}, the length of d_{active}, the length of T_{cycle}, and the total number K of slots per cycle.

- If we place a reader close to the belt, d_{active} becomes relatively large, and many tags can hear and respond to query commands; this may give rise to many collisions. In contrast, if we place a reader relatively far from the belt, d_{active} becomes relatively small and many tags can miss responding while traversing d_{active}.

- Setting K (the total number of slots per cycle) to a small value may increase the collision probability, whereas setting K to a large value makes the tags less responsive as they pass through d_{active}.

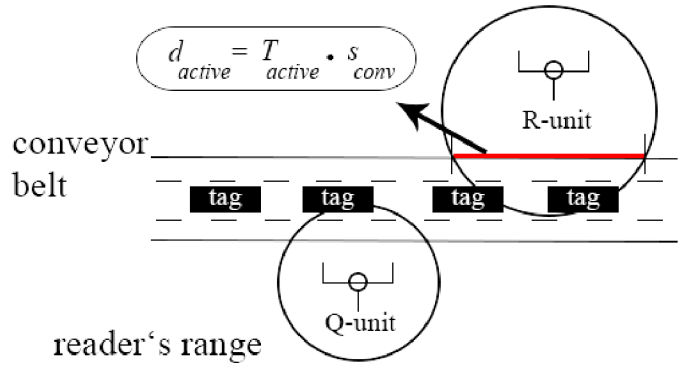
Thus, it is of interest to analyze the interaction among the above parameters of the system.

3.2.2 A Dual Reader (D-READER) Scheme

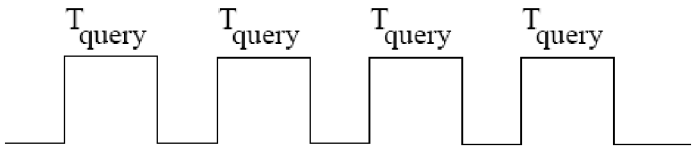
In this scheme, each identification station utilizes a pair of readers: a *query unit* (Q-unit), and a *response reading unit* (R-unit). The Q-unit cyclically broadcasts query commands, and the R-unit cyclically performs read-mute operations. Thus, a tag leaves the interrogation zone of the Q-unit with its

slot counter loaded with the tag's stored slot number k. When it enters the interrogation zone of the R-unit, and hears a read command it decrements the slot counter by one.

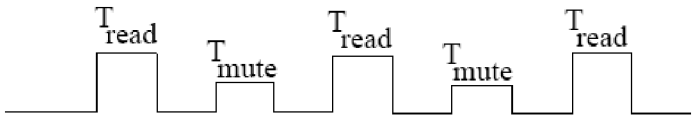
The value of the slot counter is assumed to persist in the interval between the two interrogation zones. Subject to the maximum possible persistence period of the slot counter, the separation distance between the Q-unit and the R-unit should be set sufficiently large to minimize the number of tags that can simultaneously hear both query commands, and the read-mute commands.



(a) Conveyor belt environment



(b) Q-unit T_{cycle}



(c) R-unit T_{cycle}

Figure 3.2: Layouts for the D-READER scheme

The D-READER scheme aims at achieving improved performance over the S-READER scheme. The main insights are as follows. In the S-READER

scheme, a tag spends part of the T_{active} ($= \frac{d_{active}}{s_{conv}}$) waiting for a query message before responding. Such a waiting time can be viewed as a wasted time that can be better utilized. In contrast the D-READER scheme makes an arrangement to eliminate such waiting time while ensuring that each tag hears at least one query command.

The D-READER scheme operates under the following protocol:

1. The Q-unit cyclically broadcasts query commands. The length of the interrogation zone and the frequency of broadcasting query commands should ensure that each tag hears at least one query command (and, consequently, loads its slot counter).

2. Upon receiving a query command, a tag loads its slot counter with a prestored number $k \in [1, K]$. Note that, in contrast with the S-READER scheme, $k = 1$ is the minimum slot counter value in this scheme.

3. Upon receiving a read command in the R-unit interrogation zone, if the tag is unmuted, it decrements the slot counter by 1. A tag with slot counter equals zero sends its ID. If the ID is received correctly, the reader mutes the tag.

Interest in analyzing the interaction between the various parameters here follows the same argument as in the previous section.

3.3 Useful Relations for the Single Reader Scheme

In this section, we derive some relations that we expect to be useful in analyzing the S-READER scheme. To simplify the analysis, we assume that consecutive tags are assigned consecutive slot numbers (0,1, ..., $K - 1$). The first read station on the conveyor belt encounters tags in that order. Due to tag muting, however, subsequent stations (if any) do not see such sequential numbers. Therefore, we focus on the first station.

- **Feasibility.** The speed of the conveyor belt s_{conv} should allow at least one tag to be processed. That is,

$$T_{active} \geq T_{query} + T_{mute} \tag{3.1}$$

Example: Suppose that $d_{active} = 5$ m, $s_{conv} = 1$ m/sec, $T_{query} = 0.3$ sec, $T_{read} = 0.6$ sec, and $T_{mute} = 0.3$ sec then T_{active} $(= 5$ sec$) \geq T_{query} +$ T_{mute} $(= 0.6$ sec$)$, and the inequality holds true.

- **An Upper Bound on K.** Each cycle contains T_{query} , followed by T_{mute} , and at least $K-1$ read-mute intervals, where K is the maximum number of slots per cycle. Such a cycle allows the reading of K tags (since a tag with zero slot counter can be read immediately after the query command). Indeed, setting the number of possible reading operations in T_{cycle} to a number smaller than K implies that some tags will never be read. This follows since the slot counter in each tag is reloaded by the stored slot number upon receiving a query message. On the other hand, setting the number of possible reading operations to a number larger than K implies that each cycle has an idle period where no tag will generate a reply. Such idle time is wasteful in the design (unless energy consumption is of concern). Since $T_{query} \approx T_{read}$, we approximate T_{cycle} to

$$T_{cycle} = K \,.\, (\, T_{read} + T_{mute} \,) \tag{3.2}$$

Denote by $n_{active,slots}$ the maximum number of read-mute (including query-mute) intervals a tag experiences while traversing d_{active}. That is, $n_{active,slots}$ is the maximum read-mute intervals in time T_{active}. Thus,

$$n_{active,slots} = \lfloor \frac{T_{active}}{T_{read} + T_{mute}} \rfloor \tag{3.3}$$

The parameter K should not exceed $n_{active,slots}$, else, tags with slot number greater than $n_{active,slots} - 1$ will never be read. Thus,

$$K \leq n_{active,slots} \tag{3.4}$$

Note that the above inequality implies that $T_{cycle} \leq T_{active}$. However, setting $K = n_{active,slots}$ offers tags with any given slot number $k \in [0, K-1]$

only one read mute interval to be read and muted. Therefore, it may be advantageous to set K to a smaller value.

Example: Suppose that $d_{active} = 5$ m, $s_{conv} = 1$ m/sec, $T_{query} = 0.3$ sec, $T_{read} + T_{mute} = 0.9$ sec then $n_{active,slots} = \lfloor \frac{5}{0.9} \rfloor = \lfloor 5.55 \rfloor = 5$. Thus, $K \leq 5$.

- **A Lower Bound on K.** There are many scenarios that lead to a collision events between tags with the same slot number. Of the possible scenarios, we focus on one class of scenarios that arises when conditions 1 and 2 below hold simultaneously. In the explanation below, the belt is assumed to be moving from left to right. So, the rightmost tag of a colliding pair of tags is the tag that enters d_{active} first,

  1. The two tags involved in a collision hear the same query message while traversing d_{active}. Here, we assume that such a query is the first one heard by both tags (so, the rightmost tag of the collided pair has not previously heard the query), and
  2. The time between hearing the query and the time the rightmost tag leaves d_{active} allows the tag to reply to the query.

  Evidently, if the two conditions hold then a collision occurs. As mentioned above, however, collisions may occur in other scenarios. For example, the rightmost tag may have heard the query, replied, and collided prior to the entrance of the second tag. Thus, conditions 1 and 2 are sufficient but not necessary for the occurrence of collisions.

We now derive relations that aim at avoiding the above conjunction of events for the following cases.

Case of Constant inter-tag distances. For a constant d_{inter} , condition 1 holds between a pair of tags having the same slot number k if T_{cycle} is long enough to allow at least $K + 1$ tags whose slot numbers form the sequence $(k, k + 1, \ldots, 0, 1, \ldots, k)$ to traverse d_{active}. Such a sequence of $K + 1$ tags is separated by K inter-tag distances, each of the constant length d_{inter}. Using

the notations $t_{inter} = \frac{d_{inter}}{s_{conv}}$, condition 1 then holds if $K \,.\, t_{inter} \leq K \,.\, (T_{read} + T_{mute})$. To avoid such situations, we desire to have

$$t_{inter} > T_{read} + T_{mute} \tag{3.5}$$

Or, equivalently, $d_{inter} > (d_{read} + d_{mute})$. If this desired condition does not hold, then we aim at obstructing condition 2, as discussed next.

To analyze condition 2 when condition 1 occurs, consider a sequence of $K + 1$ tags having the slot counter value sequence $(k, k + 1, \ldots, 0, 1, \ldots, k)$. That is, the rightmost tag in the sequence has slot number k. Conditions 1 and 2 are then satisfied if T_{active} is long enough to allow such a sequence of tags to enter d_{active} while leaving enough time for the rightmost tag in the sequence to reply, i.e. $K \,.\, t_{inter} + k \,.\, (T_{read} + T_{mute}) < T_{active}$. To avoid such situations, we desire to have

$$K > \frac{T_{active} - k \,.\, (T_{read} + T_{mute})}{t_{inter}} \tag{3.6}$$

Or, equivalently, $K > \frac{d_{active} - k \,.\, (d_{read} + d_{mute})}{d_{inter}}$

If we want to choose K to avoid such collisions then we want to ensure either (3.5) holds and/or (3.6) holds for all $k \in [0, K - 1]$. We remark that,

- The lower bound on K obtained by relation (3.6) decreases monotonically as k increases; the highest bound is for $k = 0$.
- There may be cases where the obtained upper bound $K \leq n_{active,slots}$ is in conflict with some lower bounds obtained by relation (3.6) for slot numbers $k = 0, 1, 2, \ldots$. In such cases, we aim at setting K so as to satisfy the upper bound while satisfying as many lower bound inequalities as possible.

Example. Consider our running example where $d_{active} = 5$ m, $s_{conv} = 1$ m/sec, $T_{query} = 0.3$ sec, $T_{read} + T_{mute} = 0.9$ sec (so, $d_{read} + d_{mute} = 0.9$ m), we remark that

- For all cases where $d_{inter} > d_{read} + d_{mute}$ (0.9 m) we can set K to the value of the upper bound given be relation (3.4); that is, $K = 5$.

- When $d_{inter} < d_{read} + d_{mute}$, say $d_{inter} = 0.6$ m, relation (3.6) gives the following lower bounds: $K > 8.3$ (for $k = 0$), $K > 6.8$ (for $k = 1$), $K > 5.3$ (for $k = 2$), $K > 3.8$ (for $k = 3$), $K > 2.3$ (for $k = 4$). Thus setting $K = 5$ violates the 3 desired relations for $k = 0, 1$, and 2.

Case of Random inter-tag distances. Here we assume that inter-tag distances d_{inter} are independent and identically distributed (i.i.d) random variables drawn from some known distribution. As in the case of constant inter-tag distances, we analyze condition 1 by considering two tags having the same slot number k. We assume that such two tags have entered d_{active} as part of a sequence $K+1$ tags whose slot numbers form a sequence $(k, k+1, \ldots, 0, 1, \ldots, k)$.

We number the K inter-tag distances in the sequence sequentially as $d_{inter,i}$, $i = 1, 2, \ldots, K$. We use the notation $t_{inter,i} = \frac{d_{inter,i}}{s_{conv}}$. Condition 1 holds if T_{cycle} is long enough to permit such events to occur. That is, $\sum_{i=1}^{K} t_{inter,i} \leq K \,.\, (T_{read} + T_{mute})$. To avoid such events with a given threshold probability, denoted p_{thr} , we desire to have

$$Prob \left[\sum_{i=1}^{K} t_{inter,i} > K \,.\, (T_{read} + T_{mute}) \right] \geq p_{thr} \tag{3.7}$$

Equivalently, we can replace time periods by their equivalent distances in relation (3.7). If relation (3.7) does not hold when K is set to the upper bound given by relation 3.4, then we aim at obstructing condition 2, as discussed next.

To analyze condition 2 when condition 1 occurs, we consider a sequence of $K + 1$ tags having the slot counter value sequence $(k, k + 1, \ldots, 0, 1, \ldots, k)$. Conditions 1 and 2 are then satisfied if T_{active} is long enough to allow such sequence of tags to enter d_{active} while leaving enough time for the rightmost tag in the sequence to reply, i.e. $\sum_{i=1}^{K} t_{inter} + k \,.\, (T_{read} + T_{mute}) < T_{active}$. To

avoid such events with a given threshold probability p_{thr} , we desire to have

$$Prob\left[\sum_{i=1}^{K} t_{inter,i} \; > T_{active} \; - k \,.\, (T_{read} \; + T_{mute} \;)\right] \geq p_{thr} \tag{3.8}$$

We aim to find the smallest $K, K \leq n_{active,slots}$ (relation 3.4), that satisfies relation (3.8) for as many slot counter values $k \in [0, K-1]$ as possible. To this end, we search the interval $[1, \lfloor n_{active,slots} \rfloor]$ for the desired K value. For any such search value k_{search} , a difficulty arises in evaluating relations (3.7) and (3.8) since they include a sum of i.i.d. random variables in the term $\sum_{i=1}^{k_{search}} t_{inter,i}$.

The literature , however, provides many important results to cope with this difficulty. Of the available results we recall the following points. Below we write $X \sim DIST$ for a random variable X that has distribution DIST.

Case of $d_{inter} \sim U(a,b)$.

Here a and b denote the minimum and maximum inter-tag distances, respectively. For such cases, a formula of the Cumulative Distribution Function (CDF) of the sum of n *i.i.d.* variables is known (below, n plays the role of variable k_{search} above). In particular, let $U_{n\_sum} = U_1 + U_2 + \ldots + U_n$ be the sum of n *i.i.d.* random variables, where each $U_i \sim U(0,1)$ (Note: n in the latter formula stands for k_{search} above.) It is known that U_{n_sum} has the Irwin-Hall distribution (see, e.g. [Reyni 1970], page 197) whose CDF is given by

$$F_{U_{n\_sum}}(u) = \begin{cases} \frac{1}{n!} \sum_{k=0}^{\lfloor x \rfloor} (-1)^k \binom{n}{k} (x-k)^{n-1} & 0 \leq u \leq n \\ \\ 0 & otherwise \end{cases} \tag{3.9}$$

We are interested in the sum

$d_{inter\_n\_sum} = d_{inter,1} + d_{inter,2} + \cdots + d_{inter,n}$

where each $d_{inter,i} \sim U(a,b)$. Therefore, $d_{inter\_n\_sum} \in [na, nb]$. Rescaling, the variables $d_{inter\_n\_sum}$ gives for any positive value d,

$$Prob[d_{inter\_n\_sum} \; \leq d] = Prob\left[U_{n\_sum} \; \leq \frac{(d-na)}{(b-a)}\right] \tag{3.10}$$

Numerical values in the following examples are obtained by programming a function called *irwin_hall (n,a,b,d)*. Here, n is the number of i.i.d. variables in the sum, a and b are the two parameters in the distribution $U(a, b)$, and d is any positive value. The function computes $Prob[d_{inter\_n\_sum} \leq d]$.

Example. In our running example, $d_{active} = 5$ m, $s_{conv} = 1$ m/sec, $T_{read} + T_{mute} = 0.9$ sec (so, $d_{read} + d_{mute} = 0.9$ m). Assume that d_{inter} ~$U(a = 0.6$ m, $b = 3.0$ m). We wish to find possible values of $K \leq 5$ that satisfy relation (3.8) for the slot number $k = 0$. Using relations (3.9) and (3.10), we get the following results

- For $k_{search} = 2$ (i.e. n = 2), and $k = 0$:
  $Prob[d_{inter\_2\_sum} > 5] = Prob[U_{2\_sum} > 1.58333] = 0.0868056$
- For $k_{search} = 3$ (i.e. n = 3), and $k = 0$:
  $Prob[d_{inter\_3\_sum} > 5] = Prob[U_{3\_sum} > 1.3333] = 0.623457$
- For $k_{search} = 4$ (i.e. n = 4), and $k = 0$:
  $Prob[d_{inter\_4\_sum} > 5] = Prob[U_{4\_sum} > 1.08333] = 0.942618$

We conclude that if we set $p_{thr} = 0.75$ then $k_{search} = 4$ is the minimum acceptable value.

Case of d_{inter} has a shifted exponential distribution.

Here we assume that $d_{inter} = a + X$, where $a \geq 0$ is a constant specifying the minimum separation distance between any two consecutive tags, and X is exponentially distributed with parameter λ (so, its average is $1/\lambda$ meters). Such a distribution allows d_{inter} to occasionally assume large values while maintaining a minimum separation distance of a meters. It is known that if X_{n_sum} is the sum of n, $n \geq 1$, i.i.d. exponential distributions, each with parameter λ then X_{n_sum} has the Erlang (n, λ)

distribution (e.g. [32]):

$$F_{X_{n\_sum}}(x) = \begin{cases} 1 - \left[exp(-\frac{x}{b})\right] \left(\sum_{i=0}^{c-1} \frac{(x/b)^i}{i!}\right) & x \geq 0 \\ 0 & otherwise \end{cases} \tag{3.11}$$

We are interested in the sum $d_{inter\_n\_sum} = d_{inter,1} + d_{inter,2} + \ldots + d_{inter,n}$. Thus, for any positive value d

$$Prob[d_{inter\_n\_sum} > d] = Prob[X_{n\_sum} > d - na] \tag{3.12}$$

Numerical values in the following examples are obtained by programming a function called *shifted_erlang (n,a,λ,d)*. Here, n is the number of *i.i.d.* variables in the sum, a is a constant shift, λ is the parameter of the Exponential distribution, and d is any positive value. The function computes $Prob[d_{inter\_n\_sum} > d]$.

Example . In our running example, d_{active} = 5 m, s_{conv} = 1 m/sec, $T_{read} + T_{mute}$ = 0.9 sec (so, $d_{read} + d_{mute}$ = 0.9 m). Assume that $d_{inter} \sim$ 0.6 m + $Exp(\lambda = 0.5)$. We wish to find possible values of $K \leq 5$ that satisfy relation (3.8) for the slot number $k = 0$. Using relations (3.11) and (3.12), we get the following results

- For $k_{search} = 2$ (i.e. n = 2), and $k = 0$:
  $Prob[d_{inter\_2\_sum} > 5] = Prob[X_{2\_sum} > 3.8] = 0.433749$
- For $k_{search} = 3$ (i.e. n = 3), and $k = 0$:
  $Prob[d_{inter\_3\_sum} > 5] = Prob[X_{3\_sum} > 3.2] = 0.783358$

We conclude that if we set $p_{thr} = 0.75$ then $k_{search} = 3$ is the minimum acceptable value.

3. In other situations, where the *i.i.d.* $W_n = X_1 + \ldots + X_n$ is difficult to assess, one can resort to a Gaussian approximation (obtained using a Central Limit Theorem) [10]. The Central Limit Theorem [10] asserts that if $E[X]$ and $Var[X]$ are the mean and variance of each random variable X_i then for large n, W_n can be approximated by the Gaussian (i.e. Normal)

distribution Normal $(nE[X], nVar[X])$. For some distributions of X_i, convergence to a Gaussian distribution is fast (e.g. $n = 4$) whereas convergence can be slow for other distributions.

3.4 Useful Relations for the Dual Reader Scheme

In this section, we derive relations for the D-READER scheme. We assume that the Q-unit transmits query commands at a rate that ensures that each tag hears at least one query command. We henceforth focus on the interrogation zone of the R-unit of length d_{active}. We use K to denote the largest used slot counter value. We assume that tags enter the interrogation zone of the R-unit with consecutive slot numbers (1,2, ..., K, 1, 2, ...).

- **Feasibility.** The speed of the conveyor belt should allow at least one tag to be read and muted. Thus,

$$T_{active} \geq T_{read} + T_{mute} \tag{3.13}$$

- **An Upper Bound on K.** As in the previous section, denote by $n_{active,slots}$ the number of slots (i.e., read-mute intervals) experienced by a tag while traversing d_{active}. Thus, we need to set K such that $n_{active,slots} \geq K$. Else (if $n_{active,slots} < K$) then some tags will never be read. On the other hand, if $n_{active,slots} > K + 1$ then there will be some idle slots. If $n_{active,slots} = K + 1$ then the last slot can be used to read a tag with slot number $k = K$ that enters the interrogation zone after a read command is issued. Thus, we want to set K such that

$$K \leq n_{active,slots} - 1 \tag{3.14}$$

  We note that, in this scheme, every tag with any slot number $k \in [1, K]$ has only one read opportunity in any reading station.

- **A Lower Bound on K.** We consider collisions that occur between two tags with slot numbers denoted k_1 and k_2, where $k_1 \leq k_2$. The

tag with slot number k_2 enters the interrogation zone first, and after hearing $k_{diff} = k_2 - k_1$ read commands (i.e., when k_2 is decremented to $k_2 - k_{diff} = k_1$) the second tag enters the zone. So, there is some time interval where the two tags have the same slot counter value of k_1. If K is the largest slot counter value then $k_{diff} \in [0, K-1]$. We distinguish the following cases.

Case $k_{diff} = 0$. Here, the conveyor belt speed and the inter-tag distances allow two tags with the same slot number $k \in [1, K]$ to enter the interrogation zone without hearing any read command. Roughly, this happens if such tags enter in time $T_{read} + T_{mute}$ (i.e., within one slot). We then have the following observations.

Case of constant d_{inter}. Since there are K inter-tag distances between the two collided tags, the situation arises when $K \,.\, t_{inter} \leq T_{read} + T_{mute}$ (where $t_{inter} = \frac{d_{inter}}{s_{conv}}$). To avoid such situation, we need

$$K > \frac{T_{read} + T_{mute}}{t_{inter}} \tag{3.15}$$

Case of random d_{inter}. As in the previous section, we use $d_{inter,i}$, $i = 1, 2, \ldots, K$, to denote K *i.i.d.* random inter-tag distances, and let $t_{inter\ i} = \frac{d_{inter\ i}}{s_{conv}}$ for any i. Here, the situation arises when $\sum_{i=1}^{K} t_{inter\ i} < T_{read} + T_{mute}$. To avoid such situation with a given threshold probability p_{thr} , we need

$$Prob\left[\sum_{i=1}^{K} t_{inter\ i} > T_{read} + T_{mute}\right] \geq p_{thr} \tag{3.16}$$

Case $k_{diff} \geq 1$. In such cases, the two tags are separated by $K - k_{diff}$ inter-tag distances. For example, if $K = 5$, and the following sequence enters the zone: 4 (= k_2), 5, 1, 2 (= k_1), where k_2 enters first, then the first and last tags are separated by $K - k_{diff} = 5 - (4 - 2) = 3$ inter-tag distances.

For the above collision scenario to arise, the total separation distance

should be traversed in time that does not differ than $k_{diff} \cdot (T_{read} + T_{mute})$ by more than $T_{read} + T_{mute}$; this latter condition is required to ensure that some time interval exists where the two tags are in the interrogation zone and have identical slot counter value.
The tight timing constraint in the above relation makes such type of collisions unlikely to occur, and hence, they can be ignored.

3.5 Simulation Results

In this section, we present simulation results to investigate the performance of the devised schemes. The results are obtained by implementing a discrete-event simulator (DES) in Java. The simulator handles a number of scenarios. In each scenario, the following parameters are input as constants: s_{conv}, d_{active}, T_{query}, T_{read}, T_{mute}, and the maximum number of slots per cycle. The inter-tag distance d_{inter} can either be constant, or distributed according to a uniform distribution $U(a, b)$. The implemented configurations are

1. One station using the S-READER scheme
2. Two stations, each using the S-READER scheme
3. One station using the D-READER scheme

Configuration 2 is implemented to investigate the improvement gained over configuration 1. In addition, since the number of readers in configuration 2 is the same as configuration 3, we investigate their relative performance. Table 3.1 shows input values that are common to all experiments.

The simulator produces the following output.

- The *missed tag* ratio (as a percentage): a missed tag exits the interrogation zone before it can be read. The ratio is the fraction of the number of missed tags to the number of all tags in a simulation run.
- The *collided tag* ratio (as a percentage): the fraction of collided tags to all tags in one run.

Parameter	Value
s_{conv}	1 m/sec
T_{query}	300 ms
T_{read}	600 ms
T_{mute}	300 ms
t_{idle}	0 ms
S-READER K	[0,3]
D-READER K	[1,4]
d_{active}	5 m

Table 3.1: Parameter Values

- The *successful tag* ratio (as a percentage): the fraction of successfully read tags to all tags in one run.

Each trial processes a sequence of 1000 tags, and each point on a curve is an average of 10 trials (for the constant d_{inter} case) and 100 trials (for the random d_{inter} case). The experiments analyze the following aspects.

- Effect of varying the parameter K on the success rate
- Comparing the performance of the three configurations mentioned above
- Effect of inserting an idle period in each cycle (e.g. $t_{idle} = 1000$ ms)

3.5.1 Effect of Varying the Parameter K

Figure 3.3 presents the obtained success rate of the S-READER scheme with one station. Tags are equally spaced apart, and we vary d_{inter} (the x-axis) in the range [1.1 m, 1.9 m]. Curves in the figure correspond to setting K in the range [2,5].

For the used problem parameters, we note that:

- $n_{active,slots} = \lfloor \frac{T_{active}}{T_{read}+T_{mute}} \rfloor = \lfloor \frac{5}{0.9} \rfloor = \lfloor 5.55 \rfloor = 5$. Thus, relation (3.5) is satisfied for $K \leq 5$. Accordingly, in Figure 3.3, we experiment with K in the range [2,5].
- Relation (3.6) $t_{inter} > T_{read} + T_{mute}$ $(= 0.9)$ is satisfied for all values of $t_{inter} \in$ [1.1 m, 1.9 m]. Thus, collisions of the type analyzed in

Section 3.3 do not occur for any K. Accordingly, failures occur because of missed tags, or the occurrence of other types of collisions. Setting $K = 5$ is expected to suffer from relatively high missed tags.

The results show that $K = 3$ and 4 give the highest success rate. The findings are in-line with the above analysis.

Figure 3.4 shows the obtained success rate of the D-READER scheme with one station. Tags are equally spaced apart and we vary d_{inter} (the x-axis) in the range [1.1 m, 1.9 m].

For the used parameters, we note that:

- $n_{active,slots} = \lfloor \frac{T_{active}}{T_{read} + T_{mute}} \rfloor = \lfloor \frac{5}{0.9} \rfloor = \lfloor 5.55 \rfloor = 5$. Thus, relation (3.14) is satisfied for $K \leq 4$. In Figure 3.4, we experiment with the K in the range [2,4].
- For the smallest t_{inter} value in the range [1.1 m, 1.9 m], the ratio $\frac{T_{read} + T_{mute}}{t_{inter}} = \frac{0.9}{1.1} = 0.818$. Thus, relation (3.15) $K > \frac{T_{read} + T_{mute}}{t_{inter}}$ is satisfied for all $K \in [2, 4]$. Thus, no collision for the case of $k_{diff} = 0$ of Section 3.4 arises.
- The scheme does not result in missed tags (due to early exit of a tag from the interrogation zone before sending its ID).

The results in Figure 3.4 show that the resulting success rate is perfect when $K = 4$, which is in-line with the above analysis. For other K values, failures occur because of the case $k_{diff} \geq 1$.

3.5.2 Performance of the S-READER scheme with two stations

The use of a cascade of two S-READER stations is expected to improve over the use of a single S-READER station. Figure 3.5 when compared with Figure 3.3, confirms the above intuition when tags are equally spaced and d_{inter} varies in the range [1.1 m, 1.9 m].

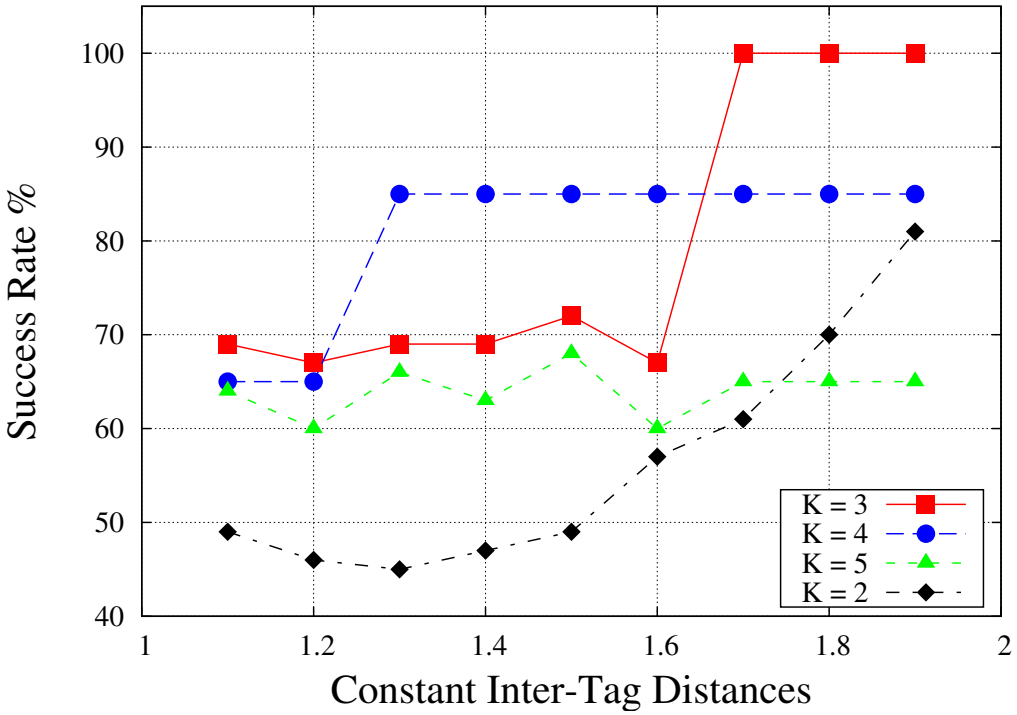


Figure 3.3: Effect of varying $K \in [2, 5]$ on the S-READER scheme (equal d_{inter})

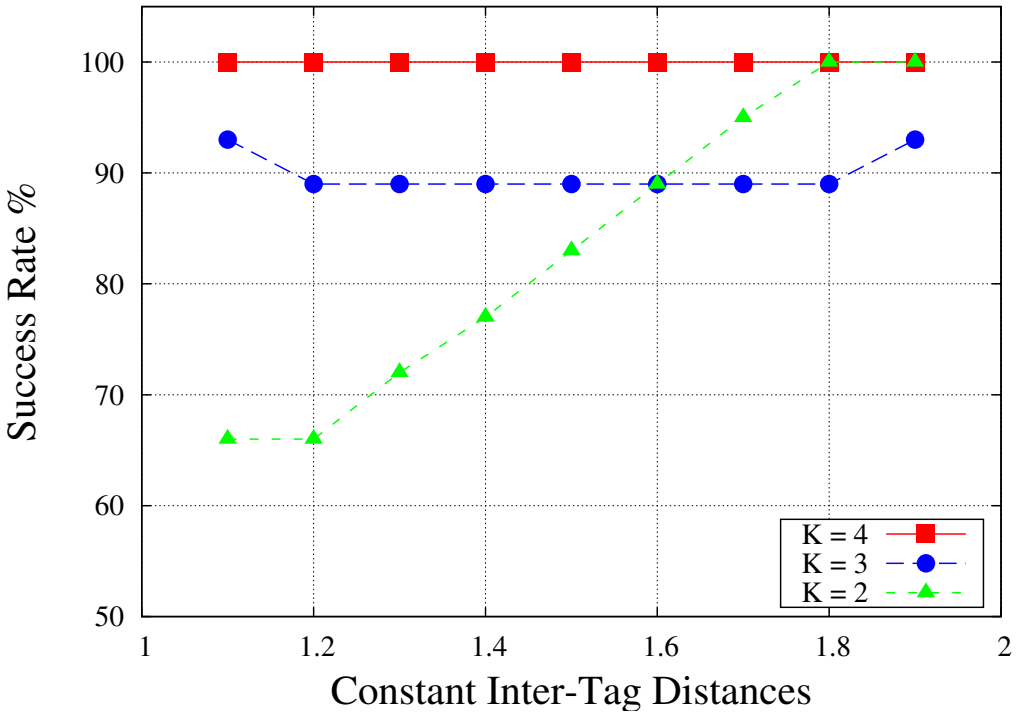


Figure 3.4: Effect of varying $K \in [2, 4]$ on the D-READER scheme (equal d_{inter})

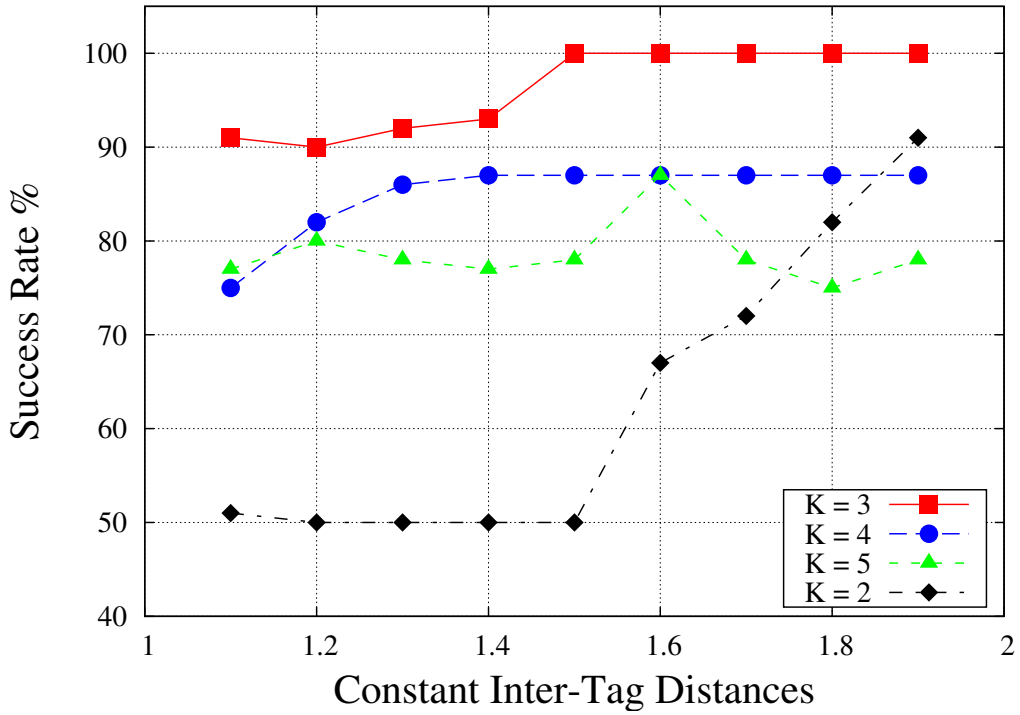


Figure 3.5: Effect of varying $K \in [2, 5]$ on the S-READER scheme with two stations (equal d_{inter})

Random d_{inter}	Avg.%	Min.%	Max.%	Std. Dev%	Con. Int%
S-READER - 1 Station	77	43	90	10.9	$\pm$ 2.2
S-READER - 2 Stations	79	46	90	10.1	$\pm$ 2.0
D-READER	90.85	43.1	100	15.4	$\pm$ 3.02

Table 3.2: Comparison of success rates of different configurations ($K = 4$,d_{inter} ~$U(0.6m, 3.0m)$)

Figure 3.6 compares the two configurations when $K = 4$. Figure 3.7a and Figure 3.7b, respectively, presents the minimum and maximum, respectively, achieved success rate by the S-READER scheme with one and two stations when $K = 4$.

We have also conducted experiments to compare the three configurations (i.e. the S-READER scheme with one station and two stations, and the D-READER scheme) when d_{inter} ~U(0.6 m, 3.0 m). The results are presented in Table 3.2.

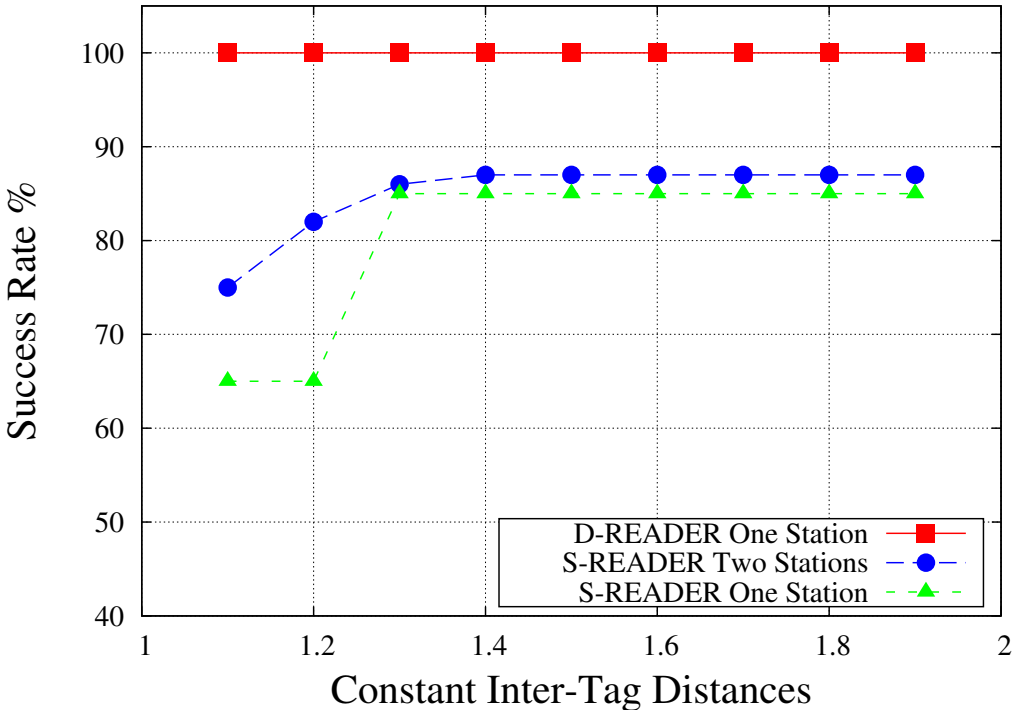
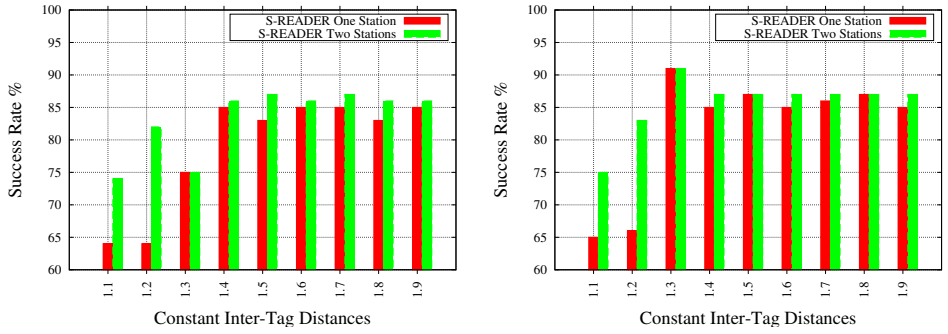


Figure 3.6: Comparing S-READER scheme with one and two stations, and the D-READER scheme ($K = 4$, equal d_{inter})



(a) Minimum Success Rate (b) Maximum Success Rate

Figure 3.7: Histograms of 3.7a and 3.7b maximum success rates of the S-READER scheme with one and two stations ($K = 4$, equal d_{inter})

3.5.3 Effect of inserting idle periods

RFID readers may be required to operate with less than 100% duty cycle. For example, to allow other co-located readers to operate, or to conserve energy. In the S-READER scheme, such experiments can be achieved by inserting an idle period t_{idle} between consecutive cycles. For example, setting $T_{query} = 0.3$ sec, $T_{read} + T_{mute} = 0.9$ sec, $t_{idle} = 1$ sec, and $K = 4$, results in $T_{cycle} = 0.3 + (3 \times 0.9) + 0.3 = 3.3$s, and thus a duty cycle of $(\frac{T_{cycle}}{T_{cycle} + t_{idle}} \times 100) = \frac{3.3}{4.3} \times 100 = 76.7\%$.

In the D-READER scheme, the requirement can be achieved by inserting an idle period t_{idle} in every sequence of K read commands. Thus, setting $T_{read} + T_{mute} = 0.9$ sec, $t_{idle} = 1$ sec, and $K = 4$ results in a duty cycle of the R-unit to $\frac{K \times (T_{read} + T_{mute})}{K \times (T_{read} + T_{mute}) + t_{idle}} \times 100 = \frac{3.6}{4.6} \times 100 = 78.2\%$.

Figure 3.8 compares the three configurations when $K = 4$, and constant $d_{inter} \in [1.1$ m, 1.9 m]. Figure 3.9 compares the three configurations when $K = 4$, and $d_{inter} \sim U(0.6$ m, 3.0 m).

In general, performance without an idle period is better than that with idle periods. For the S-READER scheme, however, an idle period can reduce collisions in the following scenario. Suppose that tags A and B have the same slot counter value. Tag A enters the interrogation zone first, followed by tag B. Both tags hear a query command, and their responses collide. Tag A exits d_{active} during the idle period. Tag B succeeds in replying in the next cycle. If the idle period does not exist, then the tags may collide again in the next cycle.

3.6 Concluding Remarks

In this chapter, we have devised two framed slotted Aloha based anti-collision schemes for identifying a sequence of tags placed on a moving conveyor belt. The S-READER scheme uses one reader per identification station. The D-

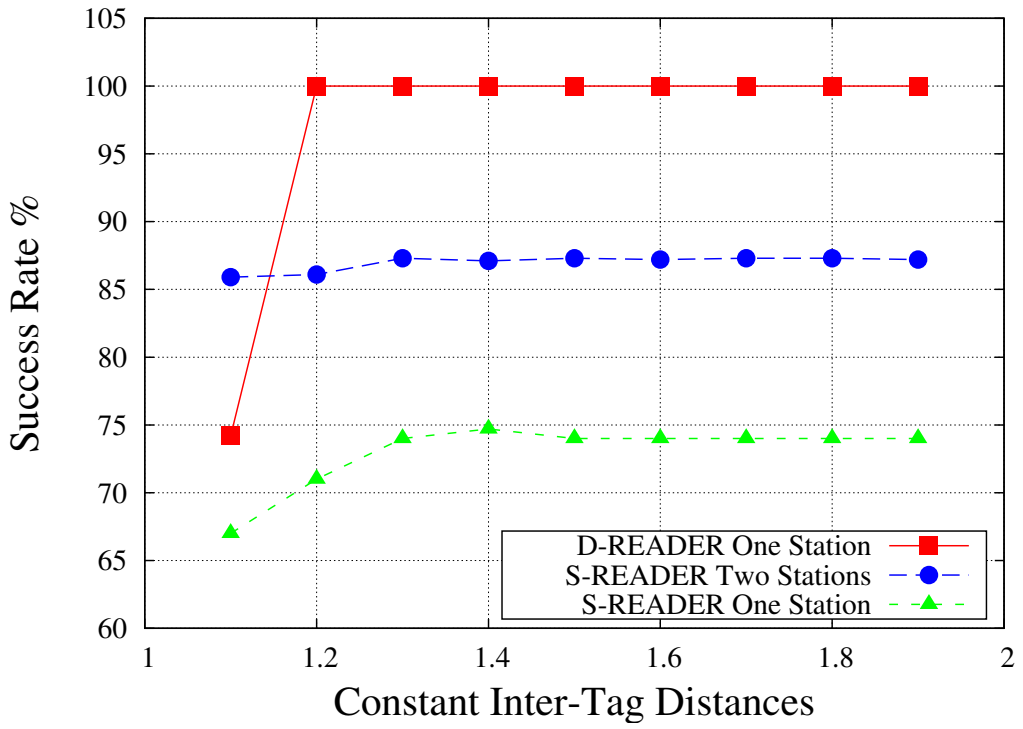


Figure 3.8: Comparison of success rates of different configurations with t_{idle} = 1 sec. ($K = 4$, equal d_{inter})

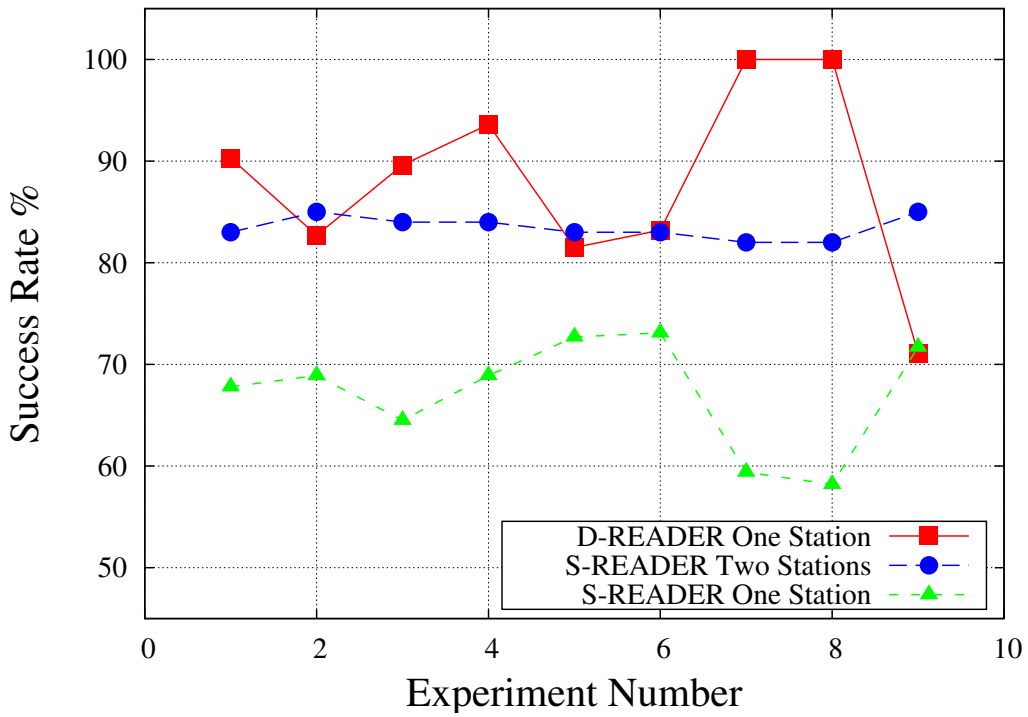


Figure 3.9: Comparison of success rates of different configurations with t_{idle} = 1 sec. ($K = 4$,d_{inter} ~U(0.6 m, 3.0 m))

READER scheme uses two readers per identification station. The first reader in the D-READER scheme, however, is not a full reader device since its only function is to emit query commands. We have derived conditions on the maximum slot count, denoted K, that aim at helping a designer to tune each scheme for improved performance. The D-READER scheme is more expensive, but has shown better performance over using a cascade of two readers running the S-READER scheme each.

Chapter 4

Mobility in UWSNs

In this chapter, we give an overview of UWSNs with emphasis on modelling uncontrolled node mobility due to water currents. The overview gives pointers to work done on fundamental networking problems such as designing medium access control (MAC) protocols, routing algorithms, connectivity and coverage algorithms, and localization methods. We then discuss some existing work on modelling node mobility in such networks.

4.1 Introduction

The area of UWSNs has attracted attention of many researchers in recent years (see, e.g., [2, 23, 35, 39, 51]). A typical UWSN contains a collection of sensor nodes that performs different tasks including sensing, communication, and data processing. Applications of such networks include gathering of oceanographic geological and biological information, offshore exploration of natural resources, and measurement of water characteristics (e.g., temperature, and oxygen levels), and pollutant content.

The traditional method of serving some of the above objectives is to record data by a device deployed in a water area under exploration during a monitoring mission. At the end of the mission, the device is retrieved, and the data is collected. Compared to the above traditional way, UWSNs are designed to allow inter-node communication as well as communication with onshore control systems. Thus, they allow better real time monitoring and transmission of the

collected data. In addition, they allow for better handling of device failure and misconfigurations.

Networking research on UWSNs span many directions, such as design of energy efficient Medium Access Control (MAC) protocols, routing protocols, localization methods, and connectivity and coverage algorithms. Below, we highlight some of these aspects.

The underwater communication channel. The design of UWSNs is challenged by the difficulty in providing high bandwidth (see, e.g., [2, 51]). In particular, it is noted that the majority of radio frequencies suffer strong attenuation in salt water. Light is also strongly scattered and absorbed underwater. Sound also suffers from various factors of attenuation, spreading, and noise. Nevertheless, acoustic communication is currently perceived as the most practical communication method.

Medium Access Control (MAC) protocols. MAC protocols for UWSNs are surveyed in [20, 52, 67, 69]. Examples of existing work on MAC protocols include the work of

- [58, 65] on nodes that have sleep periods subject to a scheduling discipline,
- [54, 38] on distributed CDMA and OFDMA based protocols, respectively, and
- [28] on the performance of the IEEE 802.11 MAC in underwater wireless channels.

Routing. Work on UWSNs routing is surveyed in [6, 8, 34]. Delay tolerant routing is also surveyed in [22, 53]. Examples of existing work on routing protocols include the work of

- [36, 46] on delay tolerant routing,
- [37] on prediction assisted routing,

- [40] on machine learning assisted routing,
- [45] on pressure routing,
- [66] on depth based routing, and
- [71] on link-state routing.

Connectivity and coverage. Work in this areas is surveyed in [33, 74]. For example, work on 2-dimensional coverage includes [7], whereas the work of [3] concerns 3-dimensional coverage.

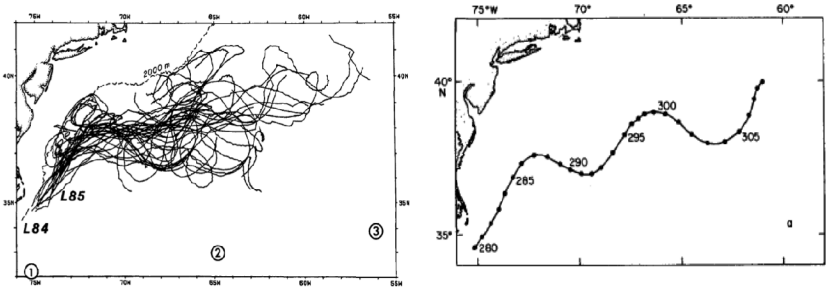
Localization. Work on UWSNs localization is surveyed in [18, 27, 62]. For example, the work of [72, 73] consider the use of mobility prediction in performing node localization.

4.2 Deployment Schemes and Mobility Models

In this section, we discuss UWSNs deployment schemes with emphasis on node mobility modelling. Deployment schemes of UWSNs are classified in [39] as being either *static*, *mobile*, or *semi-mobile*. In static deployments, nodes are attached to underwater ground, or anchored surfaces (e.g., buoys). In contrast, in mobile schemes, nodes can have either uncontrollable mobility such as nodes moving freely with water currents, or self-controlled mobility such as Autonomous Underwater Vehicles (AUVs). A useful form of controllable mobility is vertical movements induced by mechanical devices inside nodes (see, e.g., [26]). Semi-mobile deployment schemes have nodes that can either be static or mobile.

A kinematic mobility model. Work on using free mobile nodes (also called RAFOS drifters) to collect oceanographic data appears, for example, in the well-cited work of [12]. The work of [12] reports on observations collected by deploying 37 RAFOS drifters deployed off Cape Hatteras. Data are collected over a period of about 45 days. During the mission, the geographic location

of each drifter is observed by equipping each unit with a GPS, and a satellite transmitter. Figure 4.1a sketches the recorded trajectories of the 37 drifters. Figure 4.1b sketches the trajectory of drifter number 22.



(a) Trajectories of 37 drifters (b) Trajectory of drifter 22

Figure 4.1: Drifter trajectories (adpated from [12])

Work on developing a mathematical model (a kinematic model) to capture the important mobility patterns observed in the above work appears in [13]. More recently, in [17], the authors adopted a similar kinematic model to investigate networking issues. In the kinematic model, node trajectory is obtained by solving differential equations, defined by Equations 4.1 and 4.2, below.

$$\varphi(x,y,t) = -tanh[\frac{y - B(t)sin(k(x-ct))}{\sqrt{1+k^2B^2(t)cos^2(k(x-ct))}}] + cy \tag{4.1}$$

where $B(t) = A + \epsilon cos(\omega t)$, the x and y velocities are given by

$$\dot{x} = -\frac{\partial\varphi}{\partial y}; \dot{y} = \frac{\partial\varphi}{\partial x} \tag{4.2}$$

The model assumes that a node is deployed at time $t = 0$, at the origin (X, Y)= (0,0). The parameter settings used in [17] are $A = 1.2, c = 0.12, k = \frac{2\pi}{7.5}, \omega =$ 0.4, and $\epsilon = 0.3$.

The kinematic model is deterministic. However, any small variation in the Cartesian coordinates of an initial node deployment point at time $t = 0$, causes a large variation in the obtained trajectory (e.g., the number of vortices the node encounters, and the time duration of each encounter). Thus, during an interval of time $[t_1, t_2]$ after a node deployment at time $t = 0$, one

can numerically solve the above differential equations to obtain one trajectory during this time interval. Moreover, by repeating the above step for many nodes with slightly different initial deployment locations, one obtains many possible trajectories. The probabilistic mobility model, reviewed next, uses such trajectories to obtain a discrete probability distribution of possible node locations.

A probabilistic node locality model. Our work in the next chapter adopts a probabilistic node locality mode used in [43]. The model considers a time interval $[t_1, t_2]$ after deployment of a node, say x, at time $t = 0$, and small perturbations of coordinates (0,0). The model relies on identifying a geographic area A where node x is likely to be inside during the specified time interval $[t_1, t_2]$. Area A is expected to be large if the time interval $[t_1, t_2]$ is large. The model discretizes area A by superimposing a hypothetical grid of rectangles. The model assumes that the interval $[t_1, t_2]$ is small enough so that the location of x is likely to be confined to one rectangle. The idea is to identify for such a node x, a locality set, denoted $Loc(x) = \{x[1], x[2], x[3], \dots\}$ where each $x[i]$ denotes a grid rectangle that may contain node x during the interval $[t_1, t_2]$.

For example, using the kinematic model above, one can generate many possible trajectories of node x, and then identify the set $Loc(x)$. In addition, one can calculate for each such a rectangle $x[i] \in Loc(x)$, the ratio of the number of trajectories having x in rectangle $x[i]$ to the total number of generated trajectories. If the number of generated trajectories is large, one can then take such ratio as the probability that node x is in location $x[i]$; we use $p_x[i]$ to denote such a probability.

Probabilistic graphs. The information obtained in the probabilistic node locality model is used to define a *probabilistic* graph (also called probabilistic network) $G = (V, E, Loc, p)$ where

- V is a set of nodes
- E is the links relationship. If node x when placed anywhere in rectangle $x[i]$ can reach node y when placed anywhere in rectangle $y[j]$ then we set

the link relationship $E(x[i], y[j]) = 1$. Note that the definition allows us to ignore connectivity between rectangles $x[i]$ and $y[j]$ if for any point inside rectangle $x[i]$ (or $y[j]$) node x cannot reach node y. The definition serves the purpose of enabling us to compute lower bounds on the desired solutions of the problems discussed in the next chapter.

- *Loc*: for each node $x \in V$, $Loc(x)$ is the set of grid rectangles where x can lie inside.

- p: for each node $x \in V$, $p_x[i]$ is the probability that node x lies in rectangle $x[i]$.

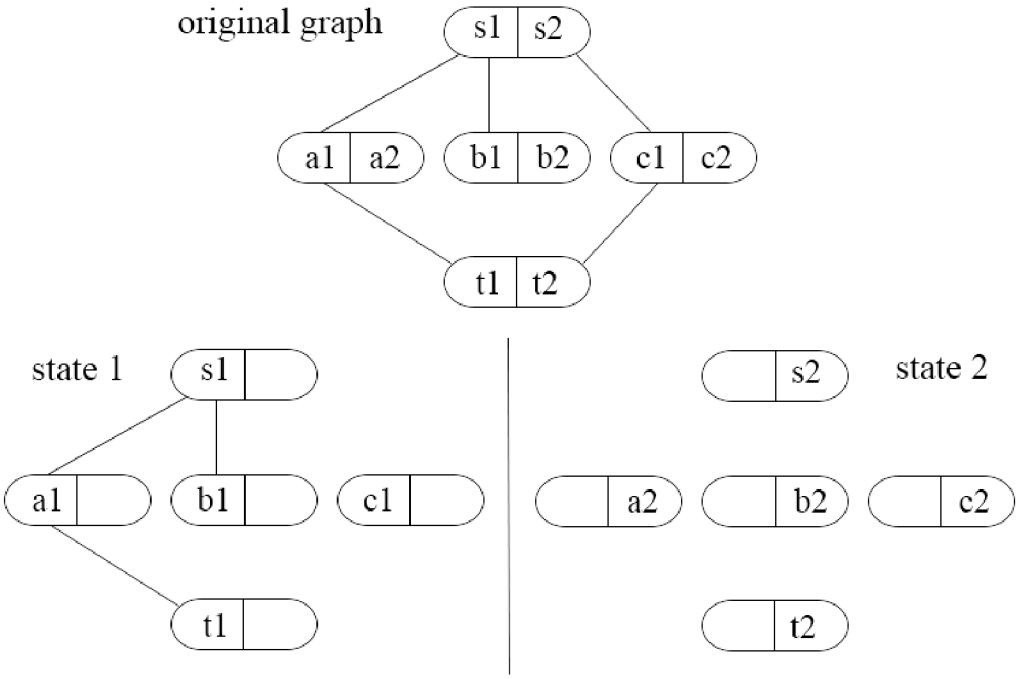


Figure 4.2: A probabilistic graph and two of its states

Example: Figure 4.2 illustrates a probabilistic graph on 4 nodes (s, a, b, c, t) where, for example, $Loc(s) = \{s1, s2\}$. We now mention the following remarks.

- First, when no confusion may arise, we sometimes use $x[i]$ to refer to
  - a possible location of node x,
  - node x itself (when in location $x[i]$), and

– the event that node x is at location $x[i]$.

- Second, when acoustic communication is used, the delay in one direction (say, $x[i]$ to $y[j]$) may be significantly different than the delay in the reverse direction (from $y[j]$ to $x[i]$). This follows since sound propagation speed depends on the direction of water currents.

- Third, a state S of a probabilistic graph $G = (V, E, Loc, p)$ is obtained by fixing a location $x[i]$ for each node $x \in V$.

Thus, the number of distinct states of G is given by the product $\prod_{x \in V} |Loc(x)|$. Assuming that nodes take different locations independently of each other, the probability that a given state $S = \{x_1[i_1], x_2[i_2], \ldots, x_n[i_n]\}$ occurs is $Pr(S) = \prod_{x[i] \in S} p_x[i]$. We use the above concepts in the next chapter.

Example. The probabilistic graph G in Figure 4.2 has 16 possible states.

The underlying graph of a probabilistic graph. To facilitate standard graph operations (e.g., drawing a graph, or computing a subgraph) on a given probabilistic graph G, we adopt the notion of the underlying graph of $G = (V, E, Loc, p)$. The underlying graph has the same set of nodes V as in G. In addition, it has an edge (x, y) if x placed in at least one location in $Loc(x)$ can reach node y when placed in at least one location in $Loc(y)$. Using the OR operation, one may write $E(x, y) = \bigvee_{x[i] \in Loc(x) \text{ and } y[j] \in Loc(y)} E(x[i], y[j])$.

Example. Figure 4.3 illustrates the underlying graph of the probabilistic graph of Figure 4.2.

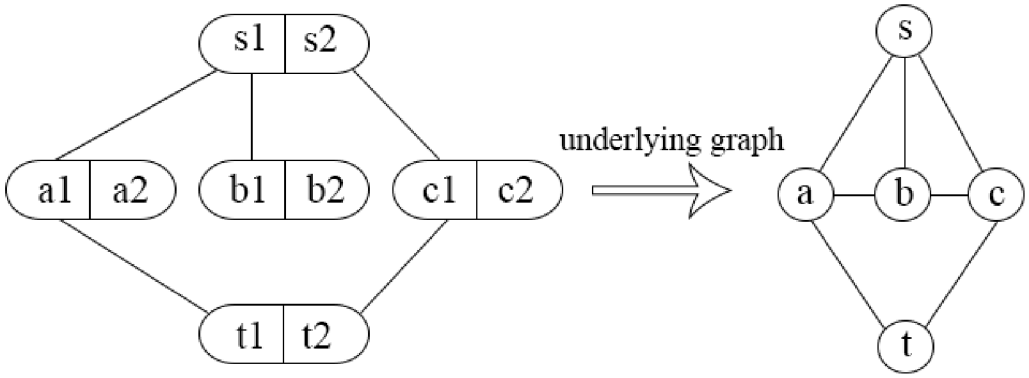


Figure 4.3: Example of the underlying graph of a probabilistic graph

4.3 Summary

In this chapter, we have outlined some research work done on UWSNs with a goal of defining the concepts of probabilistic node locality model, and probabilistic graphs. In the next chapter, we utilize these concepts to investigate two UWSN reachability problems.

Chapter 5

Two Terminal Delay Bounded Connectivity

Given two nodes in an UWSN where all nodes can move freely in water, we analyze the likelihood that one node can reach the other node by a path of bounded length or delay. Our main contribution in this chapter is on formalizing two problems that serve this purpose. The formulations adopt the probabilistic graph model discussed in the previous chapter. We devise an approach for obtaining lower bounds on the solution by approximating the structure of the given UWSN using a set of node disjoint paths. algorithms in the approach run in polynomial time of the input network model. We also conduct numerical evaluation experiments to investigate the quality of the obtained bounds.

5.1 Problem Formulation

We consider an UWSN modelled by a probabilistic graph $G = (V, E, Loc, p)$, introduced in the previous chapter. In this chapter, We deal with the following problems:

- the 2-terminal hop-bounded connectivity problem, denoted HB-$Conn_2$ problem, and
- the 2-terminal delay-bounded connectivity problem, denoted DB-$Conn_2$ problem.

The problems are defined as follows.

- **Definition (the HB-$Conn_2$ problem)**: Given a probabilistic UWSN graph $G(V, E, Loc, p)$, two distinguished nodes s and t, and a maximum number of links D, compute the probability, denoted HB- $Conn_2(G, s, t, D)$, that the network is in a state where s can reach t by a path of length $\leq D$.

- **Definition (the DB-$Conn_2$ problem)**: Given a probabilistic weighted graph $G = (V, E, Loc, p, d)$, two distinguished nodes s and t, and a maximum delay D, compute the probability, denoted DB-$Conn_2(G, s, t, D)$, that the network is in a state where s can reach t by a path of total delay $\leq D$.

We make the following remarks.

- Each of the above problems can be solved by an exact algorithm. The algorithm works exhaustively, as outlined below.

  - We generate all possible states of G. For each state S, classify the state as being either operating or failed.
  - A state S is operating if the shortest path between s and t satisfies the specified hop (or delay) constraint D.
  - Denote by $OP(G)$ the set of operating states. The desired solution is then given by the sum $\sum_{S \in OP(G)} Pr(S)$.

  The running time of the above exhaustive enumeration algorithm, however, grows exponentially with the size of the given problem instance (G, s, t, D). Thus, more efficient algorithms are needed.

- The above problems have similarity to the *2-terminal network reliability* problem with node failure, denoted $Rel_2(G)$, discussed in [24]. In particular, the Rel_2 problem is also defined with respect to two specified nodes, denoted s and t, using a type of probabilistic graphs where each node can either be operating or failed (as opposed to a mobile node being in one of several possible locations). In the Rel_2 problem, a state

S of a given probabilistic graph G is considered operating if there is an (s,t)-path of operating nodes in S. Else, the state S is failed. Rel_2 asks for computing the probability $\sum_{\text{S is operating}} Pr(S)$. Rel_2 is shown in [24] to be #P-hard (#P is a class of enumeration problems).

Although we have not shown the complexity of the HB-$Conn_2$ or the DB-$Conn_2$ problems, however, we suspect that these two reachability problems are also #P-hard.

Our main contribution in this chapter is to develop exact algorithms for the HB-$Conn_2$ and the DB-$Conn_2$ problems when the input probabilistic graph G is composed of a number of node disjoint (s,t)-paths. The algorithms run in polynomial time and can be applied to any subgraph H of the original input graph where H is composed of node disjoint (s,t)-paths. In such cases, the algorithms compute lower bounds on the desired solutions. We describe our devised algorithms in the rest of the chapter.

5.2 Key Data Structures and Functions for the HB-$Conn_2$ Algorithm

In this section, we present key data structures and operations used by our HB-$Conn_2$ algorithm.

- R_x: The algorithm initializes a table, denoted R_x, for each processed node x of the given UWSN. Table R_x has $|Loc(x)|$ rows. Each row provides a key-value mapping. The key corresponds to a possible location index, say i, of node x. The value $R_x[i]$ is the probability $p_x[i]$.
  Example: Figure 5.1 illustrates an example of two nodes x and y along with their locality sets and R tables.

- $R_{x,y}$: The algorithm computes for some pairs of nodes, e.g., a pair (x,y), a table denoted $R_{x,y}$. Table $R_{x,y}$ has at most $|Loc(x)|.|Loc(y)|$ rows.

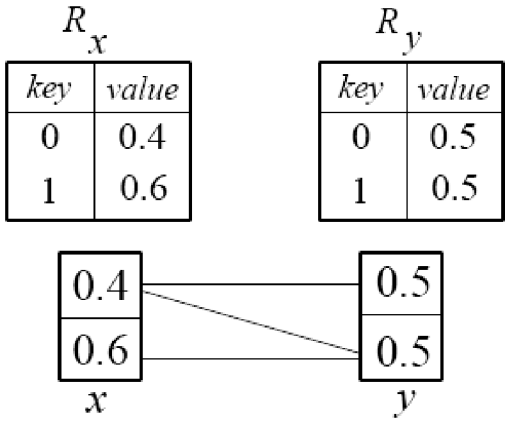


Figure 5.1: Example R_x and R_y tables

Each row provides a key-value mapping. The key corresponds to an event $(x[i], y[j])$ where nodes x and y are in locations $x[i]$ and $y[j]$, respectively. The value $R_{x,y}(i,j)$ is the probability that $x[i]$ reaches $y[j]$ by a path through some subset of nodes, as specified by the algorithm.

We next describe important operations on the above data structures.

- **Merge(R_x,R_y):** If node x is a neighbour of node y in a given underlying graph then the Merge function computes the reachability table $R_{x,y}$. The procedure is to extract the required information from the Cartesian product of the two key sets of the input tables R_x and R_y, as outlined in Algorithm 1. Figure 5.2 shows an example of the Merge operation.

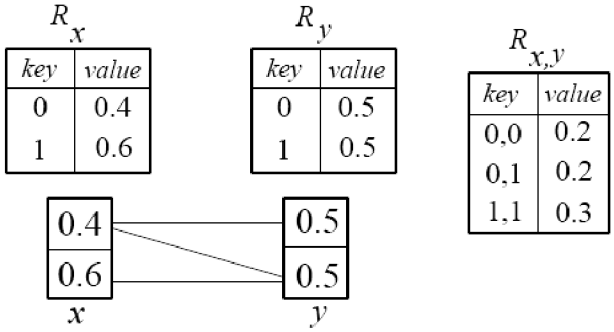


Figure 5.2: An example of a Merge operation between two nodes x and y

Running Time. We assume the use of a map data structure that can

Algorithm 1 Function Merge (R_x,R_y)

Input: tables R_x and R_y
Output: table $R_{x,y}$

1: Initialize table $R_{x,y}$ to empty
2: **foreach** (pair of keys $x[i] \in R_x$ and $y[j] \in R_y$) **do**
3: **if** (E($x[i]$,$y[j]$) == 1) **then**
4: store $p_x[i] \times p_y[j]$ in entry $R_{x,y}(x[i],y[j])$
5: **end if**
6: **end foreach**
7: return $R_{x,y}$

store and update any entry in constant time (e.g., a hash table). The function performs $O(|Loc(x)| \,.\, |Loc(y)|)$ iterations. denoting the size of the maximum locality set by ℓ_{max}, the function then runs in $O(\ell_{max}^2)$ time.

- **Merge ($R_{x,y}$, R_z):** If node x can reach node z via an intermediate node y then the Merge function computes the reachability table $R_{x,z}$. Similar to the above construction, the procedure is to extract the required information from the Cartesian product of the two key sets of the input tables $R_{x,y}$ and R_z, as outlined in Algorithm 2.

Algorithm 2 Function Merge ($R_{x,y}$,R_z)

Input: tables $R_{x,y}$ and R_z
Output: table $R_{x,z}$

1: Initialize table $R_{x,z}$ to empty
2: **foreach** (pair of keys $x[i], y[j] \in R_{x,y}$ and $z[k] \in R_z$) **do**
3: **if** (E($y[j]$,$z[k]$) == 1) **then**
4: add $R_{x,y}(x[i], y[j]) \times p_z[k]$ to entry $R_{x,z}(x[i],z[k])$
5: **end if**
6: **end foreach**
7: return $R_{x,z}$

Running Time. For any table R, we use $|R|$ to denote its length (i.e., number of keys). The loop performs $O(|R_{x,y}| \,.\, |R_z|)$ iterations. Thus, the function requires $O(\ell_{max}^3)$ time.

- **Iterative merge.** If $P = (x = x_1, x_2, \ldots, x_{q-1}, x_q = y)$ is a path of q nodes between the two end nodes x and y then by iteratively using the Merge functions, one can compute the reachability table $R_{x,y}$. For example, one can use the following sequence of function calls:
  - $R_{x_1,x_2} = \text{Merge}(R_{x_1}, R_{x_2})$
  - $R_{x_1,x_3} = \text{Merge}(R_{x_1,x_2}, R_{x_3})$
  - $\ldots$
  - $R_{x_1,x_q} = \text{Merge}(R_{x_1,x_{q-1}}, R_{x_q})$

  Running Time. Given a path on q nodes, the iterative merge performs $q-1$ merge operations. Thus, the steps require $O((q-1)\,.\,\ell_{max}^3)$ time, or $O(D\,.\,\ell_{max}^3)$ since $q \leq D$.

5.3 A Lower Bound Approach

In this section, we present a framework for obtaining lower bounds on the HB-$Conn_2$ problem. In Section 5.5, we outline the modifications necessary to deal with the DB-$Conn_2$ problem.

In general, given an instance (G, s, t, D) of the HB-$Conn_2$ problem, the approach relies on computing a lower bound on HB-$Conn_2(G, s, t, D)$ by first computing a subgraph $H \subseteq G$ composed of node disjoint (s,t)-paths and then computing HB-$Conn_2(H, s, t, D)$ exactly. We now outline the important ingredients of the framework.

1. Given an instance (G, s, t, D) of the HB-$Conn_2$ problem, we proceed by computing a subgraph $H \subseteq G$ composed of r, $r \geq 1$, node disjoint (s,t)-paths, where each path has length $\leq D$. We denote such a set of paths by $P^1, P^2, \ldots, P^r$. The kth path in the set has nodes
$$P^k = (s, first^k, second^k, \ldots, last^k, t)$$
.
We note that since $H \subseteq G$, HB-$Conn_2(H, s, t, D) \leq$ HB-$Conn_2(G, s, t, D)$. Thus, HB-$Conn_2(H, s, t, D)$ is a valid LB on HB-$Conn_2(G, s, t, D)$.

2. We now show how to compute HB-$Conn_2(H, s, t, D)$. The procedure is exact if the probabilities in the model G (and, consequently, H) are exact.

   (a) First, we use the iterative merge procedures in the previous section to compute for each path P^k the reachability table $R_{first^k,last^k}$ (denoted, R^k for short). Thus, HB-$Conn_2(H, s, t, D)$ can be computed from the tables

   $$R_s, R^1, R^2, \ldots, R^r, R_t \quad .$$

   (b) Second, for each table R^k corresponding to path P^k, $k \in [1, r]$, and for each pair of possible locations ($s[i]$,$t[j]$) of terminals s and t, we compute the probability, denoted Prob$^k s[i], t[j]$, that node $s[i]$ can reach node $t[j]$ by a route through path P^k where

   $$Prob^k(s[i], t[j]) = \sum \left(R^k(first^k[a], last^k[b]) : \begin{smallmatrix} s[i] \text{ reaches } first^k[a], \\ \text{and } last^k[b] \text{ reaches } t[j] \end{smallmatrix} \right) \tag{5.1}$$

   The above probability is conditional upon the event that nodes s and t occupy locations $s[i]$ and $t[j]$, respectively.

   (c) Third, for each pair of possible locations ($s[i]$,$t[j]$) of nodes s and t, we compute the probability that $s[i]$ can reach $t[j]$ through at least one of the r available paths, as follows

   $$R_{s,t}(s[i], t[j]) = p_s[i] \,.\, p_t[j] \,.\, \left(1 - \prod_{k=1}^{r} \left(1 - Prob^k(s[i], t[j]) \right) \right) \tag{5.2}$$

   (d) Fourth, since each of the used paths is selected so that it length$\leq D$, the desired solution is then

   $$\text{HB-}Conn_2(H, s, t, D) = \sum_{s[i] \in Loc(x), t[j] \in Loc(y)} R_{s,t}(s[i], t[j]) \tag{5.3}$$

Algorithm 3 presents a pseudo code for the function.

Running Time.

Algorithm 3 Function HB-Conn-LB

Input: a set $\{P^1, P^2, \ldots, P^r\}$ of (s,t)-paths in the underlying graph G
Output: HB-$Conn_2$ computed form the input

1: **foreach** $(k = 1, 2, \ldots, r)$ **do**
2: use the iterative merge procedure to compute table R^k
3: **foreach** (pair $(s[i], t[j]) \in Loc(s) \times Loc(t)$) **do**
4: use equation 5.1 to compute $Prob^k(s[i], t[j])$
5: **end foreach**
6: **end foreach**
7: **foreach** (pair $(s[i],t[j]) \in Loc(s) \times Loc(t)$) **do**
8: use equation 5.2 to compute $R_{s,t}(s[i], t[j])$
9: **end foreach**
10: return HB-$Conn_2$ (as computed by equation 5.3)

1. The function has two main loops. The first loop spans steps 1 through 6, and the second loop spans steps 7 through 9.

2. The first loop iterates r times. Each execution of Step 2 (the iterative merge) requires $O(D \,.\, \ell_{max}^3)$, since each processed path has at most D links. The inner loop of Steps 3 to 5 requires $O(\ell_{max}^2 \,.\, \ell_{max}^2)$ time. This follows since Step 3 iterates $O(\ell_{max}^2)$; each iteration evaluates equation 5.1 that examines $O(\ell_{max}^2)$ terms. Thus, the first loop requires $O(r \,.\, (D \,.\, \ell_{max}^3 + \ell_{max}^4))$ time.

3. The second loop (Steps 7 through 9) iterates $O(\ell_{max}^2)$ times. Each iteration evaluates equation 5.2 by computing r terms. Thus, the second loop requires $O(r \,.\, \ell_{max}^2)$ time.

4. Step 10 requires $O(\ell_{max}^2)$ time.

5. Thus, the running time of the function is dominated by the first loop that requires $O(r \,.\, (D \,.\, \ell_{max}^3 + \ell_{max}^4)$ time.

5.4 Revised Data Structures and Functions for the DB-$Conn_2$ Algorithm

In this section, we outline changes required to the material in section 5.2 to handle the DB-$Conn_2$ problem. We start by mentioning the following points.

For simplicity of analysis, we assume that the delay on each link $d(x[i],y[j])$ is an integer. Such a delay may be in units of fractions of a second (e.g., deci-, centi-, or milli-second) to satisfy a desired precision. The maximum required delay D is also an integer. Similar to the HB-$Conn_2$ problem, our approach here relies on using a set of node disjoint (s,t)- paths. However, unlike the HB-$Conn_2$ problem, each used (s,t)-path, say P= (s, $first$, $second$, ..., $last$, t), does not guarantee that each of its (s,t)-routes of the form ($s[i]$, $first[a]$, $second[b]$, ..., $last[c]$, $t[j]$) has a total delay $\leq D$. However, the merge functions discussed below are modified to exclude routes whose total delay exceed D. The revised data structures are presented below.

- R_x: For each node x, table R_x has the same structure as in Section 5.2.
- $R_{x,y}$: For some node pairs (x,y), the algorithm computes a reachability table, denoted $R_{x,y}$. Table $R_{x,y}$ has at most $|Loc(x)|\,.\,|Loc(y)|.D$ rows. Each row provides a key-value mapping. The key corresponds to an event $(x[i],y[j],d)$ where nodes x and y are in locations $x[i]$ and $y[j]$, respectively. The value $R_{x,y}(i,j,d)$ is the probability that $x[i]$ reaches $y[j]$ by a path of total delay d, through some subset of nodes, as specified by the algorithm. Care is taken so that each entry in each reachability table has a delay d $\leq D$.

  Example: Figure 5.3 illustrates an example of two nodes x and y along with their locality sets and R tables.

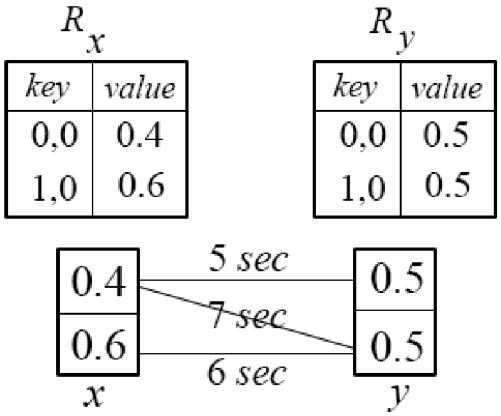


Figure 5.3: Example R_x and R_y tables

We next describe extensions of the functions in Section 5.2 to handle the current problem.

- **Merge(R_x, R_y):** If node x is a neighbour of node y in a given underlying graph then the Merge function computes the reachability table $R_{x,y}$. The revised function replaces Steps 3 and 4 in Algorithm 1 by steps 3 to 5 in Algorithm 4. The steps ignore delay communications $> D$. Figure 5.4 is an example of the Merge operation.

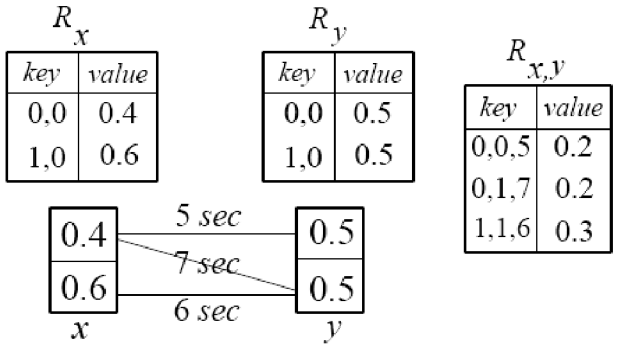


Figure 5.4: An example of a Merge operation between two nodes x and y

Running time. The revised function runs in $O(\ell^2_{max})$ time.

- **Merge($R_{x,y}, R_z$):** If node x can reach node z via an intermediate node y then the Merge function computes the reachability table $R_{x,z}$. The revised function replaces Steps 3 and 4 in Algorithm 2 by steps 3 to 6

Algorithm 4 Function Merge (R_x,R_y)

Input: tables R_x and R_y
Output: table $R_{x,y}$

```
1: Initialize table R_{x,y} to empty
2: foreach (pair of keys x[i] ∈ R_x and y[j] ∈ R_y) do
3:    if (d(x[i],y[j]) < D) then
4:       if (E(x[i],y[j]) == 1) then
5:          store p_x[i] × p_y[j] in entry R_{x,y}(x[i],y[j],d(x[i], y[j]))
6:       end if
7:    end if
8: end foreach
9: return R_{x,y}
```

in Algorithm 5. The steps compute the obtained delay from x to z, and ignore delay communication $> D$.

Algorithm 5 Function Merge ($R_{x,y}$,R_z)

Input: tables $R_{x,y}$ and R_z
Output: table $R_{x,z}$

```
1: Initialize table R_{x,z} to empty
2: foreach (pair of keys x[i], y[j] ∈ R_{x,y} and z[k] ∈ R_z) do
3:    d_{xz} = d_{xy} + d(y[j], z[k])
4:    if (d_{xz} < D) then
5:       if (E(y[j],z[k]) == 1) then
6:          add R_{x,y}(x[i], y[j]) × p_z[k] to entry R_{x,z}(x[i],z[k])
7:       end if
8:    end if
9: end foreach
10: return R_{x,z}
```

Running time. The computed table $R_{x,z}$ has length $O(D \,.\, \ell_{max}^2)$. The revised function runs in $O(D \,.\, \ell_{max}^3)$.

- **Iterative merge.** Similar to Section 5.2, if $P = (x = x_1, x_2, \ldots, x_{q-1}, x_q = y)$ is a path of q nodes between the two end nodes x and y then by iteratively using the Merge functions 4 and 5, one can compute the reachability table $R_{x,y}$.

  Running time. Given a path on q nodes, the steps require $O(D \,.\, q \,.\, \ell_{max}^3)$

time.

5.5 Revised Approach for the DB-$Conn_2$ Problem

We now outline the needed modifications to the approach in Section 5.3 to handle the DB-$Conn_2$ problem.

1. Given an instance (G, s, t, D) of the DB-$Conn_2$ problem, we compute a subgraph $H \subseteq G$ composed of r, $r \geq 1$, node disjoint (s,t)-paths. We recall that some routes obtained by some paths may have a delay $> D$. We denote such a set of paths by $P^1, P^2, \ldots, P^r$. The kth path in the set has nodes

   $$P^k = (s, first^k, second^k, \ldots, last^k, t)$$

   Similar to Section 5.3, we note that since $H \subseteq G$, DB-$Conn_2$ (H, s, t, D) $\leq$ DB-$Conn_2$ (G, s, t, D). Thus, DB-$Conn_2$ (H, s, t, D) is a valid LB on DB-$Conn_2$ (G, s, t, D).

2. We now show how to compute DB-$Conn_2$ (H, s, t, D).

   (a) First, we use the iterative merge procedures in the previous section to compute for each path P^k the reachability table $R_{first^k,last^k}$ (denoted, R^k for short). Thus, DB-$Conn_2$ (H, s, t, D) can be computed from the tables

   $$R_s, R^1, R^2, \ldots, R^r, R_t$$

   (b) Second, for each table R^k corresponding to path P^k, $k \in [1, r]$, and for each pair of possible locations ($s[i]$, $t[j]$) of terminals s and t, we compute the probability, denoted $Prob^k s[i], t[j]$, that node $s[i]$ can reach node $t[j]$ by a route of total delay at most D, through path P^k where

   $$Prob^k(s[i], t[j], D) = \sum \left(R^k(first^k[a], last^k[b], d) : \substack{s[i] \text{ to } t[j] \text{ is a route} \\ \text{with delay} < D} \right) \tag{5.4}$$

   The above probability is conditional upon the event that nodes s and t occupy locations $s[i]$ and $t[j]$, respectively.

(c) Third, for each pair of possible locations $(s[i], t[j])$ of nodes s and t, we compute the probability that $s[i]$ can reach $t[j]$ by a route of total delay $\leq D$, through at least one of the r available paths, as follows

$$R_{s,t}(s[i], t[j], D) = p_s[i] \,.\, p_t[j] \,.\, \left(1 - \prod_{k=1}^{r} \Big(1 - Prob^k(s[i], t[j], D)\Big)\right) \tag{5.5}$$

(d) Fourth, since table $R_{s,t}$ contains only paths of total delay $\leq D$, the desired solution is then

$$\text{DB-}Conn_2(H, s, t, D) = \sum R_{s,t}(s[i], t[j], D) \tag{5.6}$$

Algorithm 6 presents the pseudo code for the function.

Algorithm 6 Function DB-Conn-LB

Input: a set $P^1, P^2, \ldots, P^r$ of (s,t)-paths in the underlying graph G
Output: DB-$Conn_2$ computed form the input

1: **foreach** $(k = 1, 2, \ldots, r)$ **do**
2: use the iterative merge procedure to compute table R^k
3: **foreach** (pair $(s[i], t[j]) \in Loc(s) \times Loc(t)$) **do**
4: use equation 5.4 to compute $Prob^k(s[i], t[j], D)$
5: **end foreach**
6: **end foreach**
7: **foreach** (pair $(s[i],t[j]) \in Loc(s) \times Loc(t)$) **do**
8: use equation 5.5 to compute $R_{s,t}(s[i], t[j], D)$
9: **end foreach**
10: return DB-$Conn_2$ (as computed by equation 5.6)

Running time.

1. The function has two main loops. The first loop spans steps 1 through 6, and the second loop spans steps 7 through 9.

2. The first loop iterates r times. Each execution of Step 2 (the iterative merge) requires $O(D \,.\, q \,.\, \ell_{max}^3)$ to process a path of at most q nodes. The inner loop of Steps 3 and 4 requires $O(D \,.\, \ell_{max}^2 \,.\, \ell_{max}^2)$ time. This

follows since Step 3 iterates $O(\ell_{max}^2)$; each iteration evaluates equation 5.4 that examines $O(D \,.\, \ell_{max}^2)$ terms. Thus, the first loop requires $O(r \,.\, (D \,.\, q \,.\, \ell_{max}^3 + D \,.\, \ell_{max}^4)$ time.

3. The second loop (Steps 7 to 9) iterates $O(\ell_{max}^2)$ times. Each iteration evaluates equation 5.5 by computing r terms. Thus, the second loop requires $O(r \,.\, \ell_{max}^2)$ time.

4. Step 10 requires $O(\ell_{max}^2)$ time.

5. Thus, the running time of the function is dominated by the first loop that requires $O(r \,.\, (D \,.\, q \,.\, \ell_{max}^3 + D \,.\, \ell_{max}^4)$ time, where $q \leq$ the number of nodes in G.

5.6 Numerical Evaluation

In this section, we investigate the performance of our devised LB algorithms on sample probabilistic graphs. The algorithms are implemented in C++. The inputs and algorithms are described next.

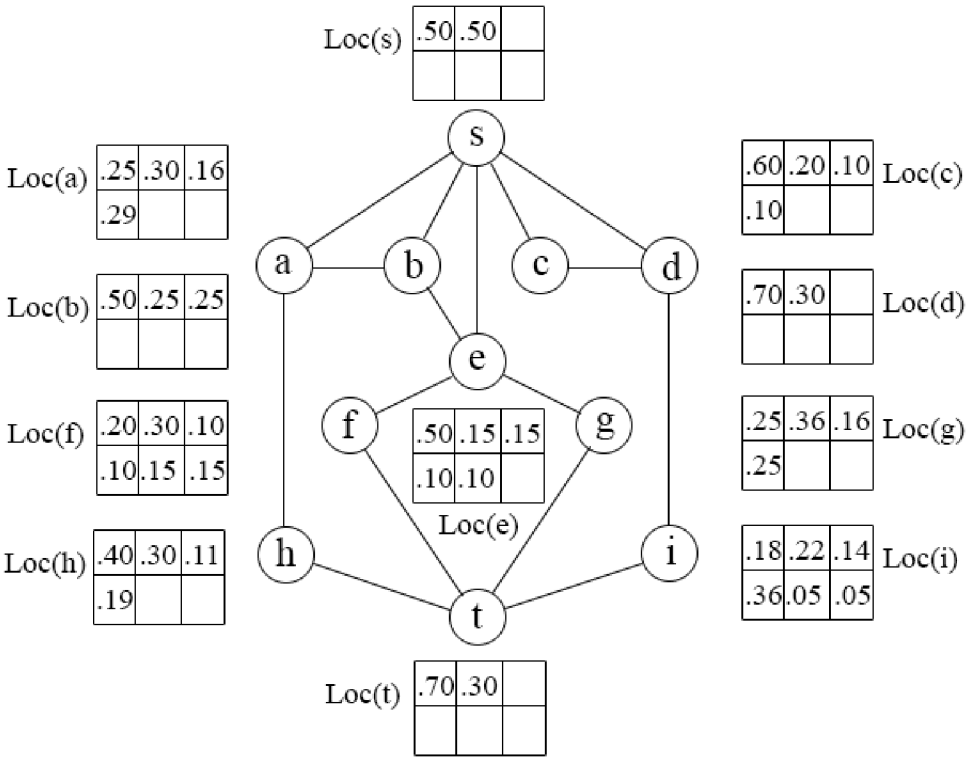
Test Networks. The following probabilistic graphs are used in our evaluation. Each graph has two distinguished terminals labelled s and t.

- G_{11} (Figure 5.5a): a sparse graph on 11 nodes. Each node has a locality set of size in the range [2,6].

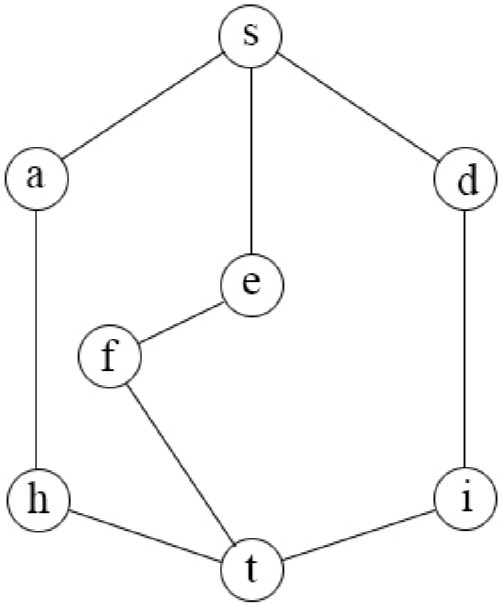
- G_{16} (Figure 5.6): a dense graph on 16 nodes. Each node has a locality set of size in the range [2,6].

- K_7 (Figure 5.8a): a complete graph on 7 nodes. Each node has a locality set of size 3.

Delays. For the DB-$Conn_2$ problem, we assign to each link of the form ($x[i]$,$y[j]$) a random delay $d(x[i], y[j])$ sampled from a uniform discrete distribution U(1 sec, 5 sec). As mentioned below, we use a shortest path algorithm in some computations. For this purpose, we set a weight to each link (x, y) of the underlying graphs G_{11}, G_{16}, and K_7. The weight of link (x, y) is the



(a) G_{11} (a sparse graph)



(b) node disjoint (s,t)-paths

Figure 5.5: (a) G_{11} (a sparse graph), (b) node disjoint (s,t)-paths

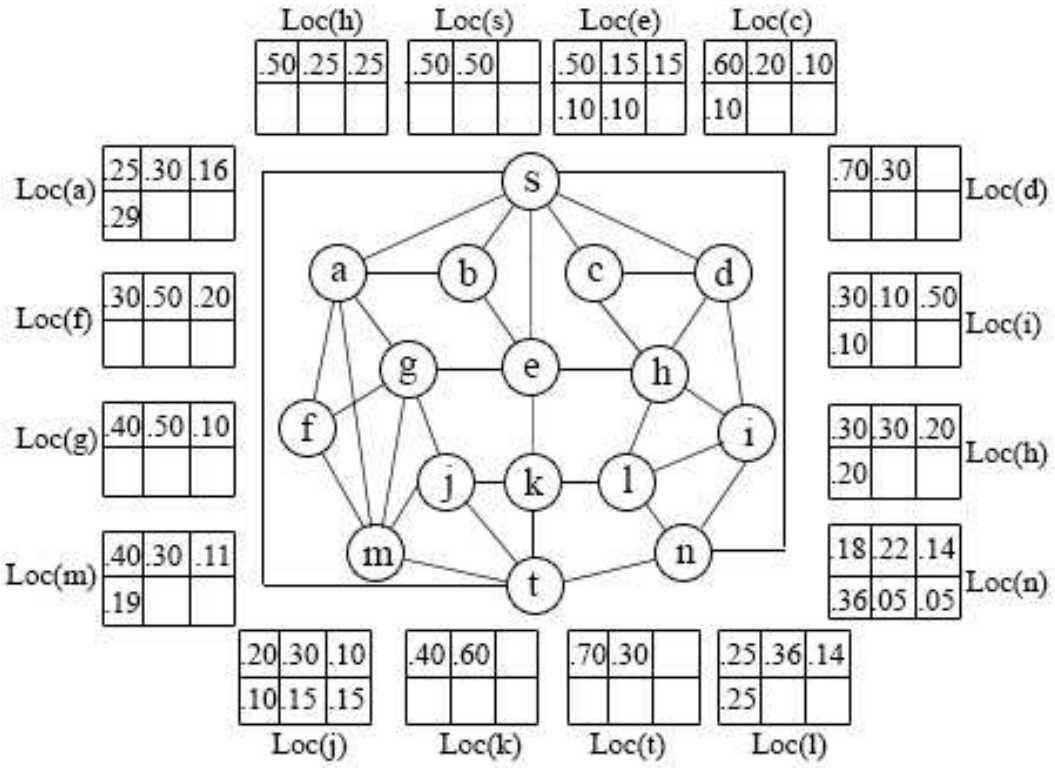


Figure 5.6: G_{16}(a dense graph)

$\sum \Big(d(x[i], y[j]) : x[i] \in Loc(x) \text{ and } y[j] \in Loc(y) \Big)$ divided by the number of links between x and y.

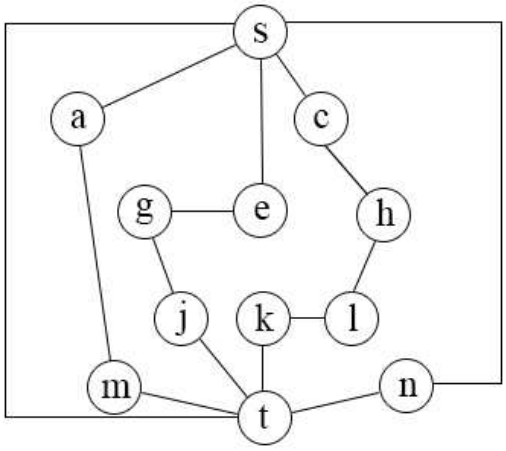
Algorithms. We compare the following algorithms in our investigation.

- **Exact algorithm.** As explained in Section 5.1, an exact algorithm generates all possible states of the underlying graph, and evaluates each state to decide whether it operates or fails. Taking the graph G_{11} as an example, the algorithm generates $2^3 \cdot 3^1 \cdot 4^4 \cdot 5^1 \cdot 6^2 \approx 1.1$ million states.

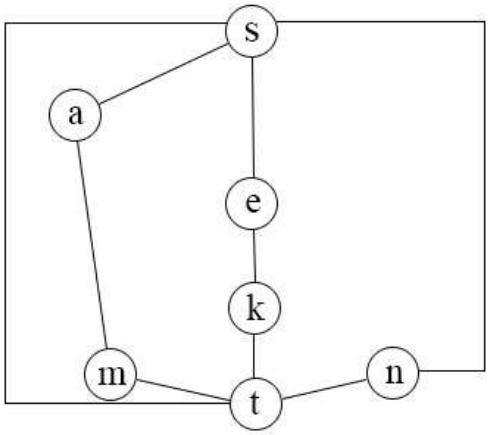
- **Monte Carlo (MC) sampling algorithm.** The algorithm generates $N = 100,000$ Random states (according to the probabilities of the node locality sets). Each state is classified as either operating or failed. If N_{op} is the number of operating states, the algorithm returns the fraction $\frac{N_{op}}{N}$.

- **The HB-$Conn_2$ (and DB-$Conn_2$) LB algorithms.** We use the following two methods for generating the required node disjoint (s,t)-paths.

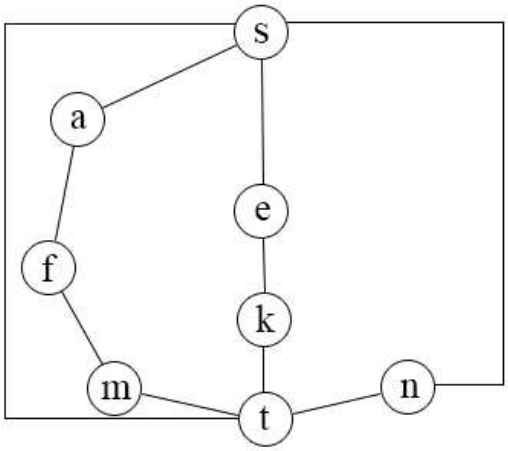
  - **Maximum flows (MF):** We assign to each link (x, y) in the un-



(a) MF node disjoint (s, t)-paths

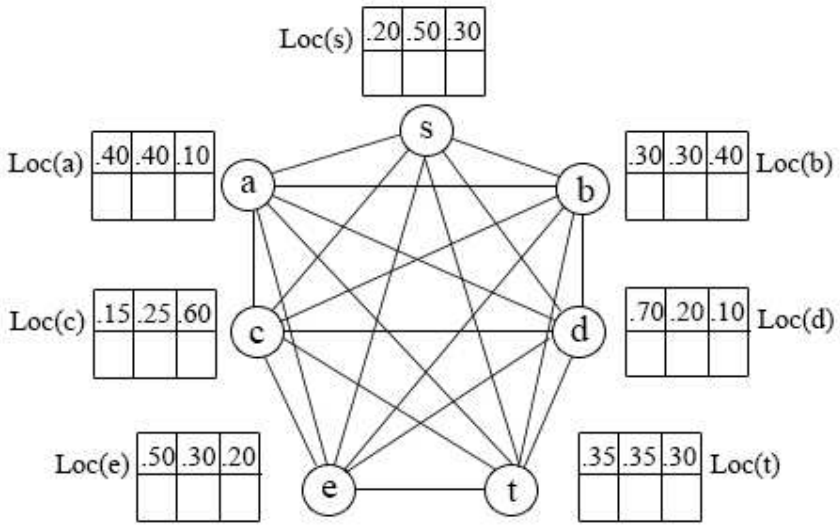


(b) SP node disjoint (s, t)-paths (Hop)

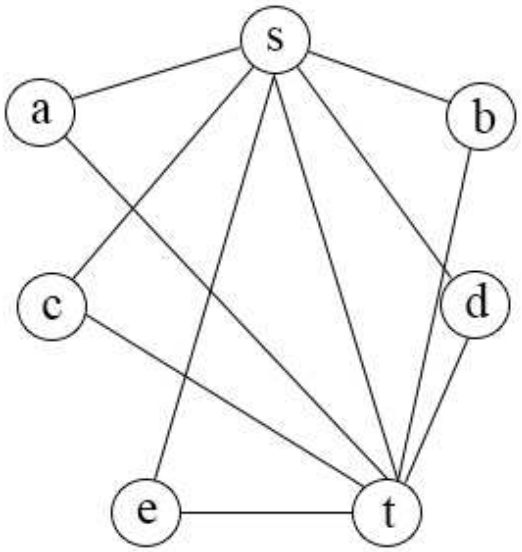


(c) SP node disjoint (s, t)-paths (Delay)

Figure 5.7: (a) MF node disjoint (s, t)-paths, (b) SP node disjoint (s, t)-paths , (c) SP node disjoint (s, t)-paths



(a) K_7 (a complete graph)



(b) node disjoint (s, t)-paths

Figure 5.8: (a) K_7 (a complete graph), (b) node disjoint (s, t)-paths

derlying graph a capacity = 1 unit. We then run a maximum flow algorithm to compute a maximum possible set of node disjoint (s,t)-paths. The procedure is used for both the HB-$Conn_2$ problem and the DB-$Conn_2$ problem.

– **Shortest paths (SP):** For the HB-$Conn_2$ problem, each link (x,y) in the underlying graph has unit weight. For the DB-$Conn_2$ problem, each link (x,y) in the underlying graph is assigned a weight equals to the $\frac{\sum\left(d(x[i],y[j]):x[i]\in Loc(x) \text{ and } y[j]\in Loc(y)\right)}{n_l}$ where n_l is the number of links between x and y. We then iteratively identify a shortest (s,t)-path and then delete the path until we cannot identify more paths.

The resulting sets of node disjoint (s,t)-paths are as follows:

- For G_{11}, the two sets of MF and SP disjoint (s,t)-paths are identical to the set shown in Figure 5.5b.
- For G_{16}, the MF and SP (for HB-$Conn_2$ and DB-$Conn_2$) paths are illustrated in Figure 5.7a, 5.7b, and 5.7c respectively.
- For K_7, the two sets of MF and SP disjoint paths are identical to the set shown in Figure 5.8b.

1. **Running Time Results**

   All algorithms are run on a laptop with 2.2 GHz Intel CPU, and 8 GByte main memory. The maximum observed running times are as follows: Exact algorithm: in the order of hours, MC algorithm: in the order of seconds, HB-$Conn_2$ algorithm: two seconds or less, and DB-$Conn_2$ algorithm: two seconds or less.

2. **HB-$Conn_2$ Results**

   Figure 5.9 (for G_{11}), Figure 5.10 (for G_{16}), and 5.11 (for K_7) present the obtained results of the four algorithms: Exact, MC, HB-$Conn_2$ with MF paths, and HB-$Conn_2$ with SP paths. In each figure, the x-axis is the

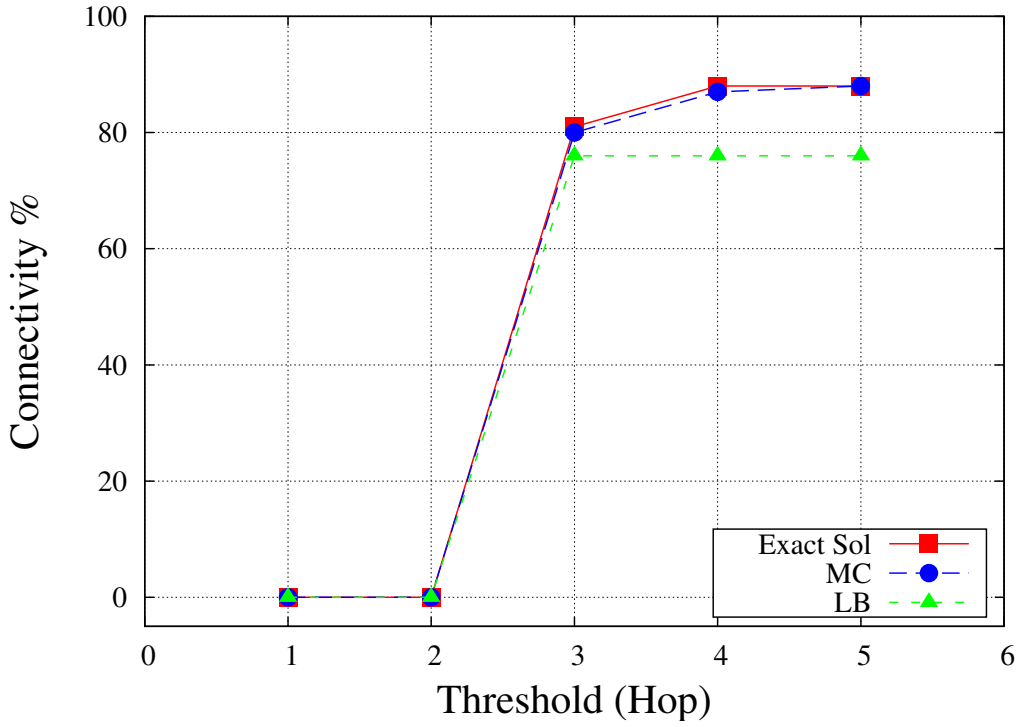


Figure 5.9: HB-$Conn_2$ (G_{11}) results for $D \in [1, 5]$

parameter $D \in [1, 5]$, and the y-axis is the HB-$Conn_2$ probability. We observe the following:

- As expected, the HB-$Conn_2$ measure takes non-decreasing values as the permitted hop bound D increases.
- The MC algorithm gives a good approximation to the exact value, although it can unpredictably overestimate or underestimate the exact value.
- The obtained LB is at most 15% lower than the exact solution.
- There is no clear winner among the MF and SP methods for computing the node disjoint paths.

3. **DB-$Conn_2$ Results**

   Figure 5.12 (for G_{11}), Figure 5.13 (for G_{16}), and 5.14 (for K_7) present the obtained results of the four algorithms: Exact, MC, DB-$Conn_2$ with MF paths, and DB-$Conn_2$ with SP paths. In each figure, the x-axis is

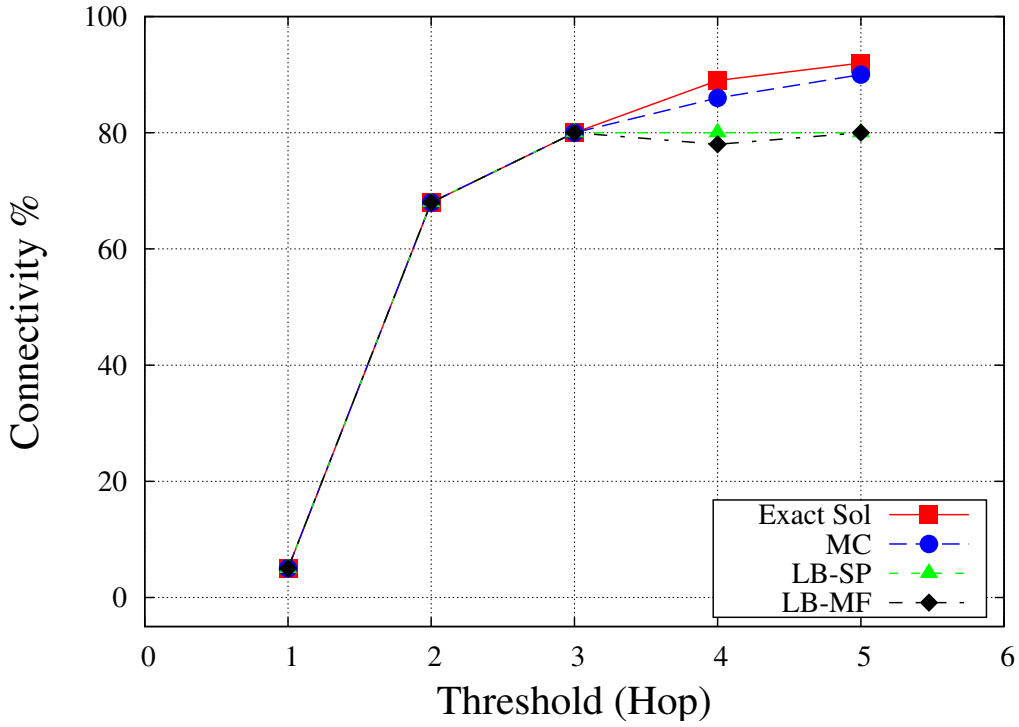


Figure 5.10: HB-$Conn_2$ (G_{16}) results for $D \in [1, 5]$

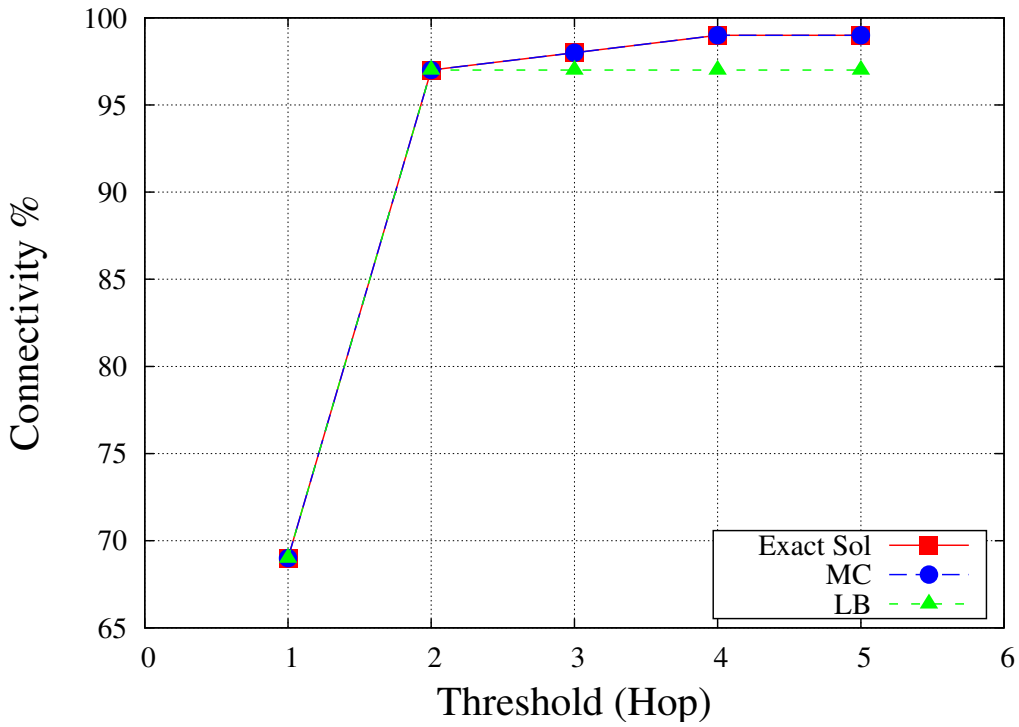


Figure 5.11: HB-$Conn_2$ (K_7) results for $D \in [1, 5]$

the parameter $D \in [6, 10]$, and the y-axis is the DB-$Conn_2$ probability. We observe the following:

- As expected, the DB-$Conn_2$ measure takes non-decreasing values as the permitted delay bound D increases.
- The MC algorithm gives a good approximation to the exact value, although it can unpredictably overestimate or underestimate the exact value.
- The obtained LB is at most 15% lower than the exact solution.
- The SP method produces better results than the MF method.

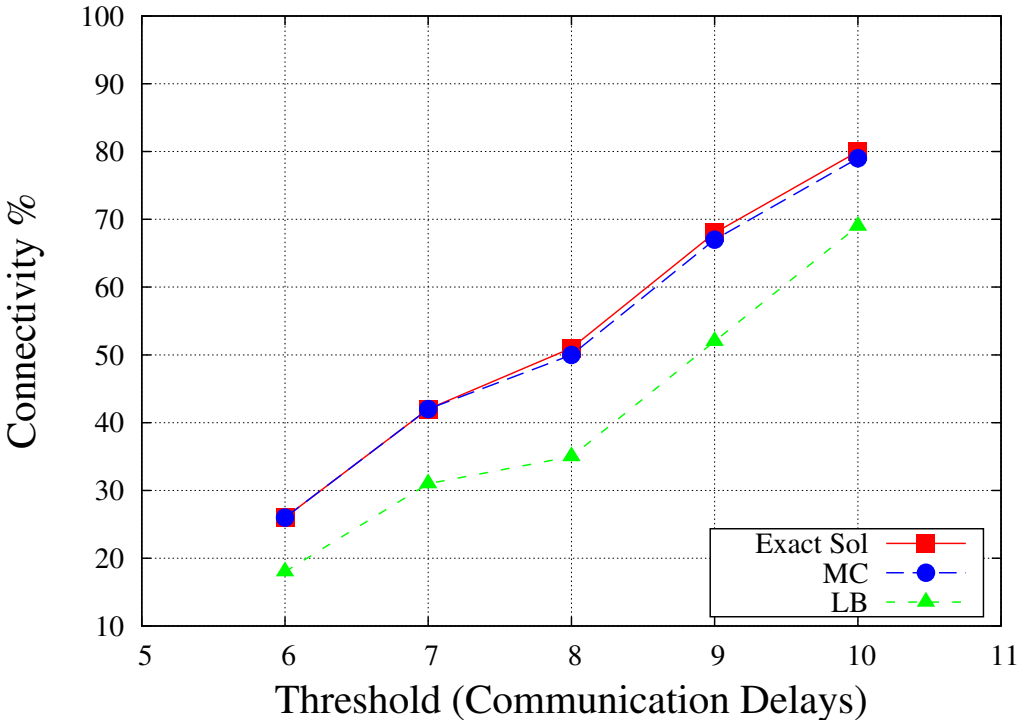


Figure 5.12: DB-$Conn_2$ (G_{11}) results for $D \in [6, 10]$

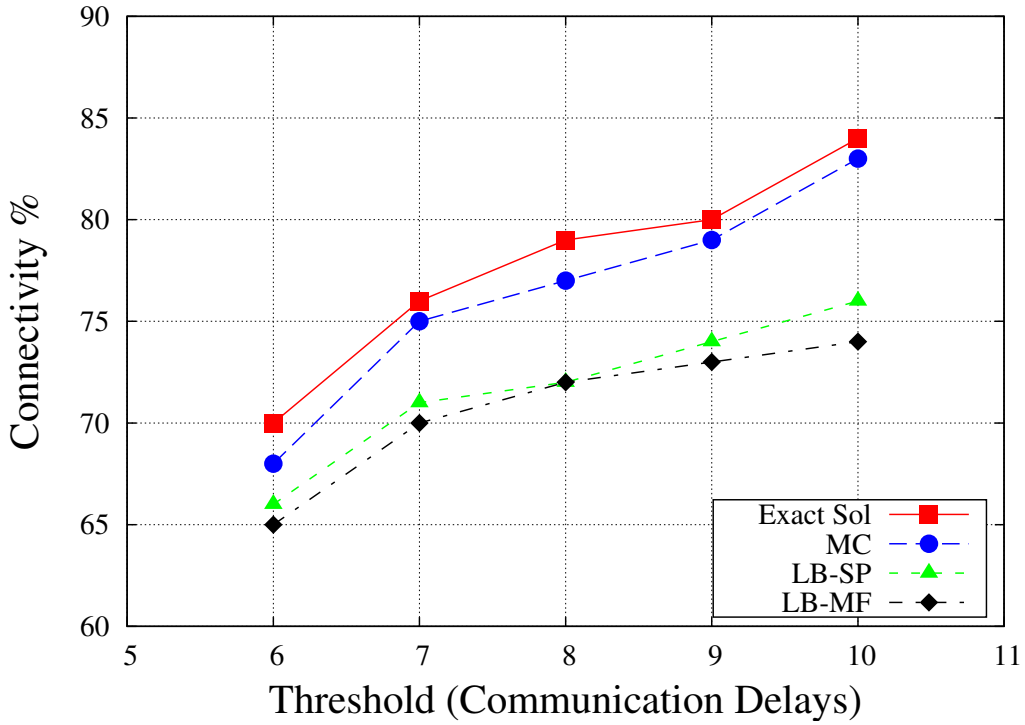


Figure 5.13: DB-$Conn_2$ (G_{16}) results for $D \in [6, 10]$

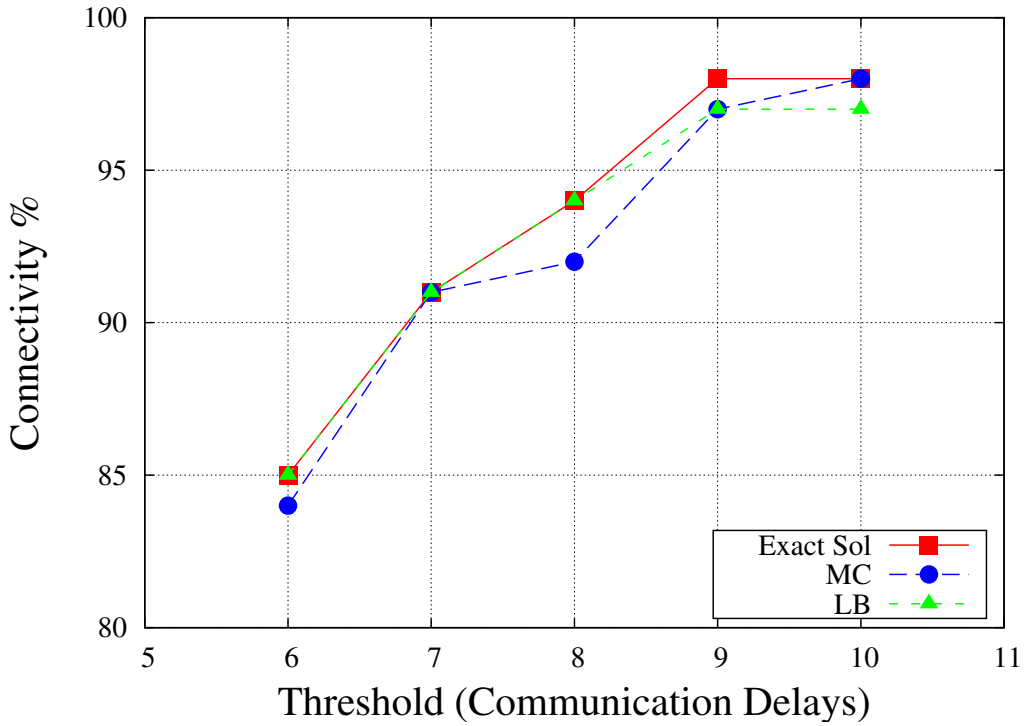


Figure 5.14: DB-$Conn_2$ (K_7) results for $D \in [6, 10]$

5.7 Concluding Remarks

In this chapter, we have formalized two 2-terminal reachability problems, called the HB-$Conn_2$ and the DB-$Conn_2$ problems. The two terminals are denoted s and t. The formulations are based on using the probabilistic graph model to capture uncontrollable node mobility. We have developed an approach to derive lower bounds on the exact solutions of the two problems.

Our devised approach belongs to the class of subgraph bounding approaches, described in [24] for network reliability problems. In particular, for an input probabilistic graph G, the approach first computes a subgraph $H \subseteq G$ composed of one, or more, node disjoint (s,t)-paths. We then apply the dynamic programming algorithms devised in the chapter to compute the exact values of HB-$Conn_2(H)$, or DB-$Conn_2(H)$. The obtained results serve as lower bounds on the desired HB-$Conn_2(G)$, or DB-$Conn_2(G)$ solutions.

Chapter 6

Concluding Remarks

The thesis has made contributions to solving the following problems. First, in Chapter 3, we have considered the problem of identifying RFID tags placed on a moving conveyor belt. Our aim is to devise effective anti-collision schemes for the problem. Our work has considered the interaction among the following 4 types of problem parameters:

- conveyor belt parameters (represented by the conveyor belt speed),
- tag placement parameters (represented by the inter-tag distances distribution),
- reader placement parameters (represented by the length of the interrogation zone), and
- anti-collision protocol parameters (represented by the maximum number K of slot counter values).

We have devised two schemes (the S-READER and the D-READER schemes) for the identification process. For each scheme, we obtained bounds on the anti-collision protocol parameter K that aim at achieving good performance. In addition, we have investigated the performance of each devised scheme by simulation.

Second, in Chapter 5, we have considered two reachability problems for UWSNs where nodes have uncontrollable mobility. The problems call for estimating the likelihood that a given node s can reach another given node t by a

path of bounded number of links or delay. We have developed an approach for computing lower bounds on the required reachability measure. Our approach models the given UWSN with a probabilistic graph G. The approach then proceeds by computing a subgraph of G composed of a number of node disjoint (s, t)-paths. We then apply the algorithms devised in Chapter 5 to compute the exact reachability measure on the computed subgraph. The obtained solutions form lower bounds on the desired solutions. We have also presented numerical evaluation results that show the quality of the obtained results relative to results obtained by using exhaustive enumeration exact algorithms and Monte Carlo sampling algorithms.

For future research, we propose the following problems as possible extensions to the thesis work.

- For the problem of identifying RFID tags placed on a moving conveyor belt, we mention the following directions.
  - Our analytical results have considered a single type of collision events where collision occurs between two tags that respond to the first heard query command. Other types of collision events, and events where a tag exit the interrogation zone before responding to a read command, exist. Thus, our analytical results can be strengthened by considering such new events.
  - Our analysis of the devised protocols assumes that consecutive tags use consecutive slot counter values (modulo the parameter K). Allowing consecutive tags to have randomly chosen slot counter values results in more flexible settings. More work is needed to analyze such general settings.
  - Our work utilizes framed slotted-Aloha based anti-collision protocols. No result seems to exist on adapting tree-based algorithms to solve the conveyor belt problem. More work can be done in this direction too.

- For the HB-$Conn_2$ and DB-$Conn_2$ problems, we propose to extend the devised lower bounding approach to use *series-parallel* subgraphs instead of node disjoint paths. A graph is series-parallel with respect to a given pair (s, t) of nodes if it can be reduced to a single edge (s, t) by repeatedly performing series reductions of degree 2 nodes (other than s and t), and parallel edge reductions. The class of (s, t)-series-parallel graphs includes as a proper subset graphs composed of a number of node disjoint (s, t)-paths. Hence, series-parallel subgraph bounds improve on node disjoint paths bounds. This direction calls for developing efficient exact algorithms to solve the HB-$Conn_2$ and DB-$Conn_2$ problems on (s, t)-series parallel graphs.

Bibliography

[1] EPCglobal. EPC radio-frequency identity protocols class-1 generation-2UHF RFID protocol for communications at 860 mhz-960 mhz version 2.0.0, 2013.

[2] I. F. Akyildiz, D. Pompili, and T. Melodia. Underwater acoustic sensor networks: research challenges. *Ad hoc networks*, 3(3):257–279, 2005.

[3] S. Alam and Z. J. Haas. Coverage and connectivity in three-dimensional underwater sensor networks. *Wireless Communications and Mobile Computing*, 8(8):995–1009, 2008.

[4] K. Ali, H. Hassanein, and A.-E. M. Taha. RFID anti-collision protocol for dense passive tag environments. In *the 32nd IEEE Conference on Local Computer Networks (LCN)*, pages 819–824. IEEE, 2007.

[5] S. Aroor and D. Deavours. Evaluation of the state of passive UHF RFID: An experimental approach. *IEEE Systems Journal*, 1(2):168, 2007.

[6] M. Ayaz, I. Baig, A. Abdullah, and I. Faye. A survey on routing techniques in underwater wireless sensor networks. *Journal of Network and Computer Applications*, 34(6):1908–1927, 2011.

[7] X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. H. Lai. Deploying wireless sensors to achieve both coverage and connectivity. In *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 131–I142. ACM, 2006.

[8] Y. Bayrakdar, N. Meratnia, and A. Kantarci. A comparative view of routing protocols for underwater wireless sensor networks. In *the IEEE OCEANS*, pages 1–5. IEEE, 2011.

[9] Z. Bin, M. Kobayashi, and M. Shimizu. Framed ALOHA for multiple RFID objects identification. *IEICE Transactions on Communications*, 88(3):991–999, 2005.

[10] A. G. Bluman. *Elementary Statistics : A Step by Step Approach*. McGraw-Hill, 2008.

[11] M. A. Bonuccelli, F. Lonetti, and F. Martelli. Tree slotted ALOHA: a new protocol for tag identification in RFID networks. In *Proceedings of the International Symposium on World of Wireless, Mobile and Multimedia Networks*, pages 603–608. IEEE Computer Society, 2006.

[12] A. Bower and T. Rossby. Evidence of cross-frontal exchange processes in the gulf stream based on isopycnal rafos float data. *Journal of Physical Oceanography*, 19(9):1177–1190, 1989.

[13] A. S. Bower. A simple kinematic mechanism for mixing fluid parcels across a meandering jet. *Journal of Physical Oceanography*, 21(1):173–180, 1991.

[14] M. Buettner and D. Wetherall. An empirical study of UHF RFID performance. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 223–234. ACM, 2008.

[15] C. Campolo, A. Molinaro, and R. Scopigno. *Vehicular Ad Hoc Networks: Standards, Solutions, and Research*. Springer, 2015.

[16] J. Capetanakis. Tree algorithms for packet broadcast channels. *IEEE Transactions on Information Theory*, 25(5):505–515, 1979.

[17] A. Caruso, F. Paparella, L. F. Vieira, M. Erol, and M. Gerla. The meandering current mobility model and its impact on underwater mobile sensor networks. In *the IEEE 27th Conference on Computer Communications (INFOCOM)*. IEEE, 2008.

[18] V. Chandrasekhar, W. K. Seah, Y. S. Choo, and H. V. Ee. Localization in underwater sensor networks: survey and challenges. In *Proceedings of the 1st ACM international workshop on Underwater networks*, pages 33–40. ACM, 2006.

[19] H. Chang. Underwater wireless sensor networks. In *Oceans*, pages 1–5. IEEE, 2012.

[20] K. Chen, M. Ma, E. Cheng, F. Yuan, and W. Su. A survey on MAC protocols for underwater wireless sensor networks. *IEEE Communications Surveys and Tutorials*, 16(3):1433–1447, 2014.

[21] W. Chen and G. Lin. An efficient anti-collision method for tag identification in a RFID system. *IEICE Transactions on Communications E Series B*, 89(12):3386 – 3392, 2006.

[22] H. H. Cho, C. Y. Chen, T. K. Shih, and H. C. Chao. Survey on underwater delay/disruption tolerant wireless sensor network routing. *IET Wireless Sensor Systems*, 4(3):112–121, 2014.

[23] S. Climent, A. Sanchez, J. V. Capella, N. Meratnia, and J. J. Serrano. Underwater acoustic wireless sensor networks: advances and future trends in physical, MAC and routing layers. *Sensors*, 14(1):795–833, 2014.

[24] C. J. Colbourn. *The combinatorics of network reliability*. Oxford University Press, 1987.

[25] D. M. Dobkin. *The RF in RFID:UHF RFID in Practice*. Newnes, 2012.

[26] M. Erol, L. F. Vieira, and M. Gerla. Localization with dive'n'rise (dnr) beacons for underwater acoustic sensor networks. In *Proceedings of the second workshop on Underwater networks*, pages 97–100. ACM, 2007.

[27] M. Erol-Kantarci, S. Oktug, L. Vieira, and M. Gerla. Performance evaluation of distributed localization techniques for mobile underwater acoustic sensor networks. *Ad Hoc Networks*, 9(1):61–72, 2011.

[28] A. C. Farrell and J. Peng. Performance of ieee 802.11 MAC in underwater wireless channels. *Procedia Computer Science*, 10:62–69, 2012.

[29] B. Feng, L. Jin-tao, G. Jun-bo, and D. Zhen-Hua. Id-binary tree stack anticollision algorithm for RFID. In *Proceedings of the 11th IEEE Symposium on Computers and Communications (ISCC)*, pages 207–212. IEEE, 2006.

[30] K. Finkenzeller and D. Muller. *RFID Handbook: Radio-frequency identification fundamentals and applications. 3rd edition*. Wiley, 2010.

[31] C. Floerkemeier. Transmission control scheme for fast RFID object identification. In *the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom)*, pages 6–pp. IEEE, 2006.

[32] C. Forbes, M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions*. John Wiley and Sons, 2010.

[33] A. Ghosh and S. K. Das. Coverage and connectivity issues in wireless sensor networks: A survey. *Pervasive and Mobile Computing*, 4(3):303–334, 2008.

[34] C. Giantsis and A. A. Economides. Comparison of routing protocols for underwater sensor networks: a survey. *International Journal of Communication Networks and Distributed Systems*, 7(3):192–228, 2011.

[35] A. Gkikopouli, G. Nikolakopoulos, and S. Manesis. A survey on underwater wireless sensor networks and applications. In *the 20th Mediterranean Conference on Control and Automation (MED)*, pages 1147–1154. IEEE, 2012.

[36] Z. Guo, Z. Peng, B. Wang, J.-H. Cui, and J. Wu. Adaptive routing in underwater delay tolerant sensor networks. In *the 6th International ICST Conference on Communications and Networking in China (CHINACOM)*, pages 1044–1051. IEEE, 2011.

[37] Z. Guo, B. Wang, and J.-H. Cui. Prediction assisted single-copy routing in underwater delay tolerant networks. In *the IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1–6. IEEE, 2010.

[38] M. Hayajneh, I. Khalil, and Y. Gadallah. An OFDMA-based MAC protocol for under water acoustic wireless sensor networks. In *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*, pages 810–814. ACM, 2009.

[39] J. Heidemann, M. Stojanovic, and M. Zorzi. Underwater sensor networks: applications, advances and challenges. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 370(1958):158–175, 2012.

[40] T. Hu and Y. Fei. An adaptive and energy-efficient routing protocol based on machine learning for underwater delay tolerant networks. In *the IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 381–384. IEEE, 2010.

[41] D. R. Hush and C. Wood. Analysis of tree algorithms for RFID arbitration. In *IEEE International Symposium on Information Theory*, pages 107–107, 1998.

[42] M. A. Islam. Probabilistic connectivity of underwater sensor networks. Master's thesis, University of Alberta, 2014.

[43] M. A. Islam and E. S. Elmallah. Tree bound on probabilistic connectivity of underwater sensor networks. In *the IEEE 39th Conference on Local Computer Networks Workshops (the WLN Workshop)*, pages 746–752. IEEE, 2014.

[44] D. K. Klair, K.-W. Chin, and R. Raad. A survey and tutorial of RFID anti-collision protocols. *IEEE Communications Surveys and Tutorials*, 12(3):400–421, 2010.

[45] U. Lee, P. Wang, Y. Noh, L. F. Vieira, M. Gerla, and J.-H. Cui. Pressure routing for underwater sensor networks. In *Proceedings of the IEEE INFOCOM*, pages 1–9. IEEE, 2010.

[46] E. Magistretti, J. Kong, U. Lee, M. Gerla, P. Bellavista, and A. Corradi. A mobile delay-tolerant approach to long-term energy-efficient underwater sensor networking. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2866–2871. IEEE, 2007.

[47] P. Mohapatra. *Ad Hoc Networks: Technologies and Protocols*. Springer Science and Business Media, 2005.

[48] J. Myung and W. Lee. Adaptive binary splitting: a RFID tag collision arbitration protocol for tag identification. *Mobile networks and applications*, 11(5):711–722, 2006.

[49] J. Myung, W. Lee, and T. K. Shih. An adaptive memoryless protocol for RFID tag collision arbitration. *IEEE Transactions on Multimedia*, 8(5):1096–1101, 2006.

[50] V. Namboodiri and L. Gao. Energy-aware tag anticollision protocols for RFID systems. *IEEE Transactions on Mobile Computing*, 9(1):44–59, 2010.

[51] J. Partan, J. Kurose, and B. N. Levine. A survey of practical issues in underwater networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 11(4):23–33, 2007.

[52] C. Petrioli, R. Petroccia, and M. Stojanovic. A comparative performance evaluation of MAC protocols for underwater sensor networks. In *the IEEE OCEANS*, pages 1–10. IEEE, 2008.

[53] D. Pompili, T. Melodia, and I. F. Akyildiz. Routing algorithms for delay-insensitive and delay-sensitive applications in underwater sensor networks. In *Proceedings of the 12th annual international conference on Mobile computing and networking*, pages 298–309. ACM, 2006.

[54] D. Pompili, T. Melodia, and I. F. Akyildiz. A distributed cdma medium access control for underwater acoustic sensor networks. In *Proc. of Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, pages 63–70, 2007.

[55] C. S. Raghavendra, K. M. Sivalingam, and T. Znati. *Wireless sensor networks*. Springer, 2006.

[56] K. M. Ramakrishnan and D. D. Deavours. Performance benchmarks for passive UHF RFID tags. *In Proceedings of the 13th GI/ITG Conference on Measurement, Modeling, and Evaluation of Computer and Communications Systems*, pages 137–154, 2006.

[57] T. S. Rappaport. *Wireless communications: principles and practice*, volume 2. Prentice Hall PTR New Jersey, 1996.

[58] V. Rodoplu and M. K. Park. An energy-efficient MAC protocol for underwater wireless acoustic networks. In *Proceedings of the MTS/IEEE OCEANS*, pages 1198–1203. IEEE, 2005.

[59] G. Roussos. *Networked RFID: systems, software and services*. Springer Science and Business Media, 2008.

[60] J. Ryu, H. Lee, Y. Seok, T. Kwon, and Y. Choi. A hybrid query tree protocol for tag collision arbitration in RFID systems. In *the IEEE International Conference on Communications (ICC)*, pages 5981–5986. IEEE, 2007.

[61] D. H. Shih, P. L. Sun, D. C. Yen, and S. M. Huang. Taxonomy and survey of RFID anti-collision protocols. *Computer communications*, 29(11):2150–2166, 2006.

[62] H.-P. Tan, R. Diamant, W. K. Seah, and M. Waldmeyer. A survey of techniques and challenges in underwater localization. *Ocean Engineering*, 38(14):1663–1676, 2011.

[63] H. Vogt. Efficient object identification with passive RFID tags. In *Pervasive computing*, pages 98–113. Springer, 2002.

[64] J. Wang, Y. Zhao, and D. Wang. A novel fast anti-collision algorithm for RFID systems. In *the IEEE International Conference on Wireless Communications, Networking and Mobile Computing (WiCom)*, pages 2044–2047. IEEE, 2007.

[65] P. Xie and J.-H. Cui. R-MAC: An energy-efficient MAC protocol for underwater sensor networks. In *International Conference on Wireless Algorithms, Systems and Applications (WASA)*, pages 187–198. IEEE, 2007.

[66] H. Yan, Z. J. Shi, and J.-H. Cui. Dbr: depth-based routing for underwater sensor networks. In *NETWORKING 2008 Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*, pages 72–86. Springer, 2008.

[67] M. A. Yigitel, O. D. Incel, and C. Ersoy. Qos-aware MAC protocols for wireless sensor networks: A survey. *Computer Networks*, 55(8):1982–2004, 2011.

[68] S. Yu, Y. Zhan, Z. Wang, and Z. Tang. Anti-collision algorithm based on jumping and dynamic searching and its analysis. *Computer Engineering*, 31(9):19–20, 2005.

[69] F. Yunus, S. H. Ariffin, and Y. Zahedi. A survey of existing medium access control (MAC) for underwater wireless sensor network (UWSN). In *the Fourth Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation (AMS)*, pages 544–549. IEEE, 2010.

[70] H. Zhang, L. Han, and Y.-L. Li. Design of hash-tree anti-collision algorithm. In *the Third International Conference on Natural Computation (ICNC)*, volume 5, pages 176–179. IEEE, 2007.

[71] S. Zhang, D. Li, and J. Chen. A link-state based adaptive feedback routing for underwater acoustic sensor networks. *IEEE Sensors Journal*, 13(11):4402–4412, 2013.

[72] Z. Zhou, J.-H. Cui, and S. Zhou. Efficient localization for large-scale underwater sensor networks. *Ad Hoc Networks*, 8(3):267–279, 2010.

[73] Z. Zhou, Z. Peng, J.-H. Cui, Z. Shi, and A. C. Bagtzoglou. Scalable localization with mobility prediction for underwater sensor networks. *IEEE Transactions on Mobile Computing*, 10(3):335–348, 2011.

[74] C. Zhu, C. Zheng, L. Shu, and G. Han. A survey on coverage and connectivity issues in wireless sensor networks. *Journal of Network and Computer Applications*, 35(2):619–632, 2012.

[75] L. Zhu and T. S. P. Yum. A critical survey and analysis of RFID anti-collision mechanisms. *IEEE Communications Magazine*, 49(5):214–221, 2011.