The University of Liverpool

Doctoral Thesis

Navigation Problems for Autonomous Robots in Distributed Environments

Author:
Thomas Gorry

Supervisors:
Dr. Russell Martin
Prof. Leszek Gąsieniec

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

in the

Complexity Theory and Algorithms Group
Department of Computer Science

June 17, 2015

“Computer science is no more about computers than astronomy is about telescopes."

Edsger Dijkstra

THE UNIVERSITY OF LIVERPOOL

Abstract

Faculty of Science and Engineering
Department of Computer Science

Doctor of Philosophy

Navigation Problems for Autonomous Robots in Distributed Environments

by Thomas Gorry

This thesis studies algorithms for Distributed Computing. More specifically however the project aimed to carry out research on the performance analysis of mobile robots in a variety of different settings. In a range of different network and geometric settings we investigate efficient algorithms for the robots to perform given tasks. We looked at a variety of different models when completing this work but focused mainly on cases where the robots have limited communication mechanisms. Within this framework we investigated cases where the robots were numerous to cases where they were few in number. Also we looked at scenarios where the robots involved had different limitations on the maximal speeds they could travel.
When conducting this work we explored two main tasks carried out by the robots that became the primary theme of the study. These two main tasks are *Robot Location Discovery* and *Robot Evacuation*. To accomplish these tasks we constructed algorithms that made use of both randomised and deterministic approaches in their solutions.

Acknowledgements

I would like to express my gratitude to my supervisors, Dr. Russell Martin and Prof. Leszek, whose guidance, knowledge and patience added greatly to my experience as a PhD student. I would also like to thank my academic advisors, Dr. Prudence W.H. Wong, Dr. Martin Gairing and Dr. Igor Potapov for their time and feedback during my studies.

I must also acknowledge all of my co-authors for their contributions to the papers we worked on together. I found these experiences both enjoyable and enlightening.

I would like also to give a very special thanks to my family for the support they provided me, both through my studies and also throughout my entire life. Without their love, encouragement and understanding i would not have finished this thesis.

Finally, in conclusion. I wish to thank all of my colleagues from The Department of Computer Science at The University of Liverpool. I have made some good friends along the way and shared some fun experiences with all of you.

Contents

List of Figures

List of Algorithms

List of Tables

Abbreviations

iff	**I**f and only **IF**
$\boldsymbol{w.h.p.}$	**W**ith **H**igh **P**robability
ME	**M**obile **E**ntity
$\mathcal{MR}$	**M**obile **R**obot
$\mathcal{FMR}$	**F**ast **M**obile **R**obot
$\mathcal{SMR}$	**S**low **M**obile **R**obot
$s.t$	**S**uch **T**hat
CCP	**C**oupon **C**ollectors **P**roblem

Dedicated to my parents, Tom and Liz, and my sister Joanne.

We all have dreams but in order to make those dreams into a reality it takes the support from a loving family.

Chapter 1

Preface

1.1 Algorithms

This Chapter has been included for readers who are perhaps unfamiliar with the topic of this work. In this section the concept of algorithms will be introduced along with explanations of what it means to analyse them as well as some technical explanation.

1.1.1 Overview

Algorithms are procedures used to solve some task. More specifically, they are well defined steps that take an input either as a single value or a set of values and then outputs either a single value or a set of values. In this way algorithms can be seen as a sequence that can be followed to "solve" a computation problem.

1.1.2 Analysis

When we talk about analysing an algorithm quite often we are measuring its efficiency in some way. The metric of efficiency for an algorithm is usually based on its speed. However, one could just as easily measure its memory or energy usage as a metric for analytical purposes.

We live in a world where there are still many problems we do not even know exist, let alone are aware of efficient solutions to them. However, for thoes problems that we are aware of but have yet to solve them efficiently we place into a subset of problems called NP-complete. It is interesting to note that there is a special property that holds for these types of problems that means if an efficient solution is found for one then that means there must also be efficient solutions for the other problems in this subset. Unfortunately, no efficient solution has yet been found for any of these problems. However, in the meantime we can use what we call approximation algorithms to get close to an efficient solution in these cases.

Definition 1.1. *running time*: The time it takes an algorithm to complete its steps and finish a task.

Definition 1.2. *steps*: it is assumed that each line of code in an algorithm is a step and that a step will take a constant time to run. That way when comparing algorithms between different computers we can still get a reliable measurement on its running time.

For this work we will be interested in the speed of an algorithm, that is the running time that an algorithm needs to complete the given task. Given that depending on what computer an algorithm is run on it may run faster or slower in comparison with the same algorithm on another computer, due to possibly different hardware configurations, we measure this running time in the number of steps executed. Furthermore, in order to understand better the performance of an algorithm for any size input we tend to express the running time of an algorithm as a function of n, $f(n)$, where n is the input size.

Following on from this it is also important to understand that for large values of n the lower order terms of the running time function are rendered inconsequential. Therefore, when looking at the running time of an algorithm it is usual to only consider non-constant factors. At this point we are considering the order of growth of the algorithm.

Definition 1.3. *order of growth*: The rate at which the number of steps an algorithm must perform to reach a solution that is given by the dominant factor in the growth function.

When we are talking about these orders of growth we are looking at the function as a way of describing the limit for the runtime of the algorithm. In Computer Science the method that we use to describe the limiting behaviour of functions is through asymptotic notation. Table 1.1 shows common examples and definitions of this form of notation.

Table 1.1: Examples of asymptotic notations and their meanings.

Notation	*Description*
$f(N) = \mathcal{O}(g(N))$	f is bounded from above by g asymptotically.
$f(N) = o(g(N))$	f is dominated by g asymptotically.
$f(N) \sim g(N)$	f is equal to g asymptotically.
$f(N) \in \Omega(g(N))$	f is bounded from below by g asymptotically.
$f(N) \in \theta(g(N))$	f is bounded from both above and below by g asymptotically.
$f(N) \in \omega(g(N))$	f dominates g asymptotically.

1.2 Distributed Computing

This thesis has its roots firmly embedded in Distributed Computing, that is computing that takes place in a Distributed Setting.

Definition 1.4. *Distributed Setting*: A setting that has no central or controlling aspect. In computing this can be seen as groups of networked computers that have processors running concurrently in parallel, each with its own memory.

Distributed Computing tackles problems by utilising the collective power of the computers or, in the case of this work, robots that inhabit the system. The algorithms in this work therefore are part of a category of algorithms called Distributed Algorithms. Quite often Distributed Algorithms will tackle separate parts of the problem with information spread across the system.

Definition 1.5. *Distributed Algorithms*: Algorithms that have been designed to execute concurrently on independent processors.

One of the biggest factors for Distributed Algorithms is the coordination of the processors involved in solving the task. However, there are huge benefits to using Distributed Algorithms in the correct setting. For example, a good Distributed Algorithm will allow for better levels of fault tolerance than a traditional algorithm as the other processors in the system should seamlessly pick up where the failed one left of. Furthermore, often it is the case that by splitting a task into sub-parts that can be carried out by each processor, or robot, in the system the task can be completed much faster.

1.3 Specific Chapter Definitions

This section of the chapter is designed to layout some specific definitions of notions that will be used later on in this work.

1.3.1 Location Discovery

The following definitions are in reference to Chapter 3.

Definition 1.6. *Arbitrary but distinct positions*: What is meant here is that each robot starts at a position that is randomly determined and is independent from the other robots starting locations.

Definition 1.7. *Unit Circle*: A unit circle usually means a circle with a radius of 1. However, for the purpose of Chapter 3 we talk of a unit circle having the circumference of 1. This is done without loss of precision as it simply allows us to normalise things with respect to 1, thereby making explanations and understandings clearer.

Definition 1.8. *Anonymous Robots*: Robots are defined as anonymous as they are unknown to one another and given their starting configuration there are no fundamental differences between them and so they could be interchanged without impact.

Definition 1.9. *Unit Speed*: Here unit speed simply means with a speed $= 1$. This gives any robots with a unit speed the ability to traverse a unit circle, as defined before, in one time step. As before, we use this definition to make explanations and understandings clearer.

Definition 1.10. *Synchronised Rounds*: The notion of synchronised rounds simply referes to the fact that all rounds are performed by the robots at the same time and at the same pace.

Definition 1.11. *Leaving marks*: When we refer to robots leaving marks it is to be understood that for a robot to leave a mark this could mean that they can leave some sort of signal or note for either other robots, or itself, to discover and use at a later date.

Definition 1.12. *Exchanging messages*: Robots exchanging messages simply refers to communication between robots. However, in Chapter 3 the robots are limited to no communication with one another outside of collisions that occur during the walking phase of a round.

Definition 1.13. *Coupon Collector's Problem (CCP)*: One player must collect m coupons. During each consecutive attempt the player draws each coupon with probability $\frac{1}{m}$. One can use a short calculation and a union bound to prove that after $\alpha \cdot m \log m$ attempts the player is left without a full set of coupons with probability at most $\frac{1}{m^{\alpha-1}}$, for any constant $\alpha > 1$ [98]. CCP can be also executed in consecutive stages, where each stage can be formed of a fixed number ℓ of consecutive attempts. In this case one can conclude that it is enough to run $\alpha \cdot \frac{m}{\ell} \log m$ stages to collect all coupons with high probability $1 - 1/m^{\alpha-1}$.

1.3.2 Evacuation Problem

The following definitions are in reference to Chapter 4 and Chapter 5.

Definition 1.14. *Group Search*: In this context Group Search is the type of problem we have carried out research on. It involves a group of robots searching for one or more locations or another robot or other robots in a given environment. In this work we look at the *Evacuation Problem.* This is a form of group search problem where one or more robots search for a location from which to evacuate the environment. The task is complete when all of the robots in the system have reached that location.

Definition 1.15. *Line*: In Chapter 4 we consider the environment of a line. Here the robots all start at some point on the line and it is assumed that this line is a one dimensional line that extends infinitively in both directions from the robots starting location.

Definition 1.16. *Disk*: In Chapter 5 we consider the environment of a disk. This disk is a unit disk where the radius of the disk is 1. The robots start off in the centre of the disk and are tasked with locating a single point that is on the perimeter of that disk.

Definition 1.17. *Mobile Robots*: For the purpose of this work all robots mentioned have the ability to move and there should be no distinction between the use of "robots" and "mobile robots".

Definition 1.18. *Maximal Speed*: The maximal speed of a robot is the fastest speed that the robot is able to travel by.

Definition 1.19. *Unit Speed*: Throughout this work we make explanation and understanding clearer by designating the maximal speeds of the robots to be that of a unit speed. A unit speed in this context simply means a speed $= 1$.

Definition 1.20. *Non-Wireless/Local Communication*: When we talk about non-wireless or local communication what is meant is that the robots are only able to communicate with one another when they occupy the same location. We assume that communication happens instantly and that there is no chance of missed or corrupt communication occurring.

Definition 1.21. *Wireless Communication*: When we talk about wireless communication it is assumed that all robots in the system are able to communicate with one another at any time no matter where they all are at that time. We assume that communication happens instantly and that there is no chance of missed or corrupt communication occurring.

Chapter 2

Introduction

2.1 Motivation and Problem Scope

This thesis looks at the area of *Distributed Algorithms* in the field of *Computer Science*, more specifically it investigates the area of *Control Problems for Mobile robots in Distributed Settings*. Work carried out on applied areas of *Graph Theory* and *Network Analysis* can also be found within this thesis.

Firstly, however, we will talk about the world of *Mobile robots*. The trends in processing power described by *Moore's law* and the trends in network traffic are increasing at different rates and so the power of our processors cannot keep up with the demand placed on our networks today. With that said, there is therefore a need for *Distributed Computing*, and that will increase further with large parts of the developing world set to be fully connected to the Internet and thus the *Cloud* in the near future. This would suggest that the future of computing will be heavily dependent on solutions that are, at the very least in part, distributed.

Already the move towards a distributed world has begun with huge leaps forward in the past few years with regards to making use of distributed computing solutions in our daily lives. There has been work done by [95] with the aim of creating a self-organising group of robots to build structures. They work in a scenario where the robots must locate the building blocks needed and then move the blocks into position to create a useful structure. The approach used is an increasingly popular one of looking to nature for the solution with a biological-inspired swarm intelligence based algorithm proposed.

There has been much talk about using teams of robotic swarms to explore planets that are incapable of supporting life and would provide a much cheaper option than sending a team of astronauts to these planets themselves. [108] propose an autonomous robotic swarm exploration to search for extra-terrestrial life on Mars. This would also have applications in a military sense where it may be too dangerous to send humans in to do reconnaissance, intrusion detection or mine clearing [110].

Leading on from the dangerous settings of the area of war, there has been significant strides in multi-robot teams for Search and Rescue situations in natural disaster zones [24, 91]. [24], for example, introduce a multi-robot algorithm for the use in search and rescue scenarios for exploration of unknown terrain. Their solution allows for parallel search and rescue operations to be run alongside each other by exploiting the robustness of distributed teams of robots.

Looking more towards the industrial front and how distributed computing can be used to enhance our economic needs [48, 76, 88, 101]. Perhaps the most famous example of this would be the success story of Kiva Systems, [114], who are now owned by Amazon and have developed and continue research on teams of mobile robots that manage the giant warehouses that house the stock sold on the online market site. The work done in [48, 76, 88] shows how swarms of small automated guided vehicles are employed to collect items from storage shelves in warehouses and take them to a picking station that helps to simultaneously improve productivity and speed. In this way operators are able to simply stand still and have the required move towards them. This method employs the use of inventory pods that are picked up and moved by hundreds of mobile robot platforms. In 2009 the largest number of Kiva robots in a single warehouse stood at 500, [93], for a supply company in the USA. Since Amazon integrated the company into their business that record has been smashed with Amazon itself having 15,000 Kiva robots spread across its 10 main warehouses in their network, [38].

Even in the sphere of ecological needs there have been attempts in recent years to use distributed systems to help and aid wherever possible. There is an initiative at Heriot-Watt University, Edinburgh, Scotland that uses a distributed array of coral maintenance droids called "corralbots" to help maintain and repair the coral reefs in the oceans that have become damaged or endangered through overfishing in those areas [20].

As it is plainly evident from the applications mentioned above this area of computing, although studied now for several years, can only continue to grow in number of applications and importance in humanities dependency upon such systems. This is why it is imperative that we grow our understanding of the mechanisms and strategies that control and govern the movements of such *Mobile Entities*, *MEs*, so that we are able to keep up with the demand for ever more intelligent and dynamic solutions based upon a distributed approach to problem solving in todays modern world.

2.2 Background

2.2.1 Search and Discovery Problems

The *Search Problem* is well-studied within the fields of operations research, computing, and mathematics. This problem deals with a searcher looking for a hidden object (or "target"), wishing to minimize a resource used in finding it. Many versions of this problem can be considered, including variations in the environment, whether the target is fixed or mobile, and, the use of a deterministic or randomized search strategy. Furthermore, there can be differences in the approach considered with respect to the resource being minimised. There has been much work done with respect to minimising the time to find the target [7–9, 13, 14, 27, 32–34, 43, 46, 47, 50, 52, 78] as well as minimising the memory used to find the target [56, 73]. In the context of search and discovery of different varieties of environments there is a large volume of robot network exploration algorithms, they mainly focus on network topology discovery either in graph-based networks [27, 32, 41, 67] or in geometric setting [43, 52, 78, 115].

Varying the number of searchers is also another variable that has been studied in the past when looking at such problems. Many papers have investigated search and discovery algorithms from the perspective of a single explorer, [5, 11, 28, 53, 56, 66, 70, 78]. There is however an obvious motivation to use multiple searchers in the time needed to complete the search as it usually allows a greater search area to be covered in a shorter time. Although, in large unknown environments where communication between searchers is limited either to a short range or perhaps to the local vicinity it is often useful to consider the case to employ a distributed approach as in [27, 32, 34, 46, 67, 68, 73, 105, 115]. There are many advantages that endorse the use of distributed mobile systems. Working together the *MEs* can not only increase their efficiency but also their reliability through redundancy. Furthermore, the cost of such *MEs* is reduced through being able to use less advanced *MEs* to complete the same tasks, either through things like reduced memory or energy used.

However, having *MEs* that are less advanced sometimes presents its own problems. For example, sometimes it is necessary for the *MEs* to perform a distributed task without all performing exactly the same set of commands. If there exists no way to communicate or identify the *MEs* from one another what can be done to break the symmetry? In this case we move away from deterministic approaches of addressing these problems and look towards randomisation for the solution as done in [54, 68].

Search and Discovery problems can also inherently be seen as types of control problems and as with other control problems many have looked towards the natural world to help come up with innovative algorithmic solutions, some based on Brownian Walks and Levy Flights [107, 119] and others looking towards the insect world with ant and bee colony mechanisms [49, 58, 117].

As seen from above there can be many variations on the Search and Discovery problem and the book by Alpern and Gal [8] is a good survey of known results for these.

2.2.2 Rendezvous and Gathering Problems

Search Problems also naturally lead into the *Rendezvous Problem*, where two or more searchers seek to meet in an environment, and this problem naturally lends itself to additional considerations of the inherent abilities of the searchers themselves, such as whether they have the same speed or different speeds, their ability to communicate and see (typically over a limited distance), and if the searchers are able to follow the same or different search strategy (e.g. do the searchers have unique identifiers so they can adopt their own search method, or are they indistinguishable and therefore must use the same (randomized or deterministic) strategy?).

Again, as with the Search and Discovery problem it is possible to approach the solution from either a centralised or distributed perspective. Approaching from the distributed side the main problem that is obvious is agreeing upon a place to meet. This of course can be made more difficult by limitations on communication. As mentioned earlier Rendezvous and Gathering Problems tend to lead on from Search Problems meaning that many of the variations studied above also apply here with a lot of research being done in a variety of settings and with a variety of constraints [37, 84, 100, 113]

2.2.3 Monitoring and Patrolling Problems

The *Monitoring Problem* is where, in a graph or geometric environment (such as a simple polygon), *MEs* are arranged in stationary positions to constantly survey the graph or region and usually the *MEs* have a limited field of vision. One of the main targets of such problems is to maxamise the visual range with the minimum number of *MEs* as there is probably a cost for each additional *ME* introduced into the system. This formulation of the problem is known widely as the *Art Gallery Problem* [36].

The problem has gained much popularity in recent years, [21, 59, 71, 121] with the emergence of more advanced robotic systems meaning that truly distributed structures can be deployed easily. The idea of *MEs* that are able to self-organise into a suitable configuration for efficient monitoring has been looked at by [121] in relation to vehicles that can communicate with each other to position themselves effectively on road systems to minimise congestion with reduced impact on travel time. Also [59] has looked into neighbour discovery in a sensor network with directional antennae.

In some circumstances, there may not be enough *MEs* to constantly monitor the environment. In this scenario it is vital that the *MEs* are able to detect this vulnerability in the system and adapt accordingly. This means that the *MEs* would need to adopt a patrolling strategy rather than a static monitoring one. This gives us the *Network Paroling Problem*, the main focus of which is to minimise the time between visits to points of the network or areas of the geometric space by the *MEs* as studied by [3, 55, 61, 103, 106, 111]. This becomes more interesting still when taking into consideration that some areas or points may be more important or vulnerable than others and so therefore the *MEs* may choose a strategy that divides their time up unevenly between each location to ensure maximum frequency to certain sections of the patrolled area.

It is worth noting here that in the setting of a general graph with edges of equal length then the *Network Patrolling Problem* is NP-Hard as if there exists only a single *ME* then the problem becomes one of finding a Hamiltonian cycle in the graph, [29].

As with the previous problems looked at in this chapter the *Network Patrolling Problem* is also subject to a wide variety of settings and constraints that can be imposed to make the problem more realistic to the real world or more interesting to study. For example, [3] uses a model where communication between the *MEs* is limited. This adds an extra dimension of complexity to the problem and can be used to accurately model a real world situation where surveillance is being performed where radio silence is necessary.

Furthermore, just like with the *Search Problem* it may be beneficial to break up the symmetry of a deterministic approach by adopting a randomised algorithm instead when patrolling [4, 69, 106]. Although, this time the breaking of symmetry may be simply to enable more efficient paroling in terms of attempting to fool any potential intruders. A Bayesian learning method was used by [106] to do just this.

The *Network Paroling Problem* has many real world applications. One of which is an intuitive jump to make from theoretical surveillance to that of Unmanned Aircraft Surveillance where work has already been done with that exact scenario in mind [2].

Again nature can also help provide useful solutions to problems with work being done using strategies taken from ant colonies to enable paroling of areas or networks based on the pheromones left behind to help ensure portions of the patrolled locations do not go untended for too long [69].

2.3 Summary of Results

2.3.1 Robot Location Discovery

The results we obtained in this area have been published in [68] and are presented in full in Chapter 3. What is presented is a randomised distributed communicationless coordination mechanism for n uniform anonymous robots located on a circle with unit circumference. It is assumed the robots are located at arbitrary positions on the ring, unknown to other robots. The robots perform actions in synchronised rounds. At the start of each round every robot chooses the direction of its movement (*clockwise* or *anticlockwise*), and moves at unit speed during that round. Robots are not allowed to pass by one another, i.e., when a robot collides with another it instantly starts moving with the same speed in the opposite direction. Robots are also unable to leave marks on the ring, have zero vision and cannot exchange messages. However, on the conclusion of each round each robot obtains (some, not necessarily all) information regarding its trajectory during this round and no other. This information can be processed and stored by the robot for further analysis.

The *Location Discovery Task* to be performed by each robot is to determine the initial position of every other robot in the system at the start of the scenario and eventually to return and stop at its own initial position, or proceed to another task such as *Boundary Patrolling*, in a fully synchronised manner. The primary motivation was to study distributed systems where robots collect the minimum amount of information that is necessary to accomplish this location discovery task.

Our original result for this problem was a fully distributed randomised (Las Vegas type, [16]) algorithm, solving the *Location Discovery Task w.h.p.* in $O(n \log^2 n)$ rounds (assuming the robots collect sufficient information). Note that this result also holds if initially the robots do not know the value of n and they have no coherent sense of direction. We believe that our work in [68] is the first attempt to solve the distributed boundary patrolling problem in the geometric ring (circle) model. Furthermore, the proof technique of the concept of virtual "batons" that robots exchange with each other upon collision, we believe, is a novel and intriguing approach to analysing the motion of the robots in the system. To our knowledge this is the first time such an approach has been used to analyse such a system and it led to us discovering a rotation of robots positions at the end of each round. This in turn had a large impact on us designing and analysisng the resulting algorithm. This method has since been explored and built upon by [45] and [44].

However, Chapter 3 presents another fully distributed randomised (Las Vegas type, [16]) algorithm that can achieve success *w.h.p* significantly faster in $n + O(\log^2 n)$ rounds. Given the constraints of the model any algorithm will need to visit all n locations anyway and so there is no escaping this cost. Following on from this, the limitations of the model lead us to believe that any approach will need some small amount of costly decisions by the robots. It is also our belief that this new algorithm is in fact the optimal solution for this problem as our approach works by each robot remembering not only where they have been but the decisions it made to get there. This allows the robot to remember any beneficial decisions and reapply them throughout the process while at the same time avoiding repeating any costs. However, we have yet to formalise a proof for this claim.

2.3.2 Evacuation Problem on the Line

The results we obtained for the *Evacuation Problem on the Line* have been published in [34] and are presented in full in Chapter 4.

We consider the *Group Search Problem on the Line*, or *Evacuation Problem on the Line*, in which k robots located on the line perform search for a specific destination. The robots are initially placed at the same point (origin) on the line L and the target is located at unknown distance d either to the left or to the right from the origin. All robots must simultaneously occupy the destination, and the goal is to minimize the time necessary for this to happen. The problem with $k = 1$ is known as the *Cow Path Problem*, and the complexity of this problem is known to be $9d$ in the worst case (when the cow moves at unit speed), where d is the distance between the origin and the destination. It is also known that this is the case for $k \geq 1$ unit-speed robots. Our results show for the first time a clear argument for this claim by showing a rather counter-intuitive result. Namely, in any metric, independently from the number of robots, group search cannot be performed faster than in time $9d$. We also examine the case of $k = 2$ robots with different speeds, showing a surprising result that the bound of $9d$ can be achieved when one robot has unit speed, and the other robot can move with speed at least $\frac{1}{3}$. Finally the case where $k = 3$ robots, with one having a speed less than 1, is briefly looked at and we show that a bound of $9d$ can yet again be achieved, but only if the slower robot's speed is at least $\frac{1}{5}$. Our analysis of this problem is made clear through our use of Minkowski Spacetime. Introduced by Hermann Minkowski, [96], the Spacetime cone that is centeral to this theory is a conveniently formulated mathematical explanation of Einstein's theory of special relativity, [60]. Our use of this concept in the analysis of the *Evacuation Problem* is to our knowlage the first time such an approach has been used here and in the related *Cow Path Problem.*

2.3.3 Evacuation Problem on the Disk

Our work on the *Evacuation Problem on the Disk* has been published in [46] and is presented in full in Chapter 5 of this thesis.

In this work k mobile robots inside a circular disk of unit radius are considered. The robots are required to evacuate the disk through an unknown exit point situated on its boundary. It is assumed all robots have the same (unit) maximal speed and start at the centre of the disk. The robots may communicate in order to inform each other about the presence (and its position) or the absence of an exit. The goal is for all the robots to evacuate through the exit in the minimum time possible.

Two models of communication between the robots were considered: In *non-wireless* (or *local*) *communication* model robots exchange information only when simultaneously located at the same point, and *wireless communication* in which robots can communicate between each other at any time.

The following question for different values of k is studied: What is the optimal evacuation time for k robots? We were able to construct algorithms to accomplish this and present lower bounds in both communication models for $k = 2$ and $k = 3$ thus indicating a difference in evacuation time between the two models. Almost-tight bounds are also obtained on the asymptotic relation between evacuation time and team size, for large k. Also in the local communication model it is shown that, a team of k robots can always evacuate in time $3 + \frac{2\pi}{k}$, whereas at least $3 + \frac{2\pi}{k} - O(k^{-2})$ time is sometimes required. In the wireless communication model, time $3 + \frac{\pi}{k} + O(k^{-4/3})$ always suffices to complete evacuation, and at least $3 + \frac{\pi}{k}$ is sometimes required. This shows a clear separation between the local and the wireless communication models.

We found that one of the remarkable points of interest for this problem was that when increasing the number of participating robots only slightly, and still when considering a relativity small number of k, the compexity of the problem itself grew rapidly.

2.4 Thesis Structure

The chapters in this thesis contain both work related to the main topic of the doctorate as well as several side interests that the author has pursued throughout its duration. The material covered in this thesis has been aranged in the following way:

Chapter 3
This chapter covers Search and Discovery problems, going into the background of the topic as well as presenting results obtained in [68], as well as progress made that expands upon this work.

Chapter 4
The material in this chapter focuses more on the collaboration of MEs to achieve goals while still in the context of Search and Discovery problems as well as introducing The Evacuation Problem and the results relating to this produced in [34].

Chapter 5
Expanding on the previous chapter, here details of the material presented in [46] as an obvious path forward following the promising work accomplished in [34] will be discussed.

Chapter 6
The final chapter shows the conclusions of this work and looks further avenues forward for this research.

2.5 Author's Contribution

Chapter 3 is based on the full version of [68], joint work completed by the author with co-authors, Tom Friedetzky, Leszek Gąsieniec, Russell Martin and Ely Porat. Chapter 4 is work based upon [34] done by the author and co-authors Marek Chrobak, Leszek Gąsieniec and Russell Martin. The research in Chapter 5 comes from [46], also done by the author with the co-authors Jurek Czyzowicz, Leszek Gąsieniec, Evangelos Kranakis, Russell Martin and Dominik Pająk.

Everything else is the author's work, written for this PhD project and supervised by Russell Martin and Leszek Gąsieniec.

TABLE 2.1: The author's publications and co-authors throughout the duration of the author's PhD studies.

Title	*Authors*	*Appeared*
Observe and Remain Silent [68]	T. Friedetzky, L. Gąsieniec, T. Gorry and R. Martin	[1]MFCS 2012
Evacuating Robots from an Unknown Exit Located on the Perimeter of a Disc [46]	J. Czyzowicz, L. Gąsieniec, T. Gorry, E. Kranakis, R. Martin and D. Pająk	[2]DISC 2014
Group Search on the Line [34]	M. Chrobak, L. Gąsieniec, T. Gorry and R. Martin	[3]SOFSEM 2015

[4]MFCS 2012: The 37th International Symposium on Mathematical Foundations of Computer Science, 2012

[5]DISC 2014: The 28th International Symposium on Distributed Computing, 2014.

[6]SOFSEM 2015: 41st International Conference on Current Trends in Theory and Practice of Computer Science, 2015

Chapter 3

Location Discovery

3.1 Introduction

This chapter is based heavily on results published in [68] at The 37th International Symposium on Mathematical Foundations of Computer Science in 2012 (MFCS'12). There is also work presented in this chapter concerning improvements to the *location discovery problem* presented in [68] that have been discussed between the author and their supervisors but at the time of writing have still yet to be published. Furthermore, it should be noted here that the initial idea for this problem and the beginnings of the initial solution are already part of the author's Master's Dissertation, [75], but have been included here as they provide the foundations for what eventually became the paper we published that was mentioned at the start of this paragraph, [68].

In this chapter we study a randomised distributed communication-less coordination mechanism for n uniform anonymous robots located on a circle with unit circumference. We assume the robots are located at arbitrary but distinct positions, unknown to other robots. The robots perform actions in synchronised rounds. At the start of each round a robot chooses the direction of its movement (*clockwise* or *anti−clockwise*) and moves at unit speed during this round. Robots are not allowed to overpass, *i.e* when a robot collides with another it instantly starts moving with the same speed in the opposite direction. Robots cannot leave marks on the ring, have zero vision and cannot exchange messages. However, on the conclusion of each round each robot has access to, some (not necessarily all), information regarding its trajectory during this round. This information can be processed and stored by the robot for further analysis.

The *location discovery task* to be performed by each robot is to determine the initial position of every other robot and eventually to stop at its initial position, or proceed to another task, in a fully synchronised manner. Our primary motivation is to study distributed systems where robots collect the minimum amount of information that is necessary to accomplish this location discovery task.

Our original result for this problem, [68], was a fully distributed randomised (Las Vegas type, [16]) algorithm solving the *location discovery problem* *w.h.p.* in $O(n \log^2 n)$ rounds (assuming the robots collect sufficient information). Note that our result also holds if initially the robots do not know the value of n and they have no coherent sense of direction. However, this chapter also presents another fully distributed randomised, Las Vegas type, algorithm that can achieve success *w.h.p.* significantly faster in $n + O(\log^2 n)$ rounds.

3.1.1 Overview

A cycle-based topology of communication networks is very often identified with the *ring* of discrete nodes in which each node has two neighbours at its opposite sides. The ring network is one of the most studied network topologies in the context of standard distributed computation tasks [92, 109, 120] as well as coordination mechanisms for mobile robots [87]. In this chapter, however, the focus is on geometric rings, later referred to as *circles*. The work presented in this chapter refers to the recently popularised concept of *swarms*, i.e., large groups of fairly primitive but cost-effective entities (robots) that can be deployed to perform an exploration or a monitoring task in a hard-to-access hostile environment for humans, for example on a planet other than Earth where there may be little or no oxygen for humans to breath. There has been substantial progress in the design of efficient distributed coordination mechanisms in a variety of models for mobile robots, e.g., see [15, 37, 84, 112]. In this chapter a version of the model introduced in [15] is considered. In that model, the robots operate in synchronised rounds, they are assumed to be anonymous, and they lack means of communication. A robot wakes up at the beginning of each round and performs its move that depends on the current location of other robots in the network. In the model from [15] the moves are assumed to be instantaneous, but this last assumption is not true in this instance. Numerous algorithms have been developed in the literature for a variety of control problems for robot swarms, see [15, 37, 63, 64, 84, 112]. Most of these algorithmic solutions, with certain exceptions, e.g., [39], impose on the participating robots access to the *global* picture, in other words the ability to monitor performance of all robots. While there is a large volume of robot network exploration algorithms, they mainly focus on network topology discovery either in graph-based networks [27, 32, 41, 67] or in geometric setting [43, 52, 78, 115]. As mentioned earlier here the focus is on the network model similar to [15] in which communication is limited to a bare minimum. In such networks, the communication deficiency of a robot is compensated by an astute observation and analysis of its own movement. The *trajectory* of a robot's movement in a given round is represented as a continuous, rectifiable curve, that connects the start and the end points of the route adopted by the robot. While moving along their trajectories, robots collide with their immediate neighbours, and information on the exact location of those collisions might be recorded and further processed. When robots are located on a circle, thanks to its closed topology, each robot may eventually conclude on the relative location of all robots' initial positions, even given only limited information about its trajectory. This procedure, in turn, enables other distributed mechanisms based on full synchronisation including equidistant distribution along the circumference and optimal boundary patrolling scheme. Most of the models adopted in the literature on swarms assume that the robots are either almost or entirely oblivious, i.e., throughout the computation process the robots follow a very simple, rarely amendable, routine of actions. Oblivious algorithms have many advantages including striking simplicity and self-stabilisation [57] properties.

3.1.2 The Model

In this chapter geometric network model, i.e., a circle with circumference one is adopted, along which a number of robots move and interact in fully synchronised rounds (each of which lasts one unit of time). The robots are uniform and anonymous to one another. Moreover, the robots do not necessarily share the same sense of direction, i.e., while each robot distinguishes between its own clockwise (C) and anticlockwise (A) directions, robots may not have a coherent view on this (see Section 3.3.4 for more on this). At the beginning of each round a robot chooses a direction of its move from $\{A, C\}$ and moves at unit speed. It is assumed that robots are not allowed to pass over each other along the circle. In particular, when a robot collides with another (robot) it instantly starts moving with the same speed but in the opposite direction. The robots cannot leave marks on the ring, they have zero visibility and cannot exchange messages. Instead, on the conclusion of each round every robot learns a specific information concerning its recent trajectory. In particular, for odd n we assume that a robot is informed about the relative distance between its location at the start and the end of this round. For even n, however, the robots must also learn about the exact time (location) of their first collisions during this round. This information can be processed or stored for further analysis. The aim of a robot is to discover the initial positions of all other robots. Our main motivation for [68] was to study distributed systems where robots collect the *minimum amount* of information necessary to accomplish the location discovery task, and the novelty comes from considering the situation where robots operate with a very limited amount of information collected during the discovery procedure. One might consider, for example, that a robot spends energy to determine its current location, and wants to minimize its energy expenditure.

Since the robots never pass over one another it can be assumed that the robots are arranged in an implicit (i.e., never disclosed to the robots) periodic order $s.t$ robots are located at random intervals along the circumference of the ring from a_0 to a_{n-1}. The original positions of a_i, are denoted by p_i for all $i \in [n]$[1]. Note that, due to the periodic order of robots, all calculations on implicit labels of robots that follow are performed modulo n.

Note however, that the circumference of the circle has to be known in advance. Otherwise, a participating robot might not be able to tell the difference between $n = 1$ and $n > 1$. In particular, if the robot imposed a limit on the traversal time until the first collision, the adversary would always choose the circumference to be large enough to accommodate distant locations between the robots preventing them from ever getting close enough. On the other hand, if the robot continues its search indefinitely, the adversary could choose $n = 1$ and the location discovery process would never end. Thus, it is important to know either n or the circumference of the circle.

[1] $[n] = \{0, 1, \ldots, n-1\}$ for any natural number n.

3.1.3 Results

We assume that n mobile robots are initially located on a circle at arbitrary, distinct and undisclosed positions. As stated previously, the task of each robot is to determine the initial position of every other robot. On the conclusion of the algorithm robots either synchronously stop at their initial positions or may proceed with another task. For the clarity of presentation, we first provide a solution to the distributed location discovery under the assumption that the robots have a coherent sense of direction, i.e. they all have the same understanding of what direction clockwise and anti-clockwise is, and the value n is known in advance to all robots. Later, in Sections 3.3.3 and 3.3.4 we provide further evidence on how these two assumptions can be dropped. Finally, we briefly describe how the location-discovery mechanism can be used to coordinate actions of robots in distributed boundary patrolling on circles, see Section 3.3.5. This last part should be seen as a natural continuation of [42] devoted to efficient centralised patrolling mechanisms designed for robots with distinct maximum speeds. We believe that our work in [68] is the first attempt to solve the distributed boundary patrolling problem in the geometric ring (circle) model. Our work in this chapter shows that we can accomplish this task in $O(n \log^2 n)$ rounds *w.h.p.*. However, we can introduce the concept of a *Stationary* choice of movement where that robot initially chooses not to move but instead remains at its starting location until a collision with another robot. Using this we also show in this chapter that with the added inclusion of a *Stationary* choice of movement at the start of a round robots are able to accomplish the task *w.h.p.* significantly faster in $n + O(\log^2 n)$ rounds..

All of the bounds in this chapter hold with high probability[2] (*w.h.p.*) for n large enough. However, one can easily modify the solutions such that by periodically repeating actions of robots, they can solve the task with the required level of confidence even for smaller, e.g., constant values of n.

3.2 Rotation mechanism

The location-discovery algorithm is formed of a number of *stages*. Each stage is a sequence of at most n consecutive rounds, each of unit duration. Recalling from

[2]With probability at least $1 - 1/n^c$ for some positive constant c.

earlier, a round lasts exactly one unit of time. Given that the ring is one of unit circumference this would be exactly enough time for the robot to walk the entire circumference and arrive back at its original starting location if it experienced no collisions along the way, i.e. if the robot was the only one in the system. At the beginning of the first round of each stage a robot a_i randomly chooses the direction (clockwise or anticlockwise) of its movement, and moves with unit speed throughout the entire stage. Later throughout the same stage, the exact location and the movement direction of a_i depends solely on the collisions with its neighbours a_{i-1} and a_{i+1}. We show that on conclusion of each round the robots always reside at the initial positions $p_0, \ldots, p_{n-1}$, where there is a $k \in [n]$, equal for all robots, such that the current location of robot a_i corresponds to p_{i+k}. It should be noted here that depending on the initial choices of all the robots in terms of their direction of movement then p_{i+k} could also be p_{i-k}. Also note that this observation allows robots to visit (and record) the initial positions of other robots. Thus, part of the limited amount of information that a robot obtains is its position, *relative to its initial starting location*, at the end of each round. A stage concludes at the end of a round when each robot a_i arrives at its original starting position p_i. We show that *w.h.p.* robots require $O(\log^2 n)$ stages to learn the locations of their counterparts. Since each stage is formed of at most n rounds, the total complexity of our algorithms is bounded by $O(n \log^2 n)$.

Throughout the discovery procedure, robots move with uniform speed one. Recall when two robots collide, they instantly bounce back without changing their uniform unit speed. While observing two indistinguishable colliding robots, one could wrongly conclude that the two robots overpass each other. We assume that at the beginning of each stage of our algorithm every robot a_i holds a unique *virtual baton* b_i. During the first collision with either a_{i-1} or a_{i+1} this baton gets exchanged for a baton currently held by the respective robot. In due course, further exchanges of batons take place. We emphasize that the concept of batons is solely a proof device in what follows, that they do not actually exist as far as the robots are concerned, and that no actual communication (or exchange of any object) takes place between the robots when they collide.

Lemma 3.1. *At the start of each round baton b_i resides at position p_i, for all $i \in [n]$.*

Proof. At the start of the location discovery procedure, b_i resides at p_i for all i. During the first round, baton b_i moves in a unidirectional manner with unit speed (being exchanged as appropriate during collisions), so b_i must arrive at p_i on the conclusion of this first round. Inductively, at the end of each round (i.e. start of the next round) of the procedure, b_i will reside at position p_i. ■

Using Lemma 3.1 it can be concluded that at the start of each round the robots populate initial locations $p_0, \ldots, p_{n-1}$. In fact, one can state a more accurate lemma.

Lemma 3.2. *There is a $k \in [n]$ s.t. at the start of each round, for all $i \in [n]$, robot a_i resides at position p_{i+k}.*

Proof. At the start of the location discovery procedure, all initial positions are populated by the robots, each carrying a (virtual) baton. From Lemma 3.1, b_i begins (and ends) each round at position p_i. Since some robot must always be carrying b_i, there is a robot occupying the location p_i at the beginning of a round, and some (possibly different) robot occupying p_i (and holding b_i) at the end of the round. The same argument holds for each i, hence all n initial locations are occupied at the end (start) of each round. Recall that the robots never overpass, i.e., robot a_i always has the same neighbours a_{i-1} and a_{i+1}. Thus, if a_i resides at position p_{i+k} for some k, then a_{i-1} and a_{i+1} must reside at the respective locations p_{i+k-1} and p_{i+k+1}. ■

Using the observation from Lemma 3.2, consider the respective locations p_{j+k_1} and p_{j+k_2} of robot a_i at the start of two consecutive rounds. One can conclude that during one round all robots rotated along the initial positions by a *rotation index* of $r = k_2 - k_1$, i.e. each robot experiences the same shift by r places (either clockwise or anticlockwise) between the beginning and the end of one round.

Lemma 3.3. *During one stage the rotation index r remains unchanged.*

Proof. The movement direction of each robot at any moment is determined by the movement direction of the virtual baton currently possessed by that robot since, during each round, a baton moves in a unidirectional manner around the ring. Since at the beginning of each round the virtual batons reside at their original positions and they do not change their directions during the entire course of the stage, the rotation index throughout each stage must remain unchanged. At the beginning of a stage, the (random) choices of the robots determine the directions of the batons during the entire stage, i.e. if robot a_i chooses "clockwise", then baton b_i will move clockwise during that entire stage. Since at the beginning of each round, the virtual batons reside in their original positions (Lemma 3.1), and they don't change their directions during the entire stage, this means the pattern of movement and collisions (swaps of batons) of the robot beginning a round at p_i will be identical that of a_i during the first round of the stage. Hence, the rotation index remains unchanged during an entire stage. ∎

Following on from this it can now be shown that the rotation index r depends on the initial choice of random directions adopted by the robots. Consider the first round of any stage. Let sets B_C and B_A contain the virtual batons that move during this round in the clockwise and anticlockwise directions, respectively, where $|B_C| = n_c$, $|B_A| = n_a$, and $n_c + n_a = n$. We say that during this stage virtual batons form a (n_c, n_a)-*configuration*.

Lemma 3.4. *In a stage with a (n_c, n_a)-configuration, the rotation index $r = n_c - n_a$.*

Proof. By Lemma 3.2 it is enough to prove the premise of the lemma for one robot. Without loss of generality, assume that baton b_i is in B_C. At the beginning of any round baton b_i is aligned with position p_i, and assume that at the beginning of the considered round b_i is carried by robot a_j.

First note that b_i can only be exchanged with batons from B_A since all batons in B_C move with the same speed in the clockwise direction. Moreover, during any round every baton from B_C is exchanged with every baton from B_A exactly twice at certain antipodal points of the ring. Why is this? Suppose $b_k \in B_A$, and let d denote the distance (along the circumference) between b_i and b_k, measured in the clockwise direction. Note that $d < 1$ since robots start at distinct locations. Then b_i and b_j meet (are exchanged by colliding robots) at time $d/2$. After additional time $1/2$ (since $d/2 + 1/2 < 1$), b_i and b_k meet again at the antipodal point of their first collision before returning to their respective positions at p_i and p_k.

Thus, during any round baton b_i is exchanged between colliding robots exactly $2n_a$ times. Also, since b_i moves in the clockwise direction during each exchange, an index of the new hosting robot is increased by one. Thus at the end of the considered round when b_i arrives at p_i it is hosted by robot a_{j+2n_a}. This leads to conclusion that the rotation index is $r = -2n_a$.

Focusing on batons from B_A, one can use an analogous argument to prove the rotation factor $r = 2n_c$. Now since $n_c + n_a = n$ it follows that $-n_a = n_c (\mathrm{mod}\ n)$ and finally $-2n_a = 2n_c (\mathrm{mod}\ n)$ admitting the uniform rotation index r across all robots.

Finally, $n_a + n_c$ (that has value 0 modulo n) is added to $-2n_a$ and the rotation index $r = n_c - n_a$ is obtained. ■

3.3 The Location Discovery Algorithm

Using the premise from Lemma 3.4, one can observe that if the rotation factor $n_c - n_a$ is relatively prime with n, denoted $\gcd(n_c - n_a, n) = 1$, a single stage with an (n_c, n_a)-configuration will last exactly n rounds. Moreover, during such a stage every robot will visit the original positions of all other robots. For example, if $n > 2$ is a prime number, one stage with $n_c, n_a \neq 0$ would be enough to discover the original positions of all robots. However, the situation complicates when n is a composite number. For example, when n is even, the difference $n_c - n_a$ is always even, meaning that n and $n_c - n_a$ cannot be relatively prime. This means that the mechanism described above will allow robots to discover at most half of the original positions.

In what follows this work first presents the discovery algorithm for odd values of n. Following on from this it is shown how this algorithm can be amended to perform discovery also for even values of n.

3.3.1 Algorithm for odd values of n

As mentioned earlier, the algorithm works in stages concluded by robots' arrival to their initial positions. It is further assumed that the robots know n and they have a coherent sense of direction.

The algorithm explores the basic properties of the network model, reflected in the functionality of the procedure Single-round, accompanied by a randomised control mechanism. The procedure Single-round describes the performance of a robot during a single round. As input the procedure accepts two parameters: current relative location, *loc*, and direction, *dir* $\in \{C, A\}$, i.e., the clockwise (C)

or the anti-clockwise (A) direction of movement. On the conclusion of the round the procedure returns two parameters: *new-dir*, i.e., the direction of the robot to move in the new round; and a real value *new-loc*, a relative distance (positive or negative) that describes the position relative to its starting point at the beginning of the round. (This allows the robot to compute its relative distance from its starting point at the beginning of the stage, or the entire discovery procedure as the model assumes that the robot has access to unlimited memory and processing power, as well as GPS information about itself.) Recalling the discussion at the beginning of this section, the set of *new-loc* data collected during the procedure is sufficient to accomplish the location discovery task if n is odd, if the robots are in an (n_c, n_a)-configuration with $\gcd(n_c - n_a, n) = 1$.

The main (randomised) control mechanism of the procedure DISCOVER is presented in Algorithm 1. Initially, the list of known points is empty. At the end of each round the content of the list is updated. Note that in step (3) the initial directions are chosen uniformly at random as this clearly is the only sensible choice.

Algorithm 1: The location-discovery procedure of a robot.

```
$the\text{-}list \leftarrow \emptyset$
repeat
    pick direction $dir$ from $\{C, A\}$ uniformly in random
    set $loc = 0$
    repeat
        $(new\text{-}dir, new\text{-}loc) \leftarrow$ SINGLE-ROUND$(loc, dir)$
        $the\text{-}list \leftarrow the\text{-}list \cup \{new\text{-}loc\}$
        $dir \leftarrow new\text{-}dir; loc \leftarrow loc + new\text{-}loc$
    until $loc = 0$
until $|the\text{-}list| = n$
return $the\text{-}list$
```

Here a stage is defined as *successful* when $\gcd(n_c - n_a, n) = 1$, i.e., when every robot visits all initial positions of other robots.

Lemma 3.5. *For any odd $n > 0$, a successful stage occurs within the first $O(\log^2 n)$ stages,* w.h.p.

Proof. The definition of a successful stage earlier clearly indicates that there is a desire to target a distribution of directions with $\gcd(n_c - n_a, n) = 1$. To simplify this task it is best to focus only on prime values of the rotation index where

$|n_c - n_a| < \sqrt{n}$. Note that the probability that during a stage the value $|n_c - n_a|$ is obtained is $2 \cdot \binom{n}{n_c}/2^n$. Using Stirling's factorial approximation one can prove that this probability is $\Omega(1/\sqrt{n})$, for all $|n_c - n_a| < \sqrt{n}$.

It is also known [94] that for a large enough integer m ($> 15,985$) the m-th prime number is not larger than $m(\log m + \log\log m)$. This can be also interpreted that for m large enough there are $\Omega(m/\log m)$ prime numbers smaller than m. In particular, it can be conculded that there are $\Omega(\sqrt{n}/\log n)$ primes between 0 and $\sqrt{n}$. Note, however that not all of these prime numbers need to be relatively prime to n. However, n can have at most $O(\log n)$ prime divisors. So there are $\Omega(\frac{\sqrt{n}}{\log n} - \log n)$ primes between 0 and $\sqrt{n}$ that are also relatively prime to n.

This leads to the conclusion that the probability that any one stage is successful is $\Omega((\frac{\sqrt{n}}{\log n} - \log n) \cdot \frac{1}{\sqrt{n}}) = \Omega(\frac{1}{\log n})$. In other words, there exists a constant $c_o > 0$ such that a stage is successful with probability at least $c_o/\log n$.

Finally, the performance of Discovery can be described by a Bernoulli process where the probability of success is $c_o/\log n$ in each stage. It is a well-known fact that after $c_o \log n$ stages of such a process, the probability of reaching a successful stage is constant, and after $c_o \log^2 n$ stages this probability is high. ■

Since each stage is composed of at most n rounds, the following can be concluded.

Theorem 3.6. *For any large enough odd n, the number of rounds required to perform full discovery of the robots' initial positions is $O(n \log^2 n)$* w.h.p.

Proof. Each stage is formed of at most n rounds. Since *w.h.p.* the algorithm accomplishes the discovery task in $O(\log^2 n)$ stages, the time complexity of Discovery is $O(n \log^2 n)$. ■

3.3.2 Amendment for even n

For the case when n is even it should be noted that for any $n_c + n_a = n$ we have that $n_c - n_a$ is also even. Thus, one cannot simply await a stage with $\gcd(n_c - n_a, n) = 1$. To deal with this problem we have came up with two solutions that can be applied, they are described below.

3.3.2.1 More information during a round

The first option available to us to be able to circumnavigate this issue is to target stages with $\gcd(n_c - n_a, n) = 2$, and in particular when $|n_c - n_a|$ is a double of a prime. Using a similar argument as in Lemma 3.5, one can prove that such a successful stage (where $\gcd(n_c - n_a, n) = 2$) occurs with probability $\Omega(1/\log n)$. In a successful stage, the robots form a bipartition $X_{\text{even}} \cup X_{\text{odd}}$, where $X_{\text{even}} = \{a_0, a_2, \ldots, a_{n-2}\}$ and $X_{\text{odd}} = \{a_1, a_3, \ldots, a_{n-1}\}$, and each robot learns the initial positions of all other robots in the same partition.

So can we solve the full location discovery problem in this case? Well, we can, provided a robot receives the same data about its new location at the end of each round as before, as well as the time until (or location of) its *first* collision in each round. We show that this very limited additional information suffices to allow the robots to solve the discovery problem. Note that this amendment is not changing the model as defined in the introduction of this chapter, but merely changing the *amount* (and type) of information a robot receives during execution of the procedure.

Recall that the calculations here are all done modulo n. With this in mind now consider a successful stage where $\gcd(n_c - n_a, n) = 2$. During this stage any robot $a_i \in X_{\text{even}}$ will visit all initial positions $p_0, p_2, \ldots, p_{n-2}$ and any $a_i \in X_{\text{odd}}$ will visit all initial positions $p_1, p_3, \ldots, p_{n-1}$. More formally this can be written as for any $i = 0, \ldots, n-1$ and $j = 0, \ldots, \frac{n}{2} - 1$, robot a_{i+2j} (respectively,a_{i+2j+1}) also visits the initial position p_i (resp. p_{i+1}) of a_i (resp. a_{i+1}). Note that if at the beginning of this stage robot a_i picks direction C and robot a_{i+1} (from the other partition) picks direction A, the two robots meet halfway between p_i and p_{i+1} after traversing distance *min-dist* $= |p_i - p_{i+1}|/2$. When this happens, the robots can retrieve the original positions of one another, i.e. robot a_i concludes that $p_{i+1} = p_i + 2 \cdot \textit{min-dist}$ and robot a_{i+1} concludes that $p_i = p_{i+1} - 2 \cdot \textit{min-dist}$.

Note also that when robots a_i and a_{i+1} pick the same direction C (or A) the distance to the first meeting with a_{i+1} that is observed by robot a_i is always longer than *min-dist*. Thus, to learn the correct distance to p_{i+1}, a single successful stage with initial directions C of a_i and A of a_{i+1} is sufficient. In other words, we need to run the procedure DISCOVERY long enough to ensure that such a stage occurs *w.h.p.* This means that a robot $a_i \in X_{\text{odd}}$ will maintain a record of its current *estimates* of the starting locations of neighbors in X_{even}, and similar for a robot

$a_i \in X_{\text{odd}}$. These estimates use the first collision information that a robot receives in each round, and the calculations described above (i.e. the first collision distance is used to estimate *min-dist* to the left or right neighbour). This record is updated, as appropriate, throughout the discovery procedure to build up a complete picture of the starting locations of the robots in the other partition.

Observe that if in the first round of a stage a_i (respectively, a_{i+1}) learns p_{i+1} (resp. p_i), then in the next $\frac{n}{2} - 1$ rounds of this stage every other robot a_{i+2j} (resp. a_{i+2j+1}) learns p_{i+1} (resp. p_i) since the directions of the batons b_i and b_{i+1} remain unchanged throughout the entire stage.

This leads us to conclusion that to solve the location-discovery problem for even n we need to run procedure DISCOVERY until, for each $i = 0, \ldots, n-1$, there is some successful stage in which robots a_i and a_{i+1} start moving during the first round in directions C and A, respectively. We show that $O(\log^2 n)$ stages of procedure DISCOVERY (modified so that a robot also collects the distance until its first collision in each round) still guarantee a solution to the discovery problem (*w.h.p.*) for even n.

Lemma 3.7. *For any even* $n > 0$, *each robot learns the positions of the others within the first* $O(\log^2 n)$ *stages* w.h.p. *when using the approach of having access to more information during a round.*

Proof. In order to simplify the proof we focus on two sets of pairs of initial positions: $P_0 = \{(p_{2j}, p_{2j+1}) : j \in [\frac{n}{2} - 1]\}$ and $P_1 = \{(p_{2j+1}, p_{2j+2)}) : j \in [\frac{n}{2} - 1]\}$. Within each set, each pair contains distinct robots' initial positions, and every such position belongs to some pair.

We split consecutive successful stages (with $\gcd(n_c - n_a, n) = 2$) of procedure DISCOVERY into two alternating sequences S_0 and S_1, where in stages from S_0 we consider pairs from P_0 and in stages from S_1 we consider pairs from P_1.

Without loss of generality, consider the sequence S_0. Recall that in these stages every robot visits every second initial position on the circle. Thus if, e.g., in the beginning of the first round of this stage robot a_{2j} moves in direction C and robot a_{2j+1} moves in direction A, after the first stage these two robots learn their relative positions, and in the remaining $\frac{n}{2} - 1$ rounds of this stage all other robots in X_{even} learn p_{2j+1} *and* all other robots in X_{odd} learn p_{2j}. Thus we need to consider enough number of stages in S_0 such that, for each $j = 0, \ldots, \frac{n}{2}$, there is a stage in which robot a_{2j} starts moving in direction C and robot a_{2j+1} starts moving in direction A.

We first assume that $|n_c - n_a| < \sqrt{n}$, which occurs *w.h.p.* Under this assumption, during each stage in S_0, we randomly populate the $\frac{n}{2}$ pairs in P_0 with pairs of directions, $(A, A), (A, C), (C, A)$ and (C, C). Since our primary interest is in the pair (C, A), we first estimate from below the expected number of (C, A) generated during each successful stage in S_0. We generate pairs sequentially at random assuming that initially the number of As and Cs is at least $\frac{n}{2} - \frac{\sqrt{n}}{2}$. We generate these pairs until either the remaining number of As or the remaining number of Cs is smaller than $\frac{n}{4} - \frac{\sqrt{n}}{2}$. This means that we generate at least $\frac{n/4}{2} = \frac{n}{8}$ pairs. One can now show that the probability of picking a mixed pair (C, A) is at least $1/5$.

Recall the *Coupon Collector's Problem* (CCP) from Definition 1.13 in which one player must collect m coupons. In this case one can conclude that it is enough to run $\alpha \cdot \frac{m}{\ell} \log m$ stages to collect all coupons with high probability $1 - 1/m^{\alpha-1}$.

We note here that random generation and further distribution of $(C, A)s$ in successful stages can be also seen as a version of coupon collection executed in stages. The $\frac{n}{2}$ pairs of positions in P_0 correspond to coupons in our version of CCP. During a single attempt in a successful stage (that occurs with probability $c_e/\log n$) a pair (C, A) is drawn with probability $1/5$ and allocated at random to one of the $n/2$ pairs in P_0. Thus, in a successful stage, in a single attempt each coupon (pair in P_0 with allocated (C, A)) is drawn with probability $2/(5n)$.

Compare now a single stage in standard CCP with a successful stage in our version of CCP. If in CCP a specific coupon is drawn more than once, the second and further attempts are void. In other words, these multiple attempts are wasted. In a valid stage of version of CCP (based on DISCOVERY), however, if an attempt results in a coupon (pair in P_0 with allocated (C, A)) that has been already collected in this stage, the attempt is continued until a not yet collected coupon is found. In other words, during a valid stage we may in fact generate more (but certainly not fewer) coupons compared to the respective stage in standard CCP.

Recall that during a successful stage at least $n/8$ sequential attempts are made, where each coupon out of $n/2$ is drawn with the probability $2/(5n)$. Since the probability defined on all coupons does not sum up to one, we may add a missing number of "null" coupons, each also drawn with probability $2/(5n)$. In turn, we obtain our version of CCP run in stages with $m = 5n/2$ coupons and stages of length $\ell = \frac{n}{8}$. Recall also that in our version of CCP $\alpha \cdot \frac{m}{\ell} \log m$ stages are required to collect all coupons *w.h.p.* $1 - \frac{1}{m^{\alpha-1}}$. Since the length of each stage is $\ell = \frac{n}{8}$ and $m = \frac{5n}{2}$, one can conclude that $\alpha \cdot \frac{5n/2}{n/8} \log(\frac{5n}{2}) = 20\alpha \cdot (\log(\frac{5}{2}) + \log n) < 25\alpha \log n$ stages of Discovery are needed to generate (C, A) for each pair in P_0, with probability $1 - 1/(\frac{5}{2} \cdot n)^{\alpha-1}$. This gives a high probability of success for $\alpha > 2$.

Similarly, one can analyse the generation of (C, A)s for all pairs in P_1. Thus *w.h.p.*, all robots can learn the position of all other robots with $O(\log^2 n)$ stages of Discovery. ■

3.3.2.2 More movement choices

The second option available to us is the introduction of a *Stationary* choice of movement at the start of each round. In this approach we say that the robots still are not aware of when their collisions occur but instead are able to now choose from three movements. Now in addition to the choices of clockwise (C) or the anti-clockwise (A) each robot is allowed to make the decision to remain stationary (S). What this means exactly is that at the start of that round any robot that chose (S) will not move in any direction. The only time a robot will move during a round after choosing this case will be when another robot that has made the choice to travel in a direction collides with the stationary robot. At this time the stationary robot would adopt the direction of the colliding robot and that robot becomes stationary. The decision is now made by each robot in the following way. They will still each decide randomly and independently their initial movement. S is chosen with a constant probability P and then with $\frac{1}{2}(1 - P)$ either C or A is chosen.

So, why does this help? Well if we take the case that $n = even$ and we apply these new settings it could happen that the number of robots actually moving, and therefore affecting the rotation discussed in Lemma 3.2, becomes odd. Thus leaving us with the same problem as before and allowing us to still use the same algorithm and assumptions about the model as when $n = odd$.

Lemma 3.8. *For any even $n > 0$, each robot learns the positions of the others within the first $O(\log^2 n)$ stages* w.h.p. *when using the approach of having more movement choices.*

The purpose of allowing a robot to make a choice from $\{C, A, S\}$ is that it will allow some permutations to exist where $n_c + n_a$ is odd, thus allowing the DISCOVERY algorithm to work as intended for cases of odd n.

From the proof for (Lemma 3.3) we know that when only the directions of C and A are used robots exchange batons when they collide and when this collision and exchange occurs the robot also takes on the directional behaviour of that baton as well. Therefore, in the case when S is also an option that may be chosen by a robot when a robot exchanges batons with a robot that chose S they receive their baton as usual and also inherit this choice to remain stationary. In the event where a robot who is in possession of a stationary baton experiences collisions from both sides at the same time then the two mobile robots that initiated the collisions will simply exchange batons with each other, leaving the stationary robot with its original stationary baton.

Now recall that when only the directions of C and A are used a robot will collide with every other robot twice during a round as shown in the proof for (Lemma 3.2). This can no longer be the case as the stationary baton will remain stationary for the stage and so will experience only one collision with every mobile robot during each round. Following on from this observation we can say that if a robot chooses a clockwise direction then it will experience a number of $(2n_a + n_s)$ collisions during a round. If the robot chooses an anti-clockwise location then it will experience a number of $(2n_c + n_s)$ collisions during a round and finally if the robot has chosen the stationary choice then it will experience $(-n_c + n_a)$ collisions per round.

Following on from this we can use a very similar argument from (Lemma 3.4) as the rotation index during a stage using the DISCOVERY algorithm is governed by those robots that are mobile. During a stage where $n_c + n_a$ is odd we can discover initial starting locations as we would if n was also odd. Essentially if a robot carried a stationary baton during a round it would "sit out" that round of discovery thereby allowing the robots that are mobile the chance to discover new starting locations during that round as if n was odd.

3.3.3 Amendment for unknown n

When n is unknown we will need another important observation. Consider a single stage where the direction of each of n batons is chosen uniformly at random. One can show that *w.h.p.* the batons form a (n_c, n_a)-configuration satisfying $|n_c - n_a| < 10\sqrt{n}$.

During the Discovery procedure, each robot is constructing a (partial) map of the initial positions of all robots. If $\gcd(n_c - n_a, n) = 1$, then in one stage each robot will learn the initial positions of all other robots (but, of course, will not know that if n is unknown). If we have $\gcd(n_c - n_a, n) > 1$, a robot visits (records the location of) at least $\frac{n}{|n_c - n_a|}$ initial robot positions during the stage. Assuming the robot collects the distance to first collision in each round, it also builds (or updates) estimates of positions of nearest neighbours in its map (when n is odd, these may coincide with positions the robot visits; when n is even these estimates are necessary to determine the entire map, as outlined in Section 3.3.2).

Because $|n_c - n_a| < 10\sqrt{n}$ *w.h.p.*, we note that (*w.h.p.*) a robot will visit at least $\sqrt{n}/10$ initial positions in any stage. Hence, after an initial small (constant) number of stages, a robot can use the number of positions visited to obtain a very good estimate (or overestimate) for n, except that it may not know if n is even or odd. a robot determines the parity of n during the Discovery procedure by observing if it actually visits the (calculated) positions of its nearest neighbours.

3.3.4 Sense of direction agreement

While a coherent sense of direction was not essential during the execution of procedure Discovery, we will need it to solve other problems, such as boundary patrolling where all robots are asked to move in one direction. Recall that when the robots start they may not share the same sense of direction.

Two types of stages (rounds) can be distinguished: (1) when the robots do not collide, and (2) when collisions happen. Note that an extra agreement procedure is not needed if a stage of type (1) occurs. When the robots do not collide (i.e. after one time unit they arrive back at their starting locations) they assume that their current direction is the clockwise direction.

Observe that the probability that all robots choose the same direction is very small ($\frac{2}{2^n} = \frac{1}{2^{n-1}}$). Therefore we focus on stages of type (2) where we also have the rotation index $r \neq 0$. When the *first* such stage occurs, the robots that experience $r > 0$ do not change their understanding of the clockwise direction, and those with $r < 0$ replace the clockwise direction with its anticlockwise counterpart.

The probability that a stage of type (2) with $r \neq 0$ occurs is $1 - \frac{1}{2^{n-1}}$ when n is odd and $1 - \binom{n}{n/2}/2^n - \frac{1}{2^{n-1}} \gg 1/2$ when n is even. Thus after $O(\log^2 n)$ stages of procedure Discovery the probability that agreement on sense of direction is reached is very high.

3.3.5 Equidistant distribution and boundary patrolling

In this section we consider application of location discovery to the boundary patrolling problem, see [42], where robots walk along the circle indefinitely with the goal of minimising the maximum time between two consecutive visits at any point located on the circle. During the location discovery procedure, robots always move with the unit speed. However, in order to obtain the equidistant distribution, that is a distribution along the circle where the space between the robots is equal, each robot must be able to set its speed as a value $0 \leq s \leq 1$ during the course of a single round.

Recall that on the conclusion of procedure Discovery every robot a_i is aware of the relative location of (or distance to) other robots. For equidistant distribution, during a single *adjustment round* each robot a_i is asked to move from p_i to a target position t_i, where $|t_{i+1} - t_i| = \frac{1}{n}$, for all $i \in [n]$.

Theorem 3.9. *In the considered distributed model of computation, n robots with the maximum speed one can reach equidistant distribution in one round, after the location of all robots have been discovered.*

Proof. Consider all distances $d_i = |p_{i+1} - p_i|$
and form a cyclic word $D = d_0, d_1 \dots d_{n-1}$ in which the end values d_0 and d_{n-1} are concatenated.

We say that this cyclic word D has *a period* w if, for all $i = 0, \dots, n-1$, $d_i = d_{i+w}$. Further, D is *non-periodic* if the shortest period of D is equal to n. Otherwise D is called *periodic*.

We distinguish two cases in relation to periodicity of D.

Case 1 D is non-periodic. A non-periodic cyclic word of length n generates n different cyclic rotations. In particular, D generates n cyclic rotations $r_0 = d_0 \dots d_{n-1}$, $r_1 = d_1 \dots d_0$, all the way to $r_{n-1} = d_{n-1} \dots d_{n-2}$. These rotations can be sorted lexicographically, where $r_i < r_j$ if $0 \leq x < y \leq n-1, d_{i+x} < d_{j+y}$ for all and $d_{i+x} = d_{j+x}$.

Since in this case all rotations are different there must exist the smallest rotation in this order, and let this be $r_s = d_s d_{s+1} \dots d_{s-1}$. Note that r_s uniquely identifies position p_s, i.e., the *breaking point* in D that forms rotation r_s. This means that all robots can compute the exact location of position p_i. This position becomes the reference point in the equidistant distribution round. In particular, during this round the robot a_s remains at position $t_s = p_s$, and every other robot p_i moves to the position t_i located at distance $\frac{i-s}{n}$ from t_s in clockwise direction (or anticlockwise, if negative).

Since the adjustment is performed in a single round, robot a_i moves towards t_i along the shortest route with the constant speed $0 \leq |t_i - p_i| < 1$. This is to allow every robot to arrive at its destination exactly at the end of the round. Note first that during this adjustment round the robots do not change their order on the circle. Thus if two robots move towards each other (from different directions) they never collide. Similarly, if two robots move in the same, say clockwise, direction they cannot collide. Otherwise, a faster robot would attempt at some point to overpass (collide with) a slower one while having a shorter distance to be traversed until the end of the round at a higher speed. Thus at the end of the adjustment round all robots arrive on time at their equidistant target positions.

Case 2 The shortest period of D is $u < n$. In this case we observe that u is a divisor of n and there are exactly u different cyclic rotations of D. Let $r_s = r_{s+u} = \dots = r_{s-u}$ be the smallest cyclic rotation. This time we have a set of $\frac{n}{u}$ breaking points $BP = \{p_s, p_{s+u}, \dots, p_{s-u}\}$.

During the adjustment round all robots residing at positions in BP do not move, i.e., $t_i = p_i$, for all $p_i \in BP$. Every other robot a_i can work out its relative location in the period including its geometric $|p_i - p_j|$ and rotation $|i-j|$ distances to the closest (in anticlockwise direction) breaking point $p_j \in BP$. During the adjustment round a_i traverses to its target position $t_i = p_j + \frac{|i-j|}{n}$ with the speed $|t_i - p_i|$. Finally, using a similar argument to the one adopted in Case 1 one can conclude that also in this case there will be no collisions and all robots will arrive on time at their equidistant target positions. ■

3.3.6 Boundary patrolling

In the *boundary patrolling problem* the mobile robots are asked to adopt movement trajectories such that the maximum time, taken across all points in space, between two consecutive visits by some (possibly different) robots is minimised.

Theorem 3.10. *In the considered distributed model of computation, n robots with the maximum speed one can adopt an optimal cyclic boundary patrolling strategy.*

Proof. On the conclusion of the adjustment round all robots start moving along the circle with maximum speed one in the same, e.g., clockwise direction. We show that this cyclic patrolling strategy is optimal. First note that this solution admits the idle time $\frac{1}{n}$ (the maximum time any point is unobserved). Note that if at any time t there exists a greater than $\frac{1}{n}$ gap between two robots a_i and a_{i+1}, the midpoint between a_i and a_{i+1} is at distance greater than $\frac{1}{2n}$ from both of them. This, in turn, means that this point was visited strictly before time $t - \frac{1}{2n}$ and will be visited strictly after time $t + \frac{1}{2n}$, admitting the idle time strictly greater than $\frac{1}{n}$. Therefore all robots must move with maximum speed either in the clockwise or anticlockwise direction. ■

3.4 Faster Algorithm

In this section a modified strategy for solving the location discovery problem on the ring is considered. This new strategy involves the robots remembering the decisions that they had made in the past in terms of the directions they picked at the start of each round. This small piece of information that is remembered actually helps in a big way by allowing the robot to learn from costly decisions it made in the past. These costly decision will become more evident later on but the basic idea of this faster approach is to attempt to prevent the robots ending a round at, "visiting", a previously discovered location. Through limiting the number of repeat "visits" to previously discovered locations we have been able to decrease the upper bound for this problem from $O(n \log^2 n)$ rounds to $n+O(\log^2 n)$ rounds.

Algorithm 2: FasterDiscovery(n: integer)

```
List $Points$, $History$, $VectorSet$
$\{C, A\}$ $dir$
$\mathbb{R}$ $location$
$Points, History, VectorSet \leftarrow \emptyset$
$location, i \leftarrow 0$
pick direction $dir$ from $\{C, A\}$ uniformly at random
append $location$ to $Points$
while $location == 0$ & $|points| < n$ & $location \notin Points$ do
    append $location$ to $Points$
    append $dir$ to $VectorSet$
    $move$ for 1 round and set $location$
end while
remove last element from VectorSet
$History \leftarrow VectorSet$
$VectorSet \leftarrow \emptyset$
while $|points| < n$ do
    IncreaseGranularity()
    remove last element from VectorSet
    $History \leftarrow VectorSet$
    $VectorSet \leftarrow \emptyset$
end while
return $Points$
```

3.4.1 Odd n

In the case for when n is odd then we perform a new FasterDiscovery algorithm as defined in Algorithm 2. The basic idea of the solution is that a robot picks a direction at random and walks for a stage discovering the starting locations of the other robots as it moves. After this initial stage is complete the robot then picks a new direction, also at random, and then follows the steps outlined in the IncreaseGranularity algorithm, shown in Algorithm 3. The new improved algorithm relies on an iterative approach to allow each robot to explore for one round in the chosen direction and then if the newly visited location has not been previously discovered the algorithm iterates over the list of stored directions used in previous stages using a saved history. However, if the visited point has been previously discovered the IncreaseGranularity algorithm comes to an end and the control of the robots movement returns to the main loop in the FasterDiscovery algorithm.

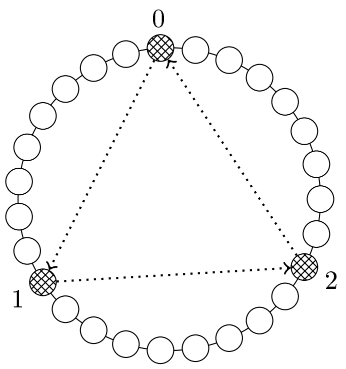
In this way it is possible for the robot to never repeat discovery patterns from previous stages over starting locations that have already been discovered as every

Algorithm 3: IncreaseGranularity()

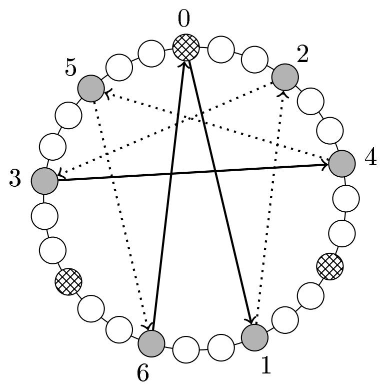
```
$\{C, A\}$ $originalDir$
$boolean\ badDir$
pick direction $dir$ from $\{C, A\}$ uniformly at random
$badDir = false$
$move$ for 1 round and set $location$
while $location = 0$ & $|points| < n$ & $location \notin Points$ do
    append $location$ to $Points$
    append $dir$ to $VectorSet$
    $originalDir = dir$
    while $location == 0$ & $|points| < n$ do
        foreach $set \in History$ do
            foreach $newDirection \in Set$ do
                $dir = newDirection$
                $move$ for 1 round and set $location$
                if $location \notin Points$ then
                    append $location$ to $Points$
                    append $dir$ to $VectorSet$
                else
                    $badDir = true$
                    $break$
                end if
            end foreach
            if $badDir == true$ then
                $break$
        end foreach
        if $badDir == true$ then
            $break$
    end while
end while
```

time the robot sets out on one of these previously walked patterns it will stop itself in the first round of that stage and start again with a new direction. The effect of this is that the running time for the discovery process is reduced from $O(n \log^2 n)$ rounds *w.h.p.* to $n + O(\log^2 n)$ rounds *w.h.p.* (see Lemma 3.11). The robots still need to perform the walking and colliding portion of the process, hence the $O(\log^2 n)$, however now we only need to do this once for each starting location discovered.

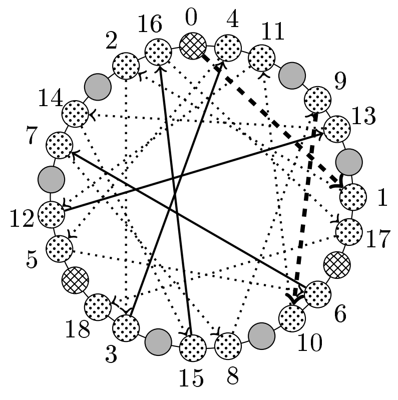
Figure 3.1 shows an example of the algorithm at work in a scenario where $n = 27$ and the robots were unable to find a *gcd*(n,rotation-index) = 1 in the first few stages. In this scenario the FasterDiscovery algorithm would allow for a faster solution than the original solution outlined earlier in this chapter. Figure 3.1



Stage 1
$n_c - n_a = 18$
$gcd(27, 18) = 9$
Vector = V_1
Vector Set = V_1, V_1, V_1
History saved = V_1, V_1



Stage 2
$n_c - n_a = 12$
$gcd(27, 18, 12) = 3$
Vector = V_2
Vector Set = (V_2, V_1, V_1, V_2, V_1, V_1, V_2)
History saved = (V_2, (V_1, V_1) V_2, (V_1, V_1))



Stage 3
$n_c - n_a = 7$
$gcd(27, 18, 12, 7) = 1$
Vector = V_3
Vector Set = (V_3, V_1, V_1, V_2, V_1, V_1, V_2, V_1, V_1,V_3, V_1, V_1, V_2, V_1, V_1, V_2, V_1, V_1)
History saved = (V_3, ((V_1, V_1), (V_2, (V_1, V_1), V_2, (V_1, V_1)),V_3, ((V_1, V_1), (V_2, (V_1, V_1), V_2, (V_1, V_1))))

FIGURE 3.1: A simple example of the algorithm at work when $n = 27$.

shows that during Stage 1 the robots have traveled along a $rotation - index$ of 18 thus giving a *gcd* of 9. From this the robots are able to complete the stage having discovered every starting location spaced at 9 starting locations apart, in this case this means that they have discovered their own starting location plus 2 other starting locations around the ring. Furthermore they have also found that by traveling for one round they move by a vector, given as V_1, meaning that after one stage they have traveled with a complete $VectorSet$ of V_1, V_1, V_1 and in doing so have visited the points labeled $0, 1$ and 2 in that order. The robot is also able to realise that the last vector it traversed brought it back to it's starting starting location (position 0), a starting location it had already discovered. This is important because it means that in future stages when the robots are iterating through the direction history portion of the algorithm they are able to save time by knowing they have no need to perform this last vector of the already discovered $VectorSet$, therefore saving 1 round of traversal in each iteration performed.

This can be seen clearly in Stage 2 of Figure 3.1. Notice how the robot will travel from it's original starting location (position 0) to a new location through a $rotation - index$ of 12 for 1 round to position 1. At this point the robot realises it has not been in this starting location before and so can switch to using the direction it chose during Stage 1. The robot then travels with this new direction for 2 rounds, thereby performing the actions mapped in the first $VectorSet$ stored in the robot's $History$ and visiting positions 2 and 3, before switching back to using the direction it had chosen at the start of Stage 2 and moving to position 4. Using this method the robot has discovered only starting locations that it had not previously discovered as it had learned that by traveling for a third round using the same direction it had used in Stage 1 it would end that round in a starting location it had previously discovered and in doing so would have been wasting time. However, the robots have not finished this stage yet as a stage only ends when the robot is either back at it's original starting location at the end of a round or all of the starting locations have been discovered. Therefore the robot now repeats itself by moving for one round using the direction it had chosen at the start of Stage 2 and then moving for another 2 rounds using the direction it had used during Stage 1. Finally, the robots move for 1 round using the direction chosen at the start of Stage 2. Stage 2 now comes to an end as the robots are all now back in their original starting locations having visited and subsequently discovered positions 5 and 6 as well.

In the final stage of the example given in Figure 3.1 the robots pick a direction again and the perform similar actions to the ones carried out in the second stage. That is each time a new starting location is discovered when traveling with the choice of direction from the current stage, the robots then iteratively perform the search process using the history produced from each of the previous stages, in the order of Stage 1, Stage 2, ... In this example Stage 3 finishes prematurity as all of the starting locations have been discovered. If this was not the case then the robots would have walked for another 10 rounds, following the pattern from the $VectorSet$, before arriving back at it's original starting location. In this example the robot moves from its starting location at position 0 to position 1 through a $rotation - index$ of 7. At this point the robot is aware this is a newly visited location and proceeds to adopt the previous moves it has made in the past when it has found a new location. This means that it first adopts the direction and subsequent moves used in Stage 1 for two rounds to move to position 2 and then 3 before adoption the direction and subsequent moves used in Stage 2 using the saved History. Doing this allows the robot to travel in the same manner as Stage 2 to discover positions $4, 5, 6, 7, 8$ and 9 before reverting back to its chosen direction for Stage 3. At this point the pattern begins again, with the robot first moving to position 10 and then iteratively going through its previous movements in Stage 1 and then Stage 2 to discover the final positions.

This method of discovery allows us to represent the ring as we have done in Figure 3.1 and to say that for each stage completed the granularity of the starting locations discovered had been increased to the $gcd(\text{n}, rotation-index_1, rotation-index_2, ..., rotation-index_i)$, where i is the stage that was completed last.

Lemma 3.11. *For any $n > 0$, each robot learns the positions of the others within the first $n + O(\log^2 n)$ rounds* w.h.p. *using the* FasterDiscovery *algorithm if the stationary (S) direction is allowed.*

Proof. Using the iterative method of the FasterDiscovery and IncreaseGranularity algorithms we can show that the total number of rounds needed to complete the *Location Discovery Problem* is $n + O(\log^2 n)$ rounds. This can be broken down into the number of rounds is required to do two things. Firstly, the n portion comes from the fact that, due to the lack of communication present in the system, each robot has to visit the *home bases* of each of the other n robots. The second portion of $O(\log^2 n)$ rounds comes from the fact that we are targeting a stage with a $gcd(n, \Psi, n_c - n_a) = 1$, where Ψ is the set of all previous $n_c - n_a$ from completed stages. Using the FasterDiscovery and IncreaseGranularity algorithms to achieve this robots start a new stage and move for one round if they finish a round at a *home base* that they have already learned in the past then the robots realise this error and instead start a new stage and pick a new random direction of movement. The number of times this happens in the pursuit of a stage with a $gcd(n, \Psi, n_c - n_a) = 1$ is $O(\log^2 n)$. This can be shown in the same way as in Lemma 3.5, where we prove that it takes $O(\log^2 n)$ rounds to achieve a successful stage, that is one where $gcd(n, n_c - n_a) = 1$. ∎

3.4.2 Even n

Previously when n is even we have had problems with the original Discover algorithm and needed to adapt it slightly to allow for detection of the first collision during each round. Using this technique and given enough time the robot could build up a picture of where the starting locations of all the robots where as there would never be a configuration where $\gcd(n_c - n_a, n) = 1$ and hence a stage where the robot could visit all of the starting locations on the ring. This issue can not be overcome in the same way if we make use of the FasterDiscovery algorithm as we do not allow for the required number of rounds for the robot to build up this reliable picture of where the starting locations are located. Instead what we can do is simply modify the FasterDiscovery and IncreaseGranularity algorithms to accept a third direction, stationary (S), along with clockwise (C) and anticlockwise (A). Robots would now make a choice between moving and S with equal probability and then if their choice was to move then they would make another random choice between C and A with equal probability. Allowing this change would mean that we can continue to use the steps in the the FasterDiscovery for each robot and still achieve a running time of $n + O(\log^2 n)$ rounds *w.h.p.*.

Lemma 3.12. *For any even $n > 0$, each robot learns the positions of the others within the first $n + O(\log^2 n)$ rounds* w.h.p. *using the* FasterDiscovery *algorithm if the stationary (S) direction is allowed.*

Proof. The purpose of allowing a robot to make a choice from $\{C, A, S\}$ is that it will allow some permutations to exist where $n_c + n_a$ is odd, thus allowing the FasterDiscovery and IncreaseGranularity algorithms to work as intended for cases of odd n.

As shown in Lemma 3.8, using the *Stationary* choice as well as *Clockwise* and $Anti - Clockwise$ the rotation of robots during a round remains the same as when only *Clockwise* and $Anti - Clockwise$ directions are allowed therefore in this setting when a stage occurs where $n_c + n_a$ is odd we can discover starting locations as we would if n was also odd.

Following on from this we can apply the FasterDiscovery and IncreaseGranularity algorithms to the problem as if n itself was odd and, as shown in Lemma 3.11, we end up solving the problem in $n + O(\log^2 n)$ rounds *w.h.p.*. ■

3.5 Conclusion

We presented a fully distributed randomised algorithm that solves the location discovery problem *w.h.p.* in $O(n \log^2 n)$ rounds. We then also extended this result to show that the location discovery problem can actually be solved much faster with the new algorithm that terminates with all starting positions discovered after $n + O(log^2 n)$ rounds.

We also show how this mechanism can be used to distribute n robots evenly on the circle and how to coordinate their joint effort in efficient boundary patrolling of the circle. We have shown that this result is also true if initially the robots are not aware of their number n and they have no coherent sense of direction.

In this chapter we focused on the case with known circumference. The question whether one can solve the location discovery problem for the case with known n and unknown circumference remains unanswered. Another promising direction would be the design of deterministic discovery algorithms in models where robots bear unique identifiers. One could also consider more complex network topologies in which likely more detailed information would have to be used by the robots to allow them to solve the discovery problem.

Chapter 4

Evacuation on the Line

This chapter introduces the Evacuation Problem, more specifically this chapter will cover work done in [34] that has been accepted to The 41st International Conference on Current Trends in Theory and Practice of Computer Science 2015 (SOFSEM'15).

We will consider the *group search problem*, or *evacuation problem*, in which k robots located on the line perform search for a specific poiont on the line that will be known as the *destination*. The robots are initially placed at the same origin on the line L and the target is located at an unknown distance d, either to the left or to the right from the origin. All robots must *simultaneously* occupy the *destination*, and the goal is to minimize the time necessary for this to happen. The problem with $k = 1$ is known as the *Cow Path Problem*, [18] and the time required for this problem is known to be $9d - o(d)$ in the worst case (when the cow moves at unit speed); it is also known that this is the case for $k \geq 1$ unit-speed robots. A clear argument for this claim is presented later in this chapter by showing a rather counter-intuitive result that was published in our paper [34]. Namely, independent of the number of robots, group search cannot be performed faster than in time $9d - o(d)$. We also examine the case of $k = 2$ robots with different speeds, showing a surprising result that the bound of $9d$ can be achieved when one robot has unit speed, and the other robot moves with speed at least $1/3$.

4.1 Introduction

Search problems are well-studied within the fields of operations research, computing, and mathematics. Indeed, nearly sixty years ago Bellman [25] asked a question that can be stated as follows: "A hiker is lost in a forest whose dimensions are known to her. What is the best path for her to follow to escape the forest?"

In general, search problems deal with a searcher looking for a hidden object (or "target"), with a goal of minimizing the time required to find it. Many versions of this problem can be considered, including variations in the environment (e.g., a geometric setting vs. a graph), whether the target is fixed or mobile, or if the target is a point in space or a boundary of a region or other curve, the use of a deterministic or randomized search strategy, and whether or not the searcher(s) have access to additional tools to aid the search (such as markers to drop in the environment) [18, 19, 27, 28, 30, 72, 77, 86, 89].

Search also naturally leads into the *rendezvous problem*, where two or more searchers seek to meet in an environment, and that problem itself lends itself to additional considerations of the inherent abilities of the searchers themselves, such as whether they have the same speed or different speeds, their ability to communicate and see their environment (typically over a limited distance), and if the searchers are able to follow the same or different search strategy, e.g. do the searchers have unique identifiers so they can adopt their own search method, or are they indistinguishable and therefore must use the same (randomized or deterministic) strategy? [10, 40, 55]. The book by Alpern and Gal [8] is a good survey of known results for both the search and rendezvous problems.

The focus of this chapter is the *group search problem* or *evacuation problem*, where k mobile robots, all starting from the origin on the line, must find and *simultaneously* gather at the target located at an unknown distance d from the origin. The inspiration for the name comes from consideration of an evacuation procedure of a building (one, say, that is on fire). Evacuation can be considered a very special case of the rendezvous problem mentioned earlier. The significant difference between rendezvous and evacuation is that all participants must gather at the same point in space, and at the same time. Some of the mobile robots might find the target location and leave it, only to return later, but the evacuation problem is only deemed to be "finished" when all mobile robots simultaneously gather at the target.

In the case of the line that is consider here, the most relevant previous results are in relation to the *cow-path problem*, a search problem that was introduced by Baeza-Yates, et al. in 1988 [18] and has since been considered in the same form and in different variations in [17, 19, 72, 81, 82, 89, 116].

The cow-path problem involves a single cow, Eloise[1], who is standing at a crossroads (defined as the origin) with w paths leading off into unknown territory. Traveling with unit speed, the goal of Eloise is to locate a target destination (say, a tasty patch of grass) that is at distance d from the origin in as small a time as possible. Eloise faces three difficulties: (1) she does not know the value of d, (2) she does not know which of the w paths leads to the goal, and (3) her eyesight is not very good, so she will not know she has found the goal until she is standing in it.

Baeza-Yates, et al. [18, 19] studied the cow-path problem, and proposed a deterministic algorithm they called *Linear Spiral Search* (detailed later) as a solution. In the case that $w = 2$ (two paths), this algorithm will find the goal in time at most $9d$, and they showed that this is optimal up to lower order terms. In the same work, the authors considered the case of $w > 2$ paths, showing an optimal (up to lower order terms) result of $\left(1 + 2\frac{w^w}{(w-1)^{w-1}}\right) d$ time bound to find the target using a deterministic search strategy.

Let us move away from cows, and into the world of *mobile robots* (robots) in what follows, opening up these robots to (possibly) have more computational power, memory, and/or communication ability than the average cow. The words *target* or *destination* will be used throughout this chapter to denote the goal of the search.

In [17] Baeza-Yates and Schott examined other variations of the cow-path problem in these stronger settings. They note the straightforward fact that if d is known by the robot then, in the worst case, it must travel for $3d$ units of time (for $w = 2$ paths). They also considered cases involving two or more robots having uniform speed. If the robots are able to communicate arbitrarily far away, then a total distance of at least $2d$ must be traveled to find the destination, and $4d$ if both robots must reach the destination. Baeza-Yates and Schott showed the total distance traveled when no communication is present, *and* both robots must reach the goal is also $9d$, the same time it would take a single robot.

[1]From the book *Eloise and the Old Blue Truck*, by Kennon Graham, illustrated by Florence Sarah Winship, a childhood favorite of one of the authors.

The previous results all applied to deterministic search algorithms. Kao, et al. [82] examined the first randomized algorithm for the cow-path problem and, for the case of $w = 2$ paths, obtained an optimal randomized $4.59112d$ bound for the search time. Those authors also give a bound for $w > 2$ paths, which they conjecture to be an optimal randomized strategy.

The cow-path problem, with either one or two robots, cannot be solved in time smaller than $9d$ (up to lower order terms), where d is the distance from the origin to the destination, and the robots have unit speed. This result is proved, and re-proved in various fashions, in [17, 19, 72, 81, 82, 89, 116]. However, [17] seems to claim that if the number of robots is greater than two, then the evacuation procedure can be performed in a smaller time. This claim is dispute here in first part of this chapter through a proof showing that $9d$ is also optimal (up to lower order terms) when the number of robots is at least 2. By doing so an alternative way of proving the lower bound of $9d$ than the papers previously mentioned have done can be presented.

In the second part of this chapter, there will be some discussion around the study of the evacuation problem where mobile robots have different maximum speeds. It will be shown, with somewhat surprising results (to the authors at least), that when there are two (or more) mobile robots, one with unit speed and the others having maximum speed at least $1/3$, then the evacuation problem can still be performed in time at most $9d$. The authors believe that this is the first result regarding the evacuation problem with mobile robots having different maximum speeds, and hope to inspire further work in this direction. Indeed, the authors know of no prior work in the field of search, rendezvous, or evacuation that considers mobile robots with differing maximum speeds.

4.2 Results

This chapter looks at the scenario where there are k robots on the line, all starting at the origin. Here, work is done in the restricted setting where communication between robots is only possible when they are in contact (i.e. occupy the same location), but it is also assumed that any communication occurs instantaneously.

This chapter examines the *evacuation problem* where all k robots must simultaneously occupy a target located at an unknown distance d from the origin. Note this does not preclude a robot from finding the target and moving away from it (to return later), but rather that the evacuation problem is only completed when all k mobile robots are on the target at the same time. The aim is to achieve this goal in as small a time as possible. It is assumed here that d is a positive integer, but most of the results achieved here can be generalised for rational or real values of d, provided that d is not too small.

In Section 4.2.1 $k(\geq 2)$ *Mobile Robots*, $\mathcal{MR}$s, having a uniform speed that (by re-scaling time) will set to 1 are considered. The Linear Spiral Search method described in [18, 19] in which a single robot can find a target in time at most $9d$ can be briefly recalled here, as well as recalling a coordinated method for two robots to solve evacuation in time at most $9d$.

Here a new proof of a lower bound of $9d - o(d)$ for the evacuation time of two or more robots having unit speed will be given. (Theorem 4.5)

In Section 4.3 the case when $k = 2$ and the robots have different maximum speeds will be looked at. Where the speeds of the robots will be normalised by setting the speed of the fastest robot to 1 and then setting the speed of the slower robot proportionately. It will be demonstrated in this chapter that, provided the speed of the slower robot is at least 1/3, then the $9d$ evacuation time bound can still be achieved. (Theorem 4.7)

In these considerations time-space diagrams will be used to support the reasoning and the proofs. A time-space is a $2d$-plane with the horizontal axis representing location on the line L and the vertical axis refers to the time t. For the purpose of this chapter only the half-plane where the values of time are positive will be taken into account. In this context, the trajectory adopted by a robot can be described as a function of time t to give a location on L.

4.2.1 Multiple robots with uniform speeds

As mentioned earlier, there has been much previous work done in this area of the problem before. However, the goal of this section is to provide a clear and complete explanation to the claim of the $9d - o(d)$ worst case in this setting for multiple searchers with uniform speeds. d will be used throughout this chapter to denote the destination as well as the (unknown) distance to that destination. This should not cause confusion as the meaning should be clear from the context.

For completeness we first recall what is known in the case of one or two searchers.

The strategy for this time bound involves at least one robot traveling to the right and at least one to the left until the destination is found or they are alerted to the destination being found elsewhere on the line. If the robot finds d then immediately it turns around and heads in the opposite direction with its maximum speed so that it is able to catch up with the other robots and inform them of the discovery of d. Once informed, a robot heads towards d with its maximum speed. After all robots have been informed, then they will gather at the target to complete the evacuation procedure.

4.2.2 A basic strategy for a single mobile robot

As a brief reminder, a search strategy in the case of a single robot for two paths was outlined in [19], referred to as Linear Spiral Search by the authors in that paper. This search strategy is given as Algorithm 4, where for simplicity we consider the two paths to be a line in this case.

Algorithm 4: A "doubling strategy" for a single mobile robot

```
begin
    $r \leftarrow 1$
    $dir \leftarrow$ left          /* $dir \in \{$left, right$\}$ */
    while (Destination not found) do
        Walk distance $r$ in direction $dir$ and return to the origin /* (Stop at
        destination if found.) */
        Reverse $dir$
        $r \leftarrow 2 \cdot r$
    end while
end
```

This deterministic search strategy for a single robot yields the search time of $9d$, which is optimal up to lower order terms [19, Theorem 2.1].

4.2.3 Evacuating two mobile robots on the line

For the evacuation problem with two robots on the line (or two paths), there are at least two strategies that will yield a $9d$ upper bound for the problem.

One strategy is that each robot ignores the existence of the other robot and simply executes their own version of Algorithm 4, independently of the other (in fact, each robot can independently begin by going left or right at the start of their own procedure). This will clearly give an evacuation time of at most $9d$, since an robot that finds the destination simply waits for the other.

A second strategy coordinates the use of the searchers to find the target.

Once the target is located, the robot that finds it informs the other, and they both return to the destination together. In order for this strategy to work, the robots must use a speed slower than their maximum speed during the initial exploration phase, so that the "finder" is able to catch up with (and inform) the other searcher.

Algorithm 5 is one such coordinated evacuation procedure. The two mobile robots move with a speed of α during the "exploration" phase where they are searching for the *evacuation point*, switching to maximum speed to inform the other robot once the destination is found, and to move to/return to the destination. Later in this chapter it is argued that a speed of $\alpha = 1/3$ gives the $9d$ search bound.

Theorem 4.1. *Algorithm 5, with $\alpha = \frac{1}{3}$, gives an evacuation procedure with time bound $9d$, where d is the distance from the origin to the destination.*

Proof. Consider the time-space diagram in Fig. 4.1. The vertical axis is time and the horizontal one is distance.

So the two robots start at the origin and move along the red lines, until one of them finds the target. The one who finds the target (e.g. the right one in Fig. 4.1) then moves at maximum speed (along the green line) to inform the other robot. Finally, the pair moves to/returns to the destination at maximum speed (this final part of the movement is not shown on the diagram).

Algorithm 5: Coordinated evacuation for two mobile robots on the line

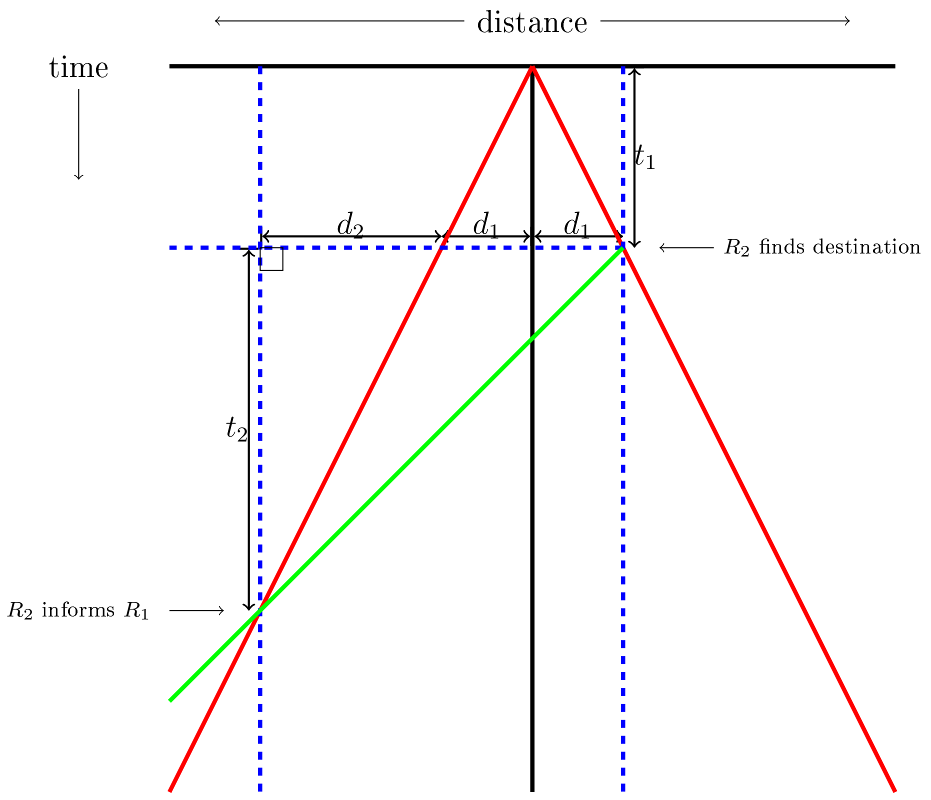
```
/* The robots are $R_1$ and $R_2$, with (current) speeds $s_1$ and $s_2$, resp. */
begin
    $s_1 \leftarrow \alpha$,  $dir(R_1) \leftarrow$ left
    $s_2 \leftarrow \alpha$,  $dir(R_2) \leftarrow$ right
    while (Destination not found) and (Not informed) do
        $R_i$ moves in direction $dir(R_i)$ at speed $s_i$
        if ($R_i$ finds the destination) then
            InformOtherrobot($R_i$)    /* See subroutine below */
        end if
    end while
    /* Now both robots know the direction of the target */
    $s_1 \leftarrow 1$,  $dir(R_1) \leftarrow$ direction towards target
    $s_2 \leftarrow 1$,  $dir(R_2) \leftarrow$ direction towards target
end


/* Subroutine to inform the other robot */
begin
    $s_i \leftarrow 1$,  $dir(R_i) \leftarrow$ opposite of current direction
    while (Not encountered other robot) do
        $R_i$ moves in direction $dir(R_i)$ at speed $s_i$
    end while
    Inform other robot of (direction of) destination
end
```

Thus, using the labels in Fig. 4.1, there are the following definitions:

$$\begin{aligned}
d_1 &= \text{The distance from the origin to the destination } (= d).\\
t_1 &= \text{Time to discover the destination by one robot.}\\
t_2 &= \text{Additional time for this robot to inform the other robot.}\\
d_1 + d_2 &= \text{The distance from the origin where the two robot will meet}\\
&\quad \text{in this scenario.}\\
\alpha &= \frac{d_1}{t_1} = \frac{d_2}{t_2} = \text{Speed used during initial exploration, and by the second}\\
&\quad \text{robot until it is informed of the location of the destination.}
\end{aligned}$$

Therefore, the total evacuation time is $t_1 + t_2 + 2d_1 + d_2$. The $2d_1 + d_2$ terms come from the time for the pair of robots to travel to the destination (after R_1 has been informed of the location of the target), as they are using their maximum (unit) speed, and must travel a total distance of $2d_1 + d_2$.

FIGURE 4.1: Time/space diagram for the evacuation procedure in Algorithm 5.



The rest is a few calculations, as the goal here is to have

$$t_1 + t_2 + 2d_1 + d_2 \leq 9d_1. \tag{4.1}$$

Using that $t_2 = 2d_1 + d_2$ (since the "finder" takes this much time to inform the other robot, and moves a distance of $2d_1 + d_2$ in doing so), it an then be found that

$$\begin{aligned} t_2 &= 2\alpha t_1 + \alpha t_2 \\ \text{or } t_2 &= \frac{2\alpha}{1-\alpha} t_1 = \frac{2}{1-\alpha} d_1, \\ \text{and } d_2 &= \alpha t_2 = \frac{2\alpha}{1-\alpha} d_1. \end{aligned}$$

Using these expressions for t_1 and t_2 in Equation (4.1), the goal is for

$$\begin{aligned}
\frac{1}{\alpha}d_1 + \frac{2}{1-\alpha}d_1 + 2d_1 + \frac{2\alpha}{1-\alpha}d_1 &\leq 9d_1 \\
\frac{1}{\alpha} + \frac{2}{1-\alpha} + 2 + \frac{2\alpha}{1-\alpha} &\leq 9 \\
(1-\alpha) + 2\alpha + 2\alpha(1-\alpha) + 2\alpha^2 &\leq 9\alpha(1-\alpha) \\
9\alpha^2 - 6\alpha + 1 &\leq 0 \\
(3\alpha - 1)^2 &\leq 0. \qquad (4.2)
\end{aligned}$$

Equation (4.2) is only satisfied when $\alpha = \frac{1}{3}$. So using Algorithm 5 with an "exploration speed" of $\alpha = \frac{1}{3}$ gives an evacuation procedure for two robots that works in time at most $9d$. ■

4.2.4 Energy conservation by coordinated evacuation

As noted earlier, the $9d$ bound on the time to complete the evacuation can be achieved by each robot using its own (independent) version of Algorithm 4. (Indeed, each robot can choose to start the procedure by first searching left or right independently of the other robot, and otherwise follow the doubling strategy of Algorithm 4.) Here this chapter remarks that the coordinated strategy of Algorithm 5 provides a benefit over each robot simply using their own version of Algorithm 4.

Namely, there is a savings if we consider the *total distance* traveled by the robots, and therefore a savings in terms of energy consumption. For Algorithm 4, each robot (in the worst-case) can travel a distance of $9d$, for a combined total distance traversed by the pair of $18d$, if each acts separately from the other.

In Algorithm 5, the total distance traveled is $d_1 + d_1 = 2d$ (for the "discovery phase"), $2d_1 + 2d_2 = 4d$ (for the "inform phase"), and $4d_1 + 2d_2 = 6d$ (to reach the evacuation target). Thus, the total combined distance traveled by both robots is $12d$, a savings of 1/3 of the total cost of the non-coordinated evacuation procedure.

4.2.5 A lower bound

In [17], the authors make the statement "if we have more [than two] robots, we can have two robots searching and coming back to certain points, while other robots can carry messages between the searchers until the goal is found. In this case, the goal can be reached in a smaller time."

It is unclear to these authors whether the authors of [17] are claiming that the evacuation problem can be solved in time smaller than $9d - o(d)$ given using more than two searchers. In any case, we would dispute such a claim, and here we want to give a new proof that $9d$ is a lower bound (up to lower order terms) on the evacuation problem (for any number of robots). We remind the reader that the lower bound of $9d - o(d)$ was proven to be optimal for a single robot in [19]. We want to investigate the lower bound for two or more robots with a maximal speed of 1.

So here we assume that there are at least two robots performing the group search problem. To facilitate our proof, we first define some notation. We suppose that the robots performing the evacuation procedure are following some fixed (but unknown) algorithmic procedure, which may or may not be coordinated. The only restrictions we impose are the ones mentioned earlier, that robots can only communicate when they occupy the same point, that this communication is instantaneous, and that the maximum speed is 1.

Definition 4.2. For $t > 0$, we define $\alpha(t) \in \left[0, \frac{\pi}{2}\right]$ as the angle, measured in radians, as follows:

$$\alpha(t) = \sup_{t' \geq t} \left\{ \arctan\left(\frac{x}{t'}\right) : (x, t') \in E \right\},$$

where E is the set of all pairs (x, t') such that some robot is at distance x from the origin at time t'.

In other words, $\alpha(t)$ defines a symmetric cone (centered around the origin) of size $2\,\alpha(t)$ in the time/space diagram that contains all terrain that is ever explored from time t to time ∞ during the evacuation procedure, if we assume that the evacuation target does not actually exist, so that the robots will be exploring the x-axis forever. This cone, inspired by Minkowski Spacetime [96], exists as two robots exploring with a maximum speed $\tan(\alpha(t))$ could never travel fast enough to travel beyond the boundaries of this cone.

It is easy to see that $\tan(\alpha(t)) \leq 1$ for all $t > 0$ since the maximum speed of the robots is 1.

We note the following facts without proof.

Fact 4.3. For any sequence $0 < t_1 < t_2 < \ldots$, we have $\alpha(t_1) \geq \alpha(t_2) \geq \ldots$. In other words, for any increasing sequence of numbers $\{t_i\}_{i=1}^{\infty}$, the sequence $\{\alpha(t_i)\}_{i=1}^{\infty}$ is a non-increasing sequence.

Fact 4.4. $\lim_{i\to\infty} \alpha(t_i) \overset{\text{def}}{=} \alpha$ exists, and is independent of the particular increasing series of numbers $\{t_i\}_{i=1}^{\infty}$ chosen.

The previous fact follows as the non-increasing sequence $\{\alpha(t_i)\}_{i=1}^{\infty}$ is bounded below (by 0), and, hence by the monotone convergence theorem, has a limit. (We can alternatively express Facts 4.3 and 4.4 in terms of the tangents of the angles.)

The following theorem and proof shows us if there exists any point that was reached by a robot outside of the previously mentioned cone of potentially explored space that happened to be the *evacuation point* then to complete the evacuation procedure in (at most) $9d$ it must also be the case that $\tan(\alpha) = \frac{1}{3}$.

Theorem 4.5. *Suppose that there are two robots performing the evacuation. If* $\tan(\alpha) \neq \frac{1}{3}$*, where* α *is the limit defined in Fact 4.4, then there exists* $d > 0$ *and* $\delta > 0$ *such that the evacuation procedure takes time at least* $(9 + \delta)d$*.*

Proof. For the sake of this proof, we may suppose that there is an "adversary" who decides where to place the evacuation target, provided that this point has not yet been visited by any robot in the evacuation procedure.

Given $\varepsilon > 0$, let us pick t_0 and $t_1 > t_0$ large enough so that:

(a) $|\tan(\alpha(t_0)) - \tan\alpha| < \varepsilon$,

(b) the position at time t_1, $z(t_1)$, of an robot Z also satisfies $\left|\frac{|z(t_1)|}{t_1} - \tan(\alpha)\right| < \varepsilon$ (Remark: We assume, without loss of generality, that the value of $\alpha(t)$ in an interval around t_1 is defined by an robot located to the right of the origin. Otherwise, we may consider a similar argument to the one that follows where $\alpha(t_1)$ is defined by an robot to the left of the origin. Hence, under our assumption, the robot could be at the point labeled Z in Fig. 4.2.5.), and

(c) the line from Z extending backwards in time at a 45° angle to the time-axis that intersects the cone defined by the angles $\pm(\alpha \pm \varepsilon)$ does so after the time t_0. (See Fig. 4.2.5.)

The purpose of the last condition is that the shaded region shown in the figure has been *un*visited by any robot (as has the corresponding region on the right-hand side of the figure, but we have not shaded that region).

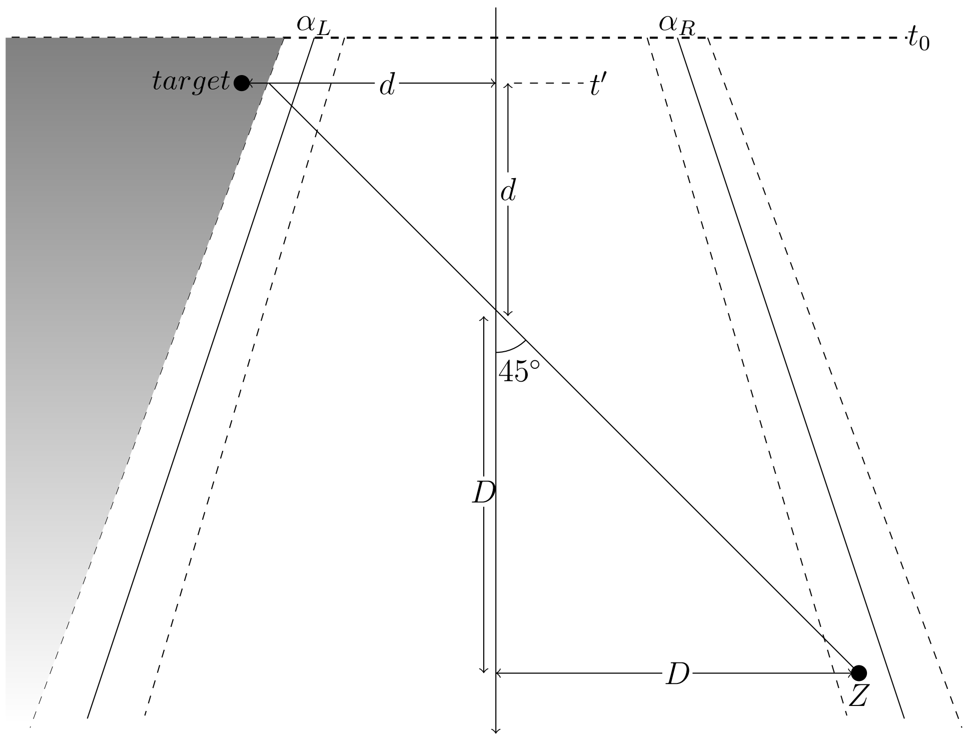


FIGURE 4.2: Time/space diagram of configuration used to establish the lower bound

Given this choice of t_0 and t_1, the evacuation point is placed slightly inside of the shaded region, as shown in Fig. 4.2.5.

With this configuration as labeled, we can make the following conclusions:

The earliest time that the evacuation point, at distance d from the origin, can be found is at time t', where t' satisfies $\tan(\alpha + \varepsilon + \zeta') = \frac{d}{t'}$, for a small $\zeta' > 0$ (to guarantee the target is in the unexplored region). Note that we can also choose ζ' so that $\zeta' < \varepsilon$.

This means that the earliest time that robot Z could learn about the existence of the target is at the time $t' + d + D$, where D satisfies $\left|\frac{D}{t'+d+D} - \tan(\alpha)\right| < \varepsilon$. This is because Z is at the specified location in the diagram at time $t' + d + D$, and obtaining the information at an earlier time would violate the “speed of light” in this time frame (which is the maximum speed of 1, indicated by the line that makes a 45° angle with the time axis).

Finally, this means that the earliest that robot Z could arrive at the evacuation point is at time $t' + 2 \cdot (d + D)$, since Z would require time $d + D$ to travel to the evacuation point from its current location at full speed.

The remaining part of this argument is some calculations in order to attempt to lower bound the sum $t' + 2 \cdot (d + D)$. First we have that

$$\begin{aligned} \tan(\alpha + \varepsilon + \zeta') &= \frac{d}{t'} \quad \text{as already noted, and} \qquad (4.3) \\ \tan(\alpha \pm \zeta'') &= \frac{D}{D + d + t'} \quad \text{for a small } 0 < \zeta'' < \varepsilon \text{ (with sign} \qquad (4.4) \\ &\qquad \text{depending upon the exact location of } Z). \end{aligned}$$

Using Taylor’s Theorem (see your favorite beginning calculus book), we note that we can write

$$\begin{aligned} \tan(\alpha + \varepsilon + \zeta') &= \tan\alpha + (\sec^2\alpha)(\varepsilon + \zeta') + g(\delta')(\varepsilon + \zeta')^2 \quad \text{and} \qquad (4.5) \\ \tan(\alpha \pm \zeta'') &= \tan\alpha \pm (\sec^2\alpha)(\zeta'') \pm g(\delta'')(\zeta'')^2 \qquad (4.6) \end{aligned}$$

where $g(z) = 2\sec^2 z \tan z$, $0 < \delta' < \varepsilon + \zeta'$, and $0 < \delta'' < \zeta''$.

The signs in (4.6) depend upon the position of Z. For the location of Z as given in Fig. 4.2.5, we have that (4.6) is actually (using the appropriate signs):

$$\tan(\alpha - \zeta'') = \tan\alpha - (\sec^2\alpha)(\zeta'') - g(\delta'')(\zeta'')^2. \qquad (4.7)$$

(The case for Z located on the other side of the angle labeled α_R is similar to the analysis we give below, and is left to the reader.)

Then, for small ε, ζ', and ζ'', we can find (small) constants C_1 and C_2 (that depend upon ε, ζ', and ζ'') such that

$$\begin{aligned} &\tan(\alpha + \varepsilon + \zeta') \leq \tan\alpha + (\sec^2\alpha)(\varepsilon + \zeta') + C_1(\varepsilon + \zeta')^2 \qquad (4.8) \\ \text{and} \quad &\tan\alpha - (\sec^2\alpha)(\zeta'') - C_2(\zeta'')^2 \leq \tan(\alpha - \zeta''). \qquad (4.9) \end{aligned}$$

We note that $C_1 \to 0$ as $\varepsilon + \zeta' \to 0$ and, similarly, $C_2 \to 0$ as $\zeta'' \to 0$.

In what follows, in order to simplify the notation somewhat, we will let $x = \tan\alpha$ and recall, of course, that $\sec^2\alpha = 1 + \tan^2\alpha = 1 + x^2$.

Therefore, from (4.3) and (4.4), and (4.8) and (4.9) we can write:

$$t' \geq \frac{d}{x + (1+x^2)(\varepsilon + \zeta') + C_1(\varepsilon + \zeta')^2}, \quad \text{and} \tag{4.10}$$

$$\frac{D}{D+d+t'} \geq x - (1+x^2)(\zeta'') - C_2(\zeta'')^2, \quad \text{from which we get} \tag{4.11}$$

$$D \geq \frac{(d+t')\,(x - (1+x^2)(\zeta'') - C_2(\zeta'')^2)}{1 - x + (1+x^2)(\zeta'') + C_2(\zeta'')^2}. \tag{4.12}$$

(Again, in the other case to consider for the location of the robot Z, one can obtain similar inequalities to use as lower bounds.)

The earliest time that robot Z can complete the evacuation procedure is $t' + 2 \cdot (d + D)$ which, using (4.10) and (4.12) is lower bounded by the function, after some simplification,

$$d \cdot h(x, \varepsilon, \zeta', \zeta'', C_1, C_2) \overset{def}{=} d\left\{\frac{1}{x + (1+x^2)(\varepsilon+\zeta') + C_1(\varepsilon+\zeta')^2} + 2 \right.$$
$$\left. + \left(\frac{2\,(x - (1+x^2)(\zeta'') - C_2(\zeta'')^2)}{1 - x + (1+x^2)(\zeta'') + C_2(\zeta'')^2}\right) \cdot \left(1 + \frac{1}{x + (1+x^2)(\varepsilon+\zeta') + C_1(\varepsilon+\zeta')^2}\right)\right\}.$$

Recall that if $\varepsilon + \zeta' \to 0$, then $C_1 \to 0$, and if $\zeta'' \to 0$, then $C_2 \to 0$.

So let us consider the function $f(x) = \frac{1}{x} + 2 + \left(\frac{2x}{1-x}\right) \cdot \left(1 + \frac{1}{x}\right)$. We claim that $f(x) \geq h(x, \varepsilon, \zeta', \zeta'', C_1, C_2)$ for a fixed x, and for all small enough $\varepsilon, \zeta', \zeta'', C_1$, and C_2, and $h(x, \varepsilon, \zeta', \zeta'', C_1, C_2)$ increases to $f(x)$ as $\{\varepsilon, \zeta', \zeta'', C_1, C_2\}$ all approach 0.

Elementary calculus tells us that $f(x)$ is minimized, under the restriction that $0 < x < 1$, when $x = \frac{1}{3}$. In this case, we have $f\left(\frac{1}{3}\right) = 9$, and $f(x) > 9$ for any other value of $x \in (0,1) - \left\{\frac{1}{3}\right\}$.

We therefore claim that for any other value of $x \in (0,1) - \left\{\frac{1}{3}\right\}$, since $f(x) > 9$, and since h increases with decreasing values of $\{\varepsilon, \zeta', \zeta'', C_1, C_2\}$, we can find (suitably small) ε, ζ', and ζ'' (and corresponding C_1 and C_2), such that $h(x, \varepsilon, \zeta', \zeta'', C_1, C_2) > 9$ as well.

This would mean that if $x(= \tan\alpha) \neq \frac{1}{3}$, then there exists a $\delta > 0$ such that $t' + 2 \cdot (d + D) \geq (9 + \delta)d$, proving the result of the theorem. ■

Intuitively, Theorem 4.5 tells us that the leftmost and rightmost boundaries of the region explored by the robots must (in the limit) grow an average of 1/3 unit distance per unit of time in order to successfully accomplish evacuation in time (at most) $9d$.

For more than two robots, we may consider the leftmost and rightmost robot at any time. The robots are anonymous, so to us (as outside observers), we cannot tell the difference if two robots "cross over" or if they meet each other and "bounce". The region that has been explored by a set of robots will still consist of a single connected segment in the line. Hence, we can conclude the following result just by considering the leftmost and rightmost robot at any moment in time, and repeating the proof of Theorem 4.5.

Corollary 4.6. *For two or more robots, if* $\tan(\alpha) \neq \frac{1}{3}$*, then there exists* $d > 0$ *and* $\delta > 0$ *such that the evacuation procedure takes time at least* $(9 + \delta)d$*.*

4.3 Robots having different maximum speeds

Now we consider two cases involving mobile robots having different maximum speeds. As before, by rescaling, we assume the maximum speed is 1. We call a mobile robot with maximum speed 1 a "fast robot". A mobile robot with speed s, where $0 < s < 1$ shall be called a "slow robot". We use the notation $\mathcal{FMR}$ to refer to the "faster" mobile robot. Similarly, we will use $\mathcal{SMR}$ to refer to the "slower" mobile robot.

Section 4.3.1 deals with the special case of one fast robot and one slow robot. In the case that $s \geq \frac{1}{3}$, we show that evacuation can still be accomplished in time $9d$, a fact that these authors found surprising when we first discovered it.

Section 4.3.2 deals with the case of two fast robots and one slow robot. Even in this case, if the slow robot is not too slow (in particular, if $s \geq \frac{1}{5}$), then evacuation can still be performed in time $9d$.

4.3.1 One fast, one slow

For one $\mathcal{FMR}$ and one $\mathcal{SMR}$ we will show that, provided $s \geq \frac{1}{3}$, the $9d$ evacuation time bound still holds using a coordinated strategy for the two mobile robots. A picture that hints at the strategies of the two robots can be seen in Fig. 4.3, but we give some brief discussion of each strategy in what follows.

4.3.1.1 The $\mathcal{FMR}$'s strategy.

The $\mathcal{FMR}$ searches for the evacuation point as if the $\mathcal{SMR}$ is not there, using the doubling strategy described in Algorithm 4 (always traveling at its maximum speed). The $\mathcal{FMR}$ follows this strategy until the evacuation target is located. Having found the target, mooving with its maximum speed of 1 it immediately seeks to make contact with the $\mathcal{SMR}$. Both robots then walk together to the evacuation target with the full speed s of the $\mathcal{SMR}$. Fig. 4.3 shows the exploration path the $\mathcal{FMR}$ takes to find the target as the solid black line, which is simply the doubling strategy from before.

One point to keep in mind is that the $\mathcal{FMR}$ knows the strategy of the $\mathcal{SMR}$, so the $\mathcal{FMR}$ knows in which direction to travel in order to find and inform the $\mathcal{SMR}$ once it locates the target.

4.3.1.2 The $\mathcal{SMR}$'s strategy

The slow mobile robot is obviously unable to mimic the path of the $\mathcal{FMR}$ due to its reduced maximum speed. Somewhat counter-intuitively, even if $s > \frac{1}{3}$, the $\mathcal{SMR}$ is instructed to use speed $\frac{1}{3}$ and follow the dashed path outlined in Fig. 4.3. It follows such a path until it is informed by the $\mathcal{FMR}$ of the location of the evacuation target, and then proceeds at maximum speed, i.e. s, to reach that destination.

The $\mathcal{SMR}$ is following its own "doubling strategy", but this takes more time to execute than it does for the $\mathcal{FMR}$. In particular, initially the $\mathcal{SMR}$ stays at the *origin* for 4 units of time, and then begins its own movements. After that, it uses a "doubling strategy" to move to distances $1, 2, 4, 8, 16, \ldots$ from the origin (on opposite sides of the origin, i.e. moving to distance 1 to the left, taking three units of time, returning to the origin, then to distance 2 to the right and returning, then to distance 4 to the left, etc.) Recall that the $\mathcal{SMR}$ is moving at speed 1/3, and, hence takes time $2 \cdot 3 \cdot 2^k$ to execute one portion of its "doubling" move, i.e. moving to distance 2^k and returning to the origin.

Observe that the $\mathcal{SMR}$ and $\mathcal{FMR}$ will meet at certain pre-defined times and locations during their trajectories. All of the meeting points, aside from the first one at the origin while the $\mathcal{SMR}$ is not moving, occur at locations that were originally extreme points (i.e. turning points) of the trajectory of the $\mathcal{FMR}$. For example, the two robots will meet at distances $1, 2, 4, \ldots$ from the origin (again, on opposite sides of the origin).

Under this strategy, the $\mathcal{SMR}$ will never discover the evacuation point before the $\mathcal{FMR}$ does so, and therefore must simply keep walking in this way until the $\mathcal{FMR}$ comes to inform it of the location of the evacuation target and take it there with the maximum speed of the $\mathcal{SMR}$.

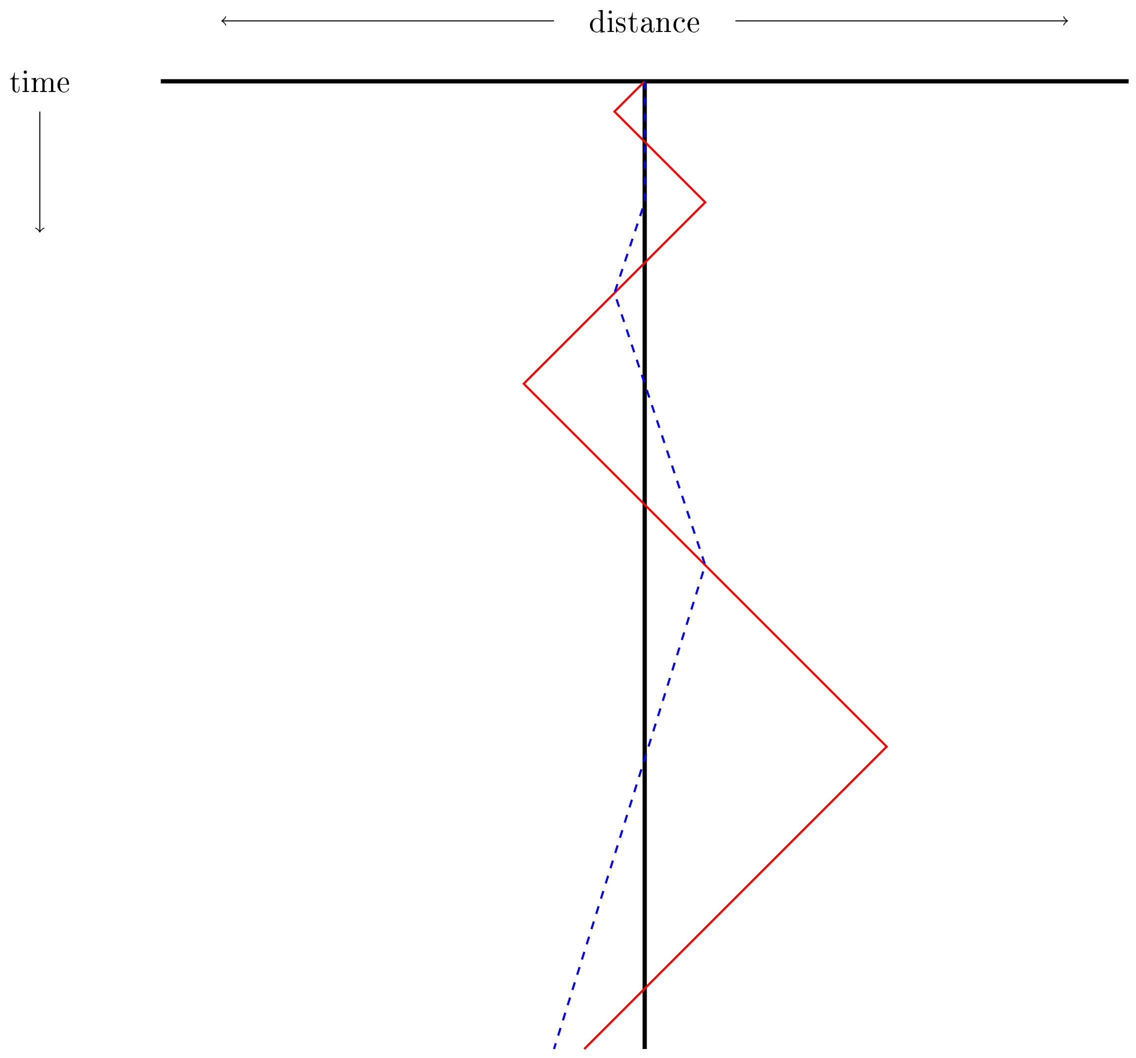


FIGURE 4.3: An optimal strategy for two different speeds, where the slower robot has a speed at least 1/3 the speed of the fast robot.

We also remark that by following the particular outlined strategy, the $\mathcal{SMR}$ is at the origin at the same moment that the $\mathcal{FMR}$ is at one of the turning points of its movements.

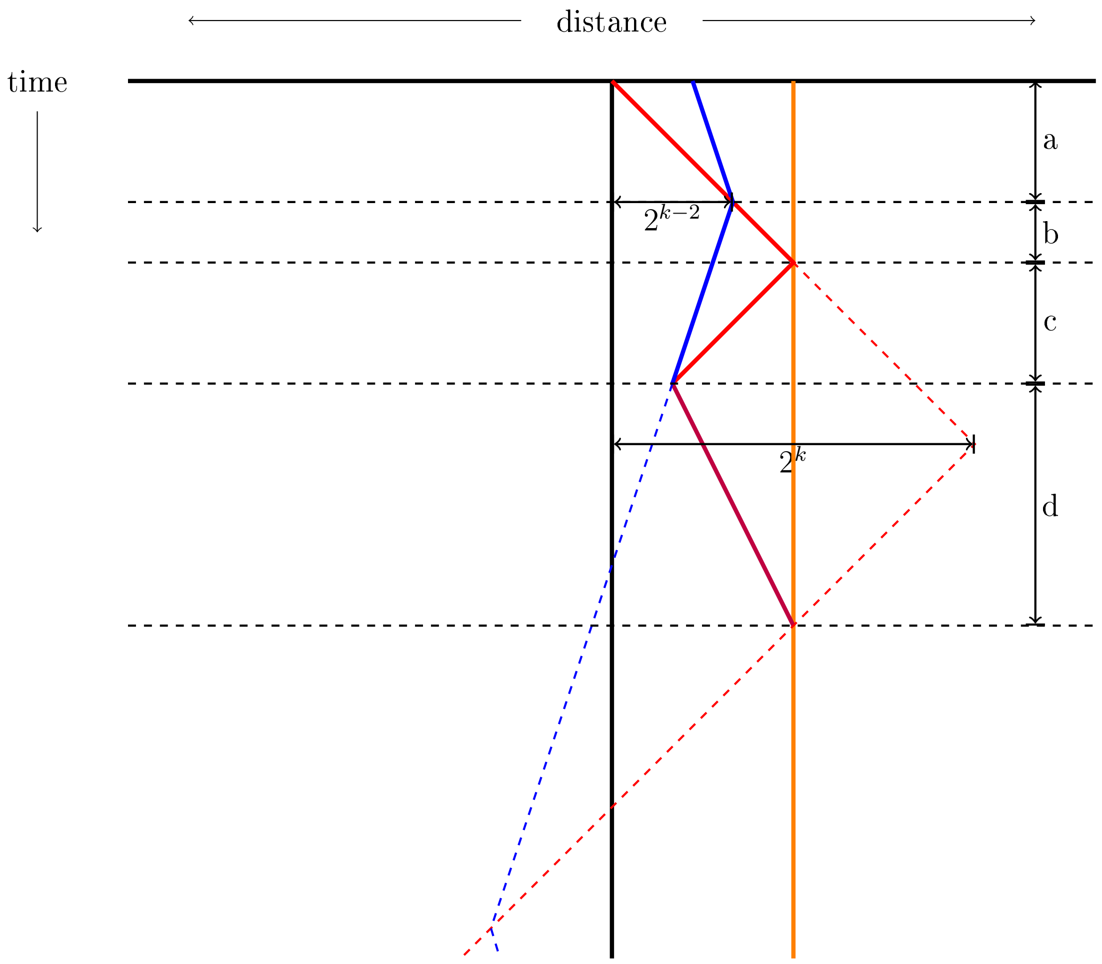


FIGURE 4.4: Example of $9d$ for evacuation problem, where $\mathcal{SMR}$ has $s \geq \frac{1}{3}$ and the evacuation point is at $2^{k-2} + \varepsilon$

4.3.1.3 Still $9d$ evacuation, when $s \geq \frac{1}{3}$

Theorem 4.7. *The coordinated strategy outlined above for the $\mathcal{SMR}$ and $\mathcal{FMR}$ gives a $9d$ upper bound for time of the evacuation problem, as long as $s \geq \frac{1}{3}$.*

Proof. We can think of the evacuation procedure as a three-step process where (1) the $\mathcal{FMR}$ locates the evacuation target, (2) the $\mathcal{FMR}$ informs the $\mathcal{SMR}$ of that location, and (3) the two robots proceed (back) to the target.

We assume that $d \geq 2$. (The $9d$ bound for $d = 1$ is easy to verify.) Note that the $\mathcal{FMR}$ will locate the target between successive "peaks" on the same side of the origin (or it will find the destination just at a "peak"), so we define k to be the integer such that $2^{k-2} < d \leq 2^k$. In particular, we can write $d = 2^{k-2} + \varepsilon$ for some $0 < \varepsilon \leq 3 \cdot 2^{k-2}$.

The "discovery phase" to locate the target will take time (at most)

$$2 \cdot 1 + 2 \cdot 2 + \cdots + 2 \cdot 2^{k-1} + d \;=\; 2\sum_{i=0}^{k-1} 2^i + d \;=\; 2 \cdot (2^k - 1) + d.$$

At the time when the $\mathcal{FMR}$ locates the evacuation target, the distance between the $\mathcal{FMR}$ and $\mathcal{SMR}$ is $\frac{4}{3}\varepsilon$. Why? The two robots crossed paths at the meeting point that is 2^{k-2} away from the origin, and since that meeting the $\mathcal{FMR}$ has moved distance ε and the $\mathcal{SMR}$ has moved a distance of $\frac{\varepsilon}{3}$ (as it is moving at speed $\frac{1}{3}$). After the $\mathcal{FMR}$ locates the target, it immediately reverses direction to inform the $\mathcal{SMR}$. At that time, the two robots are $\frac{4}{3}\varepsilon$ apart and the distance between the pair will decrease at a rate of $\frac{2}{3}$ (the relative speeds between the robots). Therefore, the time for the $\mathcal{FMR}$ to inform the $\mathcal{SMR}$ is $\frac{4}{3}\varepsilon \div \frac{2}{3} = 2\varepsilon$. Note that this also means when the $\mathcal{FMR}$ informs the $\mathcal{SMR}$, they are at distance 2ε from the evacuation target.

Finally, the two robots return to the target to complete the evacuation procedure. Thus, assuming the $\frac{1}{3}$ worst-case speed of the $\mathcal{SMR}$, this final "exit portion" will take time $2\varepsilon \div \frac{1}{3} = 6\varepsilon$.

Therefore, the entire evacuation procedure (in the worst-case, with a $\frac{1}{3}$ speed for the $\mathcal{SMR}$) will take time at most

$$\begin{aligned}
2(2^k - 1) + d + 2\varepsilon + 6\varepsilon &= 2(4 \times 2^{k-2} - 1) + d + 8\varepsilon \\
&= 2\left(4(2^{k-2} + \varepsilon) - 4\varepsilon - 1\right) + d + 8\varepsilon \\
&= 8d - 8\varepsilon - 2 + d + 8\varepsilon \\
&= 9d - 2.
\end{aligned}$$

Figure 4.4 shows in detail the paths taken by the two robots once the *evacuation point* has been found. In the diagram the *evacuation point* is shown as the orange line. The path the $\mathcal{FMR}$ took is in red, the with the originally intended path (prior to finding the *evacuation point*) shown as a dashed red line. The path that the $\mathcal{SMR}$ took is in blue, with its originally intended path shown as a dashed blue line. The purple line signifies where the two robots walked together at $s = \frac{1}{3}$ to the *evacuation point*.

■

We conjecture that when $s < 1/3$, then the evacuation time for the pair is strictly larger than $9d$, i.e. there exists a constant $\delta > 0$ such that the evacuation time is at least $(9+\delta)d - o(d)$.

4.3.2 Two (or more) fast robots, many slow robots

We finish with a remark about evacuating two (or more) fast robots, together with one or more slow robots.

Conjecture 4.8. *Given that the two $\mathcal{FMR}$s have a maximum speed of* 1 *and the slow robots have a speed of at least* $1/5$ *then the whole group of robots can still finish the evacuation in time* $9d$.

With (at least) two $\mathcal{FMR}$s, this pair can perform the coordinated evacuation procedure mentioned in Section 4.2.3. Once a fast robot discovers the evacuation target and proceeds to inform the other $\mathcal{FMR}$, any slow robots that have remained at the origin can be informed as the $\mathcal{FMR}$ passes through the origin. It takes an $\mathcal{FMR}$ time $4d$ to find the target and return to the origin, and another $5d$ time to catch up to the other $\mathcal{FMR}$, inform it, and return to the target. Hence, as long as the slow robots have a speed of at least $1/5$, they will arrive at the evacuation point at the exact same time as the fast pair, hence, the collection of all robots can still finish the evacuation in time $9d$. This can be seen clearly in Figure 4.5

4.4 Conclusion

As stated in the introduction, our main goal in the paper was to initiate study of the evacuation problem using mobile robots having different maximum speeds.

This chapter has demonstrated some cases where the original optimal $9d$ bound for homogeneous mobile robots is still obtainable in this new setting, provided the maximum speed of the slow robot(s) is not too low. Further work is necessary to

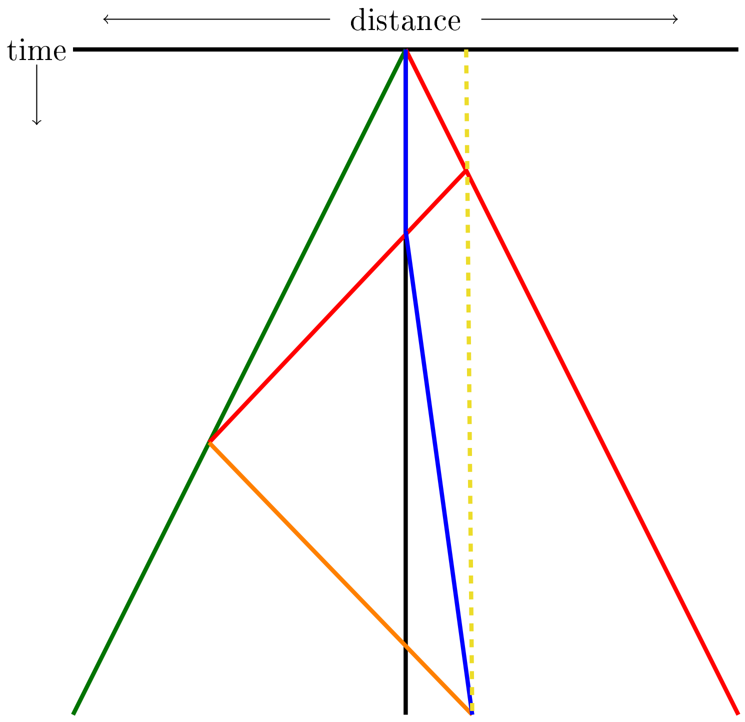


Figure 4.5: A strategy for two robots with unit speed and a third robot with speed at least 1/5 the speed of the fast robot.

investigate these problems, and the related, more general, search and rendezvous problems utilizing robots with different maximum speeds.

Chapter 5

Evacuation Problem on the Disk

This chapter is based on joint work done in [46] that has been published in 28th International Symposium on Distributed Computing (disk'14). The chapter moves on from the previous one by further study of the *evacuation problem*. The work done here extends the concept from the setting of a simple line to one of a disk type arena.

As with Chapter 4 this chapter focuses around the *Evacuation Problem*. As before we will consider k mobile robots but this time the setting is inside a circular disk of unit radius rather than a line. The robots are required to evacuate the disk through an unknown exit point situated on its boundary. We assume all robots having the same (unit) maximal speed and starting at the centre of the disk. The robots may communicate in order to inform themselves about the presence (and its position) or the absence of an exit. The goal is for all the robots to evacuate through the exit in minimum time.

We consider two models of communication between the robots: in *non-wireless* (or *local*) *communication* model robots exchange information only when simultaneously located at the same point, and *wireless communication* in which robots can communicate between each other at any time.

We study the following question for different values of k: what is the optimal evacuation time for k robots? We provide algorithms and show lower bounds in both communication models for $k = 2$ and $k = 3$ thus indicating a difference in evacuation time between the two models. We also obtain almost-tight bounds on the asymptotic relation between evacuation time and team size, for large k.

We show that in the local communication model, a team of k robots can always evacuate in time $3 + \frac{2\pi}{k}$, whereas at least $3 + \frac{2\pi}{k} - O(k^{-2})$ time is sometimes required. In the wireless communication model, time $3 + \frac{\pi}{k} + O(k^{-4/3})$ always suffices to complete evacuation, and at least $3 + \frac{\pi}{k}$ is sometimes required. This shows a clear separation between the local and the wireless communication models.

5.1 Introduction

5.1.1 Overview

Consider a team of mobile robots inside an environment represented by a circular disk of unit radius. The robots need to find an exit being a point at an unknown position on the boundary of the disk in order to evacuate through this point. The exit is recognized when visited by a robot. The robots may communicate in order to exchange the knowledge about the presence (or the absence) of the exit acquired through their previous movements. We consider two communication models. In the *non-wireless* (or *local*) model, communication is possible between robots which arrive at the same point (in the environment) at the same moment, while the *wireless* model allows broadcasting a message by a robot, which is instantly acquired by other robots, independently of their current positions in the environment. The robots start at the centre of the disk and they can move with a speed not exceeding their maximum velocity (which is the same for all robots). The objective is to plan the movements of all robots, which result in the shortest worst-case time needed for all robots to evacuate.

5.1.2 Related work

Mobile robots are autonomous entities traveling within geometric or graph-modeled environments. Besides *mobility*, robots possess the ability to *perceive* the environment, *compute*, and *communicate* among themselves. They collaborate in order to perform tasks assigned to them. When robots operate in geometric environments (then they are usually called robots) their performance is measured by the geometric distance traveled, most often disregarding their computing, communicating and environment-perceiving activities.

When the geometric environment is not known in advance by the mobile robots, in many papers their task consisted in exploring the environment[5, 6, 51, 78]. The coordination of exploration between multiple robots has been mainly studied by the robotics community [31, 118, 122]. However even if the main objective assigned to the robots is different from exploration, often part of their activity is devoted to the recognition or mapping of the terrain and/or the position of the robots within it [85, 97, 102]. When the map of the environment is known to the robots, a lot of research was devoted to search games, when the searchers usually try to minimize the time to find an immobile or a moving hider [8, 12, 90]. The literature of the case of mobile fugitives, often known as cops and robbers or pursuit-evasion games is particularly rich [35, 65], with numerous variations related to the type of environment, speed of evasion and pursuit, robots visibility and many others [99]. The searching for a motionless point target in the simple environment presented in this chapter has some similarities with the lost at sea problem, [74, 79], the cow-path problem [23, 26], and with the plane searching problem [17, 19].

The problem of evacuation has been studied for grid polygons from the perspective of constructing centralized evacuation plans, resulting in the fastest possible evacuation from the rectilinear environment [62]. Previously, [22] considered evacuation planning as earliest-arrival flows with multiple sources giving the first algorithm strongly polynomial in input/output size.

Evacuation in a distributed setting, when the mobile robots (know the simple environment but not the exit positions) has been recently asked in [34], examined in Chapter 4, for the case of a line. We proved that evacuation of multiple uniform robots is as hard as the cow-path problem. Evacuation of two robots without wireless communication was discussed with the research group of M. Yamashita during the visit of the second co-author at Kyushu University [83]. The discussion focused on laying the foundations for the lower bound presented in this chapter and seeking ways to improve the respective upper bound. However, the main objective of our problem is to find a compromise between, on one hand, spreading sufficiently the robots so that they can find the exit point fast in parallel, and, on the other hand, not to spread them too far so that, when one robot finds the exit, the escape route to it of the other robots is not too long.

5.1.3 Preliminaries

The environment is a disk of unit radius. The robots start their movement at the centre of the disk. We assume that the perception device of the robot permits to recognize a boundary point of the environment when the robot arrives there. Similarly, we assume that a robot recognizes the presence of other robots at the same position as well the fact that the robot is currently at the exit point. We also assume that the robots are labeled, i.e. they may execute different algorithms. Each such algorithm instructs the robot to make the moves with a speed not exceeding its maximal speed. In particular, the algorithm may ask the robot to move towards the centre of the disk or a chosen point on its boundary or to follow the boundary clockwise or counterclockwise. The movement may be changed when the perception mechanism allows the robot to acquire some knowledge about the environment (e.g. the exit point, boundary point, a meeting point with another robot). The robots are allowed to stay motionless at the same point. If A and B are points on the perimeter of the disk, by $\widehat{AB}$ we will denote arc from A to B in the clockwise direction and by AB we will denote the cord connecting A and B. The length of $\widehat{AB}$ will be denoted by $|\widehat{AB}|$ and the length of AB will be denoted by $|AB|$.

5.1.4 Outline and results of the paper

In Section 5.2 we consider the evacuation problem for two robots, while Section 5.3 analyzes the case of three robots. Section 5.4 proves tight asymptotic bounds for k robots. Each section is divided into two parts consisting of the analysis for the local communication and wireless models, respectively. The complexity details corresponding to the three sections are in Table 5.1. It shows the results that we have obtained for both the local communication and wireless models for different numbers of robots in the system. We have been able to produce both upper and lower bound results for these and they are presented in the table.

These results establish a clear separation between the local communication and the wireless communication models.

TABLE 5.1: Upper and Lower bounds for $k \geq 2$ robots

The strategy used to obtain the non-wireless result for $k \geq 4$ can be applied successfully to instances where $k < 4$ as well. However, the results given for those scenarios are an improvement over this strategy in those cases.

Model	*Bound*	$k = 2$	$k = 3$	$k \geq 4$
Local	Upper	∼5.74 (Th 5.1)	∼5.09 (Th 5.13)	$3 + \frac{2\pi}{k} < 4.58$ (Th 5.13)
	Lower	∼5.199 (Th 5.2)	∼4.519 (Th 5.10)	$3 + \frac{2\pi}{k} - O(k^{-2})$ (Th 5.15)
Wireless	Upper	∼4.83 (Th 5.5)	∼4.22 (Th 5.11)	$3 + \frac{\pi}{k} + O(k^{-4/3})$ (Th 5.16)
	Lower	∼4.83 (Th 5.8)	∼4.159 (Th 5.12)	$3 + \frac{\pi}{k} > 3.785$ (Th 5.17)

5.2 Two Robots

Consider a disk centered at K. Two robots, say r_1, r_2, start at K moving with constant speed, say 1, searching for an exit located at an unknown point on the perimeter of the disk. In the following we prove upper and lower bounds for the two robot case in the local communication and wireless cases.

5.2.1 Local Communication

Algorithm 6 indicates the robot trajectory for evacuation without wireless communication.

In the following theorem we give a bound on the worst-case evacuation time of Algorithm 6 .

Theorem 5.1. *Algorithm 6 evacuates the robots from an unknown exit located on the perimeter of the disk in time* $1 + \alpha/2 + 3\sin(\alpha/2)$, *where the angle* α *satisfies the equation* $\cos(\alpha/2) = -1/3$. *It follows that the evacuation algorithm takes time* ~ 5.74.

Proof. We calculate the time required until both robots from Algorithm 6 reach the exit. Denote $x = |\widehat{BA}| = |\widehat{AC}|$, $y = |BD| = |\widehat{CD}|$ and $\alpha = |\widehat{BD}|$. According to the definition of the above Algorithm 6 the total time required is $f(\alpha) = 1 + x + 2y$.

Algorithm 6: Algorithm for two robots without wireless communication.

1. Both robots move to an arbitrary point A on the perimeter.
2. At A the robots move along the perimeter of the disk in opposite directions; robot r_1 moves counter-clockwise and robot r_2 moves clockwise until one of the two robots, say r_1, finds the exit at B.
3. Now robot r_1 is at point B and r_2 is at point C (symmetric to B). Robot r_1 chooses a point D such that the length of the chord BD is equal to the length of the arc $\widehat{CD}$ and moves towards D.
4. Since the length of the chord BD is equal to the length of the arc $\widehat{CD}$, both robots arrive at D at the same time. Robot r_1 has knowledge about the location of the exit thus both robots can now follow the straight line DB and exit.

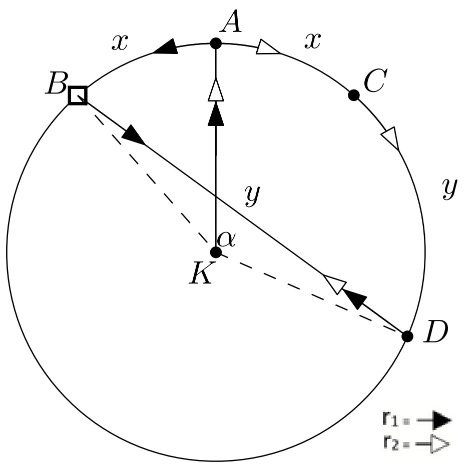


FIGURE 5.1: Evacuation of two robots without wireless communication.

Observe that $\alpha = 2x + y$, and $y = 2\sin(\alpha/2)$, because y is a chord of the angle α. By substituting x and y in the definition of the function f we can express the evacuation time as a function of the angle α as follows. $f(\alpha) = 1 + \frac{\alpha - y}{2} + 2y = 1 + \frac{\alpha}{2} + \frac{3y}{2} = 1 + \frac{\alpha}{2} + 3\sin(\alpha/2)$. Now we differentiate with respect to α and we obtain: $\frac{df(\alpha)}{d\alpha} = \frac{1}{2} + \frac{3}{2}\cos(\alpha/2)$. It is easy to see that this derivative equals 0 for the maximum of function $f(\alpha)$, which yields as value for α the solution of $\cos(\alpha/2) = -1/3$. This proves the theorem. ■

We remark however that Algorithm 6 is not optimal. We can introduce the following modification to Algorithm 6. Let trajectory of a robot be a movement of the robot who neither had discovered the exit nor had been notified about the exit. In Algorithm 6 the trajectory is radius KA and then starting from A, a semicircle (in some direction) of the perimeter. With these new modifications the trajectory of Algorithm 6 becomes:

1. radius KA,
2. part of the semicircle of length z_1 to point E,
3. interval EF of length z_2 towards the centre of the disk,
4. interval FE of length z_2 back to the perimeter,
5. remaining part of the semicircle.

When the robot is moving towards the centre (point 3), the potential length y of the cord that needs to be traversed to get to the exit (if the exit is discovered by the other agent) is shorter than in Algorithm 6. We place the point E such that if the other agent discovered the exit in the worst case point then the agents will meet in the interior of the disk, not on the perimeter. Experiments carried out by one of the co-authors of this work showed that if $z_1 = 2.64$ and $z_2 = 0.5$ then the worst case evacuation time of the modified algorithm is 5.64, [104].

In the following we state and prove a lower bound.

Theorem 5.2. *It takes at least* $3 + \frac{\pi}{4} + \sqrt{2}$ *(*~ 5.199*) time units for two robots to evacuate from an unknown exit located in the perimeter of the disk.*

Proof. At the beginning, both robots are located at the centre K of the disk. It takes at least 1 time unit for both of them to move to the perimeter of the disk.

In less than an additional $\pi/4$ time units the two robots cover at most a length of $\pi/2$ of the perimeter. The main idea is to observe, that until that time of the movement we can always construct a square $ABCD$ with sides equal to $\sqrt{2}$ whose all vertices are not yet visited by neither of the two robots. The vertices represent positions where an adversary can place an exit. Using an adversary argument it can be shown that an additional $2 + \sqrt{2}$ time units are required for robot evacuation. We give details of this argument in the following two lemmas.

Lemma 5.3. *For any* $\varepsilon > 0$*, at time* $1 + \frac{\pi}{4} - \varepsilon$ *there exists a square inscribed in the disk none of whose vertices has been explored by a robot.*

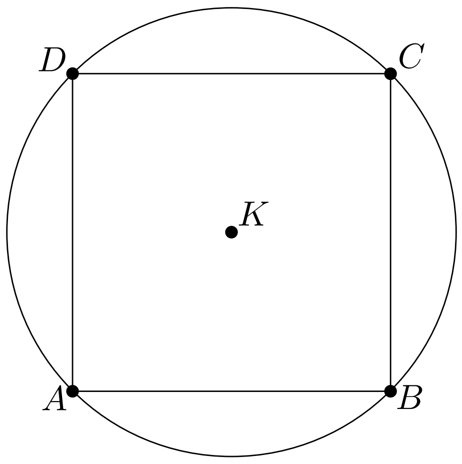


FIGURE 5.2: Forming a square $ABCD$ of positions not yet explored by the robots.

Proof. (Lemma 5.3) The proof is easily derived by rotating a square inscribed in the disk continuously for an angle of $\pi/2$. More precisely assume on the contrary that such an inscribed square does not exist. Consider a partition of perimeter of the disk into four arcs of length $\pi/2$, E_1, E_2, E_3, E_4. Any point $e_1 \in E_1$ uniquely defines an inscribed square with vertices $e_1 \in E_1, e_2 \in E_2, e_3 \in E_3, e_4 \in E_4$. Moreover for a different $e'_1 \in E_1$, $e'_1 \neq e_1$ vertices of the inscribed square $\{e'_1, e'_2, e'_3, e'_4\}$ are different $e'_i \neq e_i$ for all $i \in 1, 2, 3, 4$. By the assumption, for any $e_1 \in E_1$ at least one of the vertices $\{e_1, e_2, e_3, e_4\}$ of the inscribed square has to be explored (denote it by e^*). Thus for any e_1 we can identify an explored vertex $e^*(e_1)$. Since for different e_1, the inscribed square is different then the function $e^*(e_1)$ is an injection. Thus the image of the function $e^*(e_1)$ is a set of length $\pi/2$ of explored points. But such set does not exist because at time $1 + \pi/4 - \varepsilon$ the total length of the set of explored points less than $\pi/2$. Therefore we obtain a contradiction at time $1 + \frac{\pi}{4} - \varepsilon$ that an inscribed square, none of whose vertices has been explored by a robot, does exist. ■

Lemma 5.4. *For any square inscribed in the disk none of whose vertices has been explored by a robot it takes more than $2 + \sqrt{2}$ time to evacuate both robots from a vertex of the square.*

Proof. (Lemma 5.4) Take the square $ABCD$ with unexplored vertices. Consider any evacuation algorithm $\mathcal{A}$. We allow the algorithm to place the robots on arbitrary positions of the disk (possibly also on vertices of the square). The adversary can run the algorithm with undefined position of the exit and place the exit depending on the behaviour of the robots. The adversary will run the algorithm from perspective of a fixed robot r and will place the exit at a some point P. The placement of the exit at point P in time t is possible if robot r has no information whether the exit is located in P. Formally we say that a point P is *unknown* to robot r at time t if for any time moment $t' \in [0, t]$ robot r is at distance more than t' from P. This means that even if other robot started at P it could not meet r at any time in the interval $[0, t]$. Take a robot r and the first time moment t when the third vertex of the square is visited by a robot. Consider two cases

Case 1. $\sqrt{2} \leq t < 2$.

Denote the vertex visited by r in time t by A. The adversary places the exit in the antipodal point C. Observe that point C is unknown to r at time t. This is because if r was at distance at most t' from C at some time $t' \in [0, t]$ then it would be at distance $2 - t'$ from A and would reach A no sooner than at time 2, which is a contradiction as $t < 2$. Thus placement of the exit in C cannot affect movement of r until time t. Therefore, the adversary can place the exit in C and the evacuation time in this case will be at least $t + 2 \geq 2 + \sqrt{2}$.

Case 2. $2 \leq t$.

Time moment t is the first time when three vertices of the square are explored (it is possible that in t both robots explore a new vertex). Therefore, at time t, some robot r has knowledge about at most three vertices. The adversary simply places the exit in the vertex unknown to r and the evacuation time of r will be at least $t + \sqrt{2} \geq 2 + \sqrt{2}$.

Observe that t cannot be smaller than $\sqrt{2}$ because within time t at least one robot has to traverse at least one side of the square. This proves the lemma. ■

Clearly, the proof of Theorem 5.2 is an immediate consequence of Lemmas 5.3 and 5.4. ■

5.2.2 Wireless communication

Algorithm 7 indicates the robot trajectory for evacuation with wireless communication.

Algorithm 7: Algorithm for two robots with wireless communication.

1. Both robots move to an arbitrary point A on the perimeter.
2. At A the robots start moving along the perimeter of the disk in opposite directions: robot r_1 moves counter-clockwise and robot r_2 moves clockwise until one of the robots, say r_1, finds the exit at B.
3. Robot r_1 notifies r_2 using wireless communication about the location of the exit and robot r_2 takes the shortest chord to B.

Theorem 5.5. *Algorithm 7 is an algorithm for evacuating two robots from an unknown exit located on the perimeter of the disk which takes time at most* $1 + \frac{2\pi}{3} + \sqrt{3}$.

Proof. Consider the maximum evacuation time of Algorithm 7. If the angular distance between A and B equals x, then the length of the chord taken by the robot r_2 equals to $c(x) = 2\sin(\pi - x)$ (see Figure 5.3). Thus the evacuation time

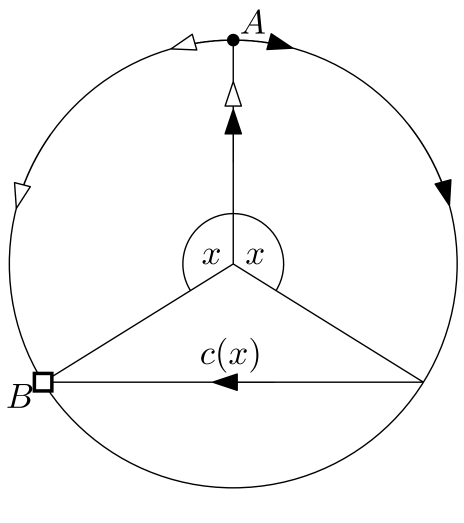


FIGURE 5.3: Evacuation of two robots with wireless communication.

T satisfies $T \leq \max_{0\leq x\leq\pi}\{1 + x + 2\sin(\pi - x)\} = \max_{0\leq x\leq\pi}\{1 + x + 2\sin x\}$. The function $f(x) = 1 + x + 2\sin x$ in the interval $[0, \pi]$ is maximized at the point $x^* = 2\pi/3$ and $f(x^*) = 1 + 2\pi/3 + \sqrt{3}$. This proves the theorem. ■

In order to prove the lower bound we need to show the following lemma.

Lemma 5.6. *Consider a perimeter of a disk whose subset of total length* $u+\varepsilon > 0$ *has not been explored for some* $\varepsilon > 0$ *and* $\pi \geq u > 0$. *Then there exist two unexplored points between which the distance on the perimeter is at least* u.

Proof. Denote by U the set of all unexplored points. We have that $|U| = u + \varepsilon$. First consider case when $u < \pi$. Throughout the proof we will consider only points on the perimeter of the disk. Let $dist(x_1, x_2)$ be defined as the length of the shorter arc connecting x_1 and x_2.

Assumption 1 *Assume on the contrary that two unexplored points between which the distance on the perimeter is at least u do not exist.*

Under such assumption we will construct subsets N, L, R of the set of all unexplored points ($N, L, R \subset U^c$). Set N is defined as the set of all antipodal points of points in U, (if $x \in U$, then $y \in N$ if and only if $dist(x, y) = \pi$). For any $x \in U$, by $x + \pi$ we denote the point antipodal to x. To construct L and R take any $x \in U$. Let x' and x'' be the unexplored point closest to $x + \pi$ in the clockwise and counter-clockwise direction respectively. We construct arc L as the set of points on the perimeter at distance at most $\pi - u$ from $x' + \pi$ (antipodal to x') in the counter-clockwise direction. Analogically R is the set of points at distance at most $\pi - u$ from $x'' + \pi$ in the clockwise direction (see Figure 5.4).

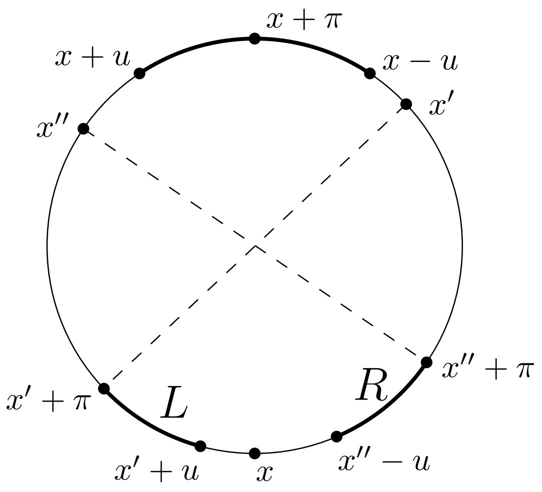


FIGURE 5.4: Construction of sets L and R.

Observe that all points belonging to sets N, L, R are explored. Every point $y \in N$ is antipodal to some unexplored point $y + \pi \in U$, thus if y is unexplored then we have a pair of unexplored points y, $y + \pi$ at distance π. If a point y' in L is unexplored then we have a pair of unexplored points x', y' at distance at least u. Finally if a point y'' in R is unexplored then we have a pair of unexplored points x'', y'' at distance at least u. All these cases lead do contradiction with Assumption 1.

We want to show that $|L \cup R| = 0$. First note that

$$dist(x+\pi, x') > \pi - u, \tag{5.1}$$

because if $dist(x+\pi, x') \leq \pi - u$, then $dist(x, x') \geq u$ which is impossible due to Assumption 1 since both x and x' are unexplored. Similarly we observe that

$$dist(x+\pi, x'') > \pi - u. \tag{5.2}$$

By equation 5.1 we have that $dist(x'+\pi, x) = dist(x', x+\pi) > \pi - u$ thus set L is a subset of the semicircle from x to $x+\pi$ in the clockwise direction. Similarly we show that R is a subset of the semicircle from x to $x+\pi$ in the counter-clockwise direction. Thus $L \cup R$ contains at most one point (in the case when $x = x' = x''$). Thus $|L \cup R| = 0$.

Observe also that $|L \cup N| = 0$, because all points in the arc from $x+\pi$ to x' in the clockwise direction are explored (x' is the closest unexplored). Similarly $|R \cup N| = 0$.

Thus $|N \cup L \cup R| = |N| + |L| + |R| = u + \pi - u + \pi - u = 2\pi - u$. Since all points in N, L, R are explored we have

$$|U| = 2\pi - |U^c| \leq 2\pi - |N \cup L \cup R| = u$$

which is a contradiction because $|U| > u$. If $u = \pi$ it is sufficient to consider set N. Observe that all elements from set N are explored and $|N| = \pi$. We obtain contradiction because $|U| > \pi$. ■

Lemma 5.7. *For any $k \geq 2$ and x satisfying $\pi/k \leq x < 2\pi/k$ and any evacuation algorithm it takes time at least $1 + x + 2\sin(xk/2)$ to evacuate from an unknown exit located in the perimeter of the disk.*

Proof. Consider an algorithm $\mathcal{A}$ whose evacuation time equals to T. In any evacuation algorithm using k robots, at time moment $1+x$, the total length of explored arcs of the perimeter equals at most xk (because robots need time 1 to go from the centre to the perimeter). Thus the total length of the unexplored part of the perimeter is at least $2\pi - xk$ and $\pi \geq 2\pi - xk > 0$. Thus using Lemma 5.6 at time moment $1+x$ there exists a pair of unexplored points whose distance on the perimeter is at least $2\pi - xk - \varepsilon$ for any $\varepsilon > 0$. Take this pair of points and consider a chord connecting them. Such chord has length at least $2\sin(\pi - xk/2 - \varepsilon/2)$ and has both endpoints unexplored. Thus the adversary can place the exit in any of two endpoints. If a robot visits one endpoint of the chord, the adversary places the exit in the other and such agent will have to walk at least the length of the chord more. Thus the total evacuation time is at least $1 + x + 2\sin(\pi - xk/2 - \varepsilon/2)$. This holds for any $\varepsilon > 0$, thus by taking the limit $\varepsilon \to 0$ we obtain $T \geq 1 + x + 2\sin(xk/2)$. ■

Theorem 5.8. *For any algorithm it takes at least* $1 + \frac{2\pi}{3} + \sqrt{3}$ *time in the worst case for two robots to evacuate from an unknown exit located in the perimeter of the disk.*

Proof. The theorem is a direct consequence of Lemma 5.7 by taking $k = 2$ and $x = 2\pi/3$. ■

5.3 Three Robots

In this section we analyze evacuation time for three robots in both local communication and wireless models.

5.3.1 Local Communication

The first lemma provides a lower bound which is applicable for any k robots in the local communication model.

Lemma 5.9. *For any* $k \geq 3$ *and* $1 < \alpha < 2$, *it takes in the worst case at least* $\min\left\{3 + \frac{\alpha\pi}{k}, 3 + 2\sin\left(\pi - \frac{\alpha\pi}{2}\right)\right\}$ *time to evacuate from an unknown exit located on the perimeter of the disk in the model without wireless communication.*

Proof. (Lemma 5.9) Take any evacuation algorithm $\mathcal{A}$. Denote by $\mathcal{A}_r^p(t)$ the position of robot r in time t if the exit is located at point p. Since we are considering the worst case, we need to show that there exists a point p^* on the perimeter such that if the exit is located at p^* then the evacuation time of the algorithm $\mathcal{A}$ is at least $3 + \frac{2\pi}{k} - O(k^{-2})$. Consider the following three time intervals: $I_1 = [0, 1)\,, I_2 = \left[1, 1 + \frac{\alpha\pi}{k}\right), I_3 = \left[1 + \frac{\alpha\pi}{k}, 3\right)$. Since algorithm $\mathcal{A}$ is deterministic, the robots will follow a fixed trajectory, independent of the location of the exit until finding the exit or being notified about it by some other robot. Denote these trajectories by $p_1(t), p_2(t), \dots p_k(t)$. Consider two cases:

Case 1. *There exists a robot r and time $t^* \in I_3$ such that point $p = p_r(t^*)$ of the trajectory of the robot r is on the perimeter of the disk.*

We will argue that the adversary can place the exit at point p^* being antipodal of p. We need to prove that if the exit is at point p^* then until time t^* robot r will be unaware of the location of the exit and will follow the trajectory $p_r(t)$. Consider the trajectory followed by robot r in algorithm $\mathcal{A}$ if the exit is at point p^*. Robot r is following the trajectory $p_r(t)$ until finding the exit or being notified about it. We want to show that robot r cannot be notified about the exit until time t^*. Assume on the contrary that $1 \le t' < t^*$ is the first moment in time when r either discovered the exit or met a robot carrying information about the location of the exit. Thus we have that $\mathcal{A}_r^{p^*}(t) = p_r(t)$, for all $t \in [0, t']$. First note that since $p = p_r(t^*)$ we have that $dist(\mathcal{A}_r^{p^*}(t'), p^*) = dist(p_r(t'), p^*) > t' - 1$. The last inequality is true because if the distance between $p_r(t')$ and p^* would be at most $t' - 1$ then the distance to p would be at least $3 - t'$ (because p and p^* are antipodal) and robot r following trajectory $p_r(t)$ would not be able to reach p until time t^* (recall $t^* < 3$), which is a contradiction since $p_r(t^*) = p$. Now observe that in algorithm $\mathcal{A}$ if the exit is located at p^* then for any time moment $t' \le 3$, any robot carrying information about the location of the exit is at distance at most $t' - 1$ from p^* (it is because robots can exchange informations only when they meet and the maximum speed of a robot is 1). Thus it is not possible that robot r in time t' obtain the information about the exit by meeting another robot. It is also not possible that $p_r(t') = p^*$, because robot r following trajectory $p_r(t)$ would not be able to reach p until time t^*. Thus such t' does not exist and we have: $\mathcal{A}_r^{p^*}(t) = p_r(t), \quad$ for all $t \in [0, t^*]$. In time moment t^* robot r following algorithm $\mathcal{A}$ is at distance 2 from the exit located at p^*. Thus the total evacuation time is at least $t^* + 2 \ge 3 + \alpha\pi/k$, since $t^* \ge 1 + \alpha\pi/k$ (because $t^* \in I_3$).

Case 2. *None of the trajectories $p_1(t), p_2(t), \dots p_k(t)$ in the interval I_3 is equal to a point on the perimeter.*

In this case we consider robots following the trajectories $p_1(t), p_2(t), \dots, p_k(t)$ in the time interval $[0, 3)$. We need Lemma 5.6.

The set of points U on the perimeter of the disk that were not visited by any robot following such trajectories satisfies $|U| \ge 2\pi - \alpha\pi$ because in this case robots can explore the perimeter only in time interval I_2 of length $\alpha\pi/k$. Thus by Lemma 5.6 there exists a pair of unexplored points at distance at least $2\pi-\alpha\pi-\varepsilon$ for any $\varepsilon > 0$. The chord connecting these two points has length at least $2\sin(\pi - \alpha\pi/2 - \varepsilon/2)$. Take this chord and denote its endpoints by u_1 and u_2. The adversary can run the algorithm $\mathcal{A}$ until moment t' when one of the points u_1, u_2 is visited and the adversary can place the exit in the other one. Note that until moment t' robots are following trajectories $p_r(t)$ because none of the robots has any information about the exit, thus $t' \geq 3$. Now the first robot that visited one of the points u_1, u_2 still needs to travel at least $2\sin(\pi-\alpha\pi/2-\varepsilon/2)$ because the exit is on the other end of the chord. Thus exploration time is in this case at least $3+2\sin(\pi - \alpha\pi/2 - \varepsilon/2)$. We showed that the worst case time of evacuation T for any correct algorithm satisfies $T \geq \min\left\{3 + \frac{\alpha\pi}{k}, 3 + 2\sin\left(\pi - \frac{\alpha\pi}{2} - \frac{\varepsilon}{2}\right)\right\}$, for any $\varepsilon > 0$. The claim of the lemma follows by passing to the limit as $\varepsilon \to 0$. ■

Theorem 5.10. *It takes at least* 4.519 *time in the worst case to evacuate three robots from an unknown exit located in the perimeter of the disk in the model without wireless communication.*

Proof. We have by Lemma 5.9 that the evacuation time T of any evacuation algorithm $\mathcal{A}$ satisfies $T \geq \min\{3 + \frac{\alpha\pi}{k}, 3 + 2\sin(\pi - \alpha\pi/2)\}$ for any $k \geq 3$. To prove the statement we numerically find such α that $\frac{\alpha\pi}{3} = 2\sin\left(\pi - \frac{\alpha\pi}{2}\right)$. If we set $\alpha = 1.408$, we obtain $T \geq \min\left\{3 + \frac{\alpha\pi}{3}, 3 + 2\sin\left(\pi - \frac{\alpha\pi}{2}\right)\right\} > 4.519$. This proves the theorem. ■

5.3.2 Wireless communication

We have three robots r_1, r_2, r_3 and consider Algorithm 8.

The upper bound is proved in the following theorem.

Theorem 5.11. *It is possible to evacuate three robots from an unknown exit located on the perimeter of the disk in time at most* $\frac{4\pi}{9} + \frac{2\sqrt{3}+5}{3} + \frac{1}{600} < 4.22$ *in the model with wireless communication.*

Proof. Consider the evacuation time of Algorithm 8. If the exit is discovered within time $1+y$, then since we are working in the wireless communication model, time for evacuation is at most 2 after the discovery as the furthest away a robot

Algorithm 8: Algorithm for three robots with wireless communication.

1. Robot r_1 moves to an arbitrary point A of the perimeter, robots r_2 and r_3 move together to the point B at angle $y = 4\pi/9 + 2\sqrt{3}/3 - 401/300$ in the clockwise direction to the radius taken by robot r_1.
2. Robot r_1 moves in the counter-clockwise direction. Robot r_2 moves in the clockwise direction. Robot r_3 moves in the counter-clockwise direction for time y. Then r_3 moves towards the centre. Then r_3 moves towards the perimeter at angle $\pi - y/2$ in the clockwise direction to radius RB.
3. A robot that discovers the exit sends notification to other robots.
4. Upon receiving notification a robot walks to the exit using the shortest path.

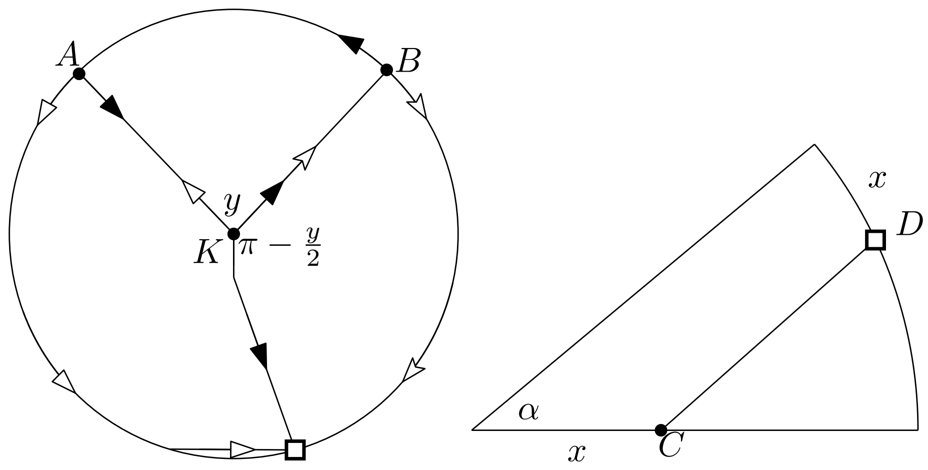


FIGURE 5.5: Evacuation of three robots with wireless communication. Evacuation of three robots with wireless communication. $|CD| = \sqrt{1 - 2x\cos(\alpha - x) + x^2}$

could be from that point at this time would be the diameter of the disk. Thus if the discovery is within time $1 + y$, the evacuation is in time at most $3 + y$. If the exit is discovered after time $1 + y$ then it is discovered either by r_1 or r_2 (robot r_3 explores part of the perimeter of length y thus he finishes exploration in time $1 + y$).

Consider the evacuation time of r_1 if the exit is discovered by r_2. Robot r_1 explores an assigned arc until being notified and upon notification he takes the chord to the exit. If the exit is discovered at time $1 + y'$ then the evacuation time of r_1 is

$$T_{r_1} = 1 + y' + 2\sin(\pi - y/2 - y'),$$

and $y' \in [0, \pi - y/2]$. In this interval the function $f(y') = 1 + y' + 2\sin(\pi - y/2 - y')$ is maximized when $y' = 2\pi/3 - y/2$ and the maximum value is $1 + 2\pi/3 - y/2 + \sqrt{3}$. Thus we have

$$T_{r_1} \leq 1 + \frac{2\pi}{3} - \frac{y}{2} + \sqrt{3} = \frac{4\pi}{9} + \frac{2\sqrt{3}+5}{3} + \frac{1}{600}$$

The evacuation time of r_2 can be bounded similarly.

Consider the evacuation time of r_3. Consider the case when the exit is discovered by r_1 or r_2 at some point of time in the interval $[1+y, 2+y]$. In this interval, robot r_3 is moving towards the centre. A path from the point A to the centre and from the centre to the exit has length 2 (two times the radius). A path taken by the robot is shorter by the triangle inequality, because the robot after the diskovery of the exit is not continuing to the centre but goes to the exit using the shortest path. Thus if the exit is discovered within interval $[1+y, 2+y]$ then the evacuation time of r_3 is at most

$$T_{r_3} \leq 3 + y = \frac{4\pi}{9} + \frac{2\sqrt{3}}{3} + \frac{998}{600} < \frac{4\pi}{9} + \frac{2\sqrt{3}+5}{3} + \frac{1}{600}.$$

Finally consider the evacuation time of robot r_3 in the case when the exit is discovered after time $2 + y$. In this case the exit is discovered while robot r_3 is walking from the centre towards the perimeter. If the exit is discovered at time $2 + y + x$ then robot r_3 walked distance x from the centre. The length of the segment he takes to the exit equals $\sqrt{1 - 2x\cos(\alpha - x) + x^2}$ (see Figure 5.3), where α is length of the arc traversed by r_2 (or equivalently by r_1) after time $2+y$. At time $2+y$ the total length of the explored perimeter equals $3y+2$. Thus $\alpha = \pi - 3y/2 - 1$. Thus the evacuation time of r_3 is in this case at most

$$T_{r_3} \leq 2 + y + \max_{x\in[0,\alpha]}\{x + \sqrt{1 - 2x\cos(\alpha - x) + x^2}\}$$

We have that $\alpha = \pi - 3y/2 - 1 = \pi/2 - \sqrt{3} + 201/200 < 1/3$. In the interval $[0, 1/3]$ the cos function is decreasing thus $-2x\cos(1/3 - x) \geq -2x\cos(\alpha - x)$ for any $x \in [0, 1/3]$, thus we have

$$\begin{aligned}\max_{x\in[0,\alpha]}\left\{x + \sqrt{1 - 2x\cos(\alpha - x) + x^2}\right\} &\leq \max_{x\in[0,1/3]}\left\{x + \sqrt{1 - 2x\cos(\alpha - x) + x^2}\right\}\\ &\leq \max_{x\in[0,1/3]}\left\{x + \sqrt{1 - 2x\cos(1/3 - x) + x^2}\right\}\end{aligned}$$

To complete the proof we show the following

Claim: $x + \sqrt{1 - 2x\cos(1/3 - x) + x^2} \leq 1.005$ *for every* $x \in [0, 1/3]$

First we change the variable $z = 1/3 - x$. We have

$$\max_{x\in[0,1/3]} \left\{ x + \sqrt{1 - 2x\cos\left(\frac{1}{3} - x\right) + x^2} \right\} \max_{z\in[0,1/3]} \left\{ \frac{1}{3} - z + \sqrt{1 - 2\left(\frac{1}{3} - z\right)\cos z + \left(\frac{1}{3} - z\right)^2} \right\}$$

Now using Lemma 5.14 we have

$$\begin{aligned} \frac{1}{3} - z + \sqrt{1 - 2\left(\frac{1}{3} - z\right)\cos z + \left(\frac{1}{3} - z\right)^2} &\leq \frac{1}{3} - z + \sqrt{1 - 2\left(\frac{1}{3} - z\right)\left(1 - \frac{z^2}{2}\right) + \left(\frac{1}{3} - z\right)^2} \\ &= \frac{1}{3} - z + \frac{2}{3}\sqrt{1 + 3z + 3z^2 - \frac{9}{4}z^3} \end{aligned}$$

In order to prove that $\frac{1}{3} - z + \frac{2}{3}\sqrt{1 + 3z + 3z^2 - \frac{9}{4}z^3} \leq 1.005$ it is equivalent to show that

$$1 + 3z + 3z^2 - \frac{9}{4}z^3 \leq \frac{162409}{160000} + \frac{1209}{400}z + \frac{9}{4}z^2,$$

because for $z \in [0, 1/3]$, $1 + 3z + 3z^2 - \frac{9}{4}z^3 > 0$. Thus we need to show that

$$0 \leq z^3 - \frac{z^2}{3} + \frac{z}{100} + \frac{1}{150} + \frac{1}{40000}$$

The polynomial $z^3 - \frac{z^2}{3} + \frac{z}{100} + \frac{1}{150} + \frac{1}{40000}$ in the interval $[0, 1/3]$ has the minimal value for $z = 1/9 + \sqrt{73}/90$, and the minimal value is bigger than 0. This finishes the proof of the claim.

Using the claim we have that in the case when the exit is discovered after time $2 + y$, the evacuation time T_{r_3} of robot r_3 satisfies

$$T_{r_3} \leq 2 + y + 1.005 = \frac{4\pi}{9} + \frac{2\sqrt{3} + 5}{3} + \frac{1}{600}.$$

We bounded the evacuation time of each robot in every possible position of the exit thus the evacuation time T of the algorithm satisfies

$$T \leq \frac{4\pi}{9} + \frac{2\sqrt{3} + 5}{3} + \frac{1}{600}.$$

■

The lower bound is proved in the following theorem.

Theorem 5.12. *For any algorithm it takes at least* $1+\frac{2}{3}\arccos\left(-\frac{1}{3}\right)+\frac{4\sqrt{2}}{3} \sim 4.159$ *time in the worst case for three robots to evacuate from an unknown exit located in the perimeter of the disk.*

Proof. The theorem is a direct consequence of Lemma 5.7 by taking $k = 3$ and $x = 2/3\arccos(-1/3)$. ∎

5.4 k Robots

We prove asymptotically tight bounds for k robots in both the local communication and wireless models.

5.4.1 Local Communication

The trajectory of the robots is given in Algorithm 9.

Algorithm 9: Algorithm for k robots without wireless communication.

```
1. The $k$ robots “spread” at equal angles $2\pi/k$ and they all reach the
   perimeter of the disk in time 1.
2. Upon reaching the perimeter, they all move clockwise along the perimeter
   for $2\pi/k$ time units.
3. In one time unit, all robots move to the centre of the disk. Since at least
   one robot has found the exit it can inform the remaining robots.
4. In one additional time unit all robots move to the exit.
```

Theorem 5.13. *It is possible to evacuate k robots from an unknown exit located on the perimeter of the disk in time* $3+\frac{2\pi}{k}$ *in the model with local communication.*

Proof. Clearly Algorithm 9 is correct and attains the desired upper bound. ∎

The following technical lemma provides bounds on the sin and cos functions based on their corresponding Taylor series expansions, [1].

Lemma 5.14. *For any* $x \geq 0$ *the following bound on values of* $\sin x$ *and* $\cos x$ *hold:*

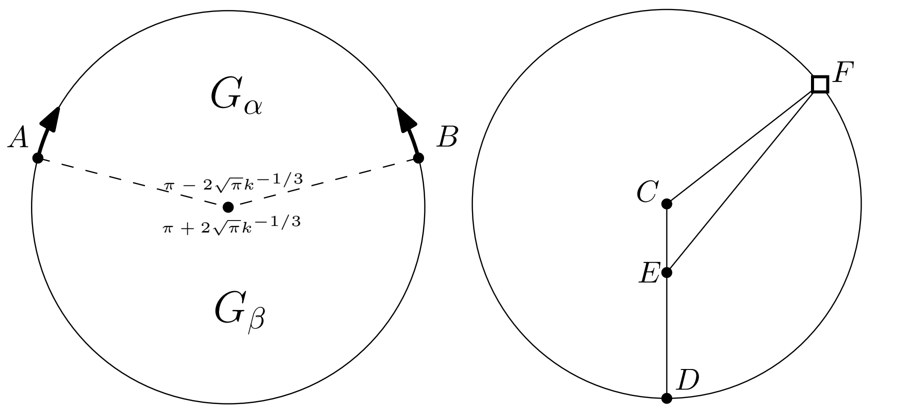


FIGURE 5.6: k robots with wireless communication
Extremal (leftmost and rightmost) robots from group G_α are moving towards the interior of the arc $\widehat{AB}$.
$|DE| + |EF| < |DC| + |CF|$

(1) $\sin x \geq x - x^3/3!$

(2) $\cos x \leq 1 - x^2/2! + x^4/4!$

Theorem 5.15. *It takes time at least* $3+\frac{2\pi}{k}+O(k^{-2})$ *in the worst case to evacuate* k *robots from an unknown exit located on the perimeter of the disk in the model without wireless communication.*

Proof. We have by Lemma 5.9 that the evacuation time T of any evacuation algorithm $\mathcal{A}$ satisfies $T \geq \min\{3+\frac{\alpha\pi}{k}, 3+2\sin(\pi-\alpha\pi/2)\}$. If we set $\alpha = 2k/(k+1)$ then taking into account Lemma 5.14 we obtain:

$$T \geq \min\left\{3 + \frac{\pi}{k+1}, 3 + 2\sin\left(\frac{\pi}{k+1}\right)\right\} \geq 3 + \frac{\pi}{k+1} - \frac{\pi^3}{3!(k+1)^3}$$
$$= 3 + \frac{\pi}{k} - \frac{\pi}{k(k+1)} - \frac{\pi^3}{3!(k+1)^3} = 3 + \frac{\pi}{k} - O(k^{-2}),$$

This proves the theorem. ∎

For $k \geq 3$ robots we conjecture that the time T required to find an exit on the perimeter of a disk is exactly $3 + \frac{2\pi}{k}$.

5.4.2 Wireless communication

The trajectory of the robots is given in Algorithm 10.

Algorithm 10: Algorithm for k robots with wireless communication.

```
1. Divide the team of robots into two groups: Group $G_\alpha$ of size $k_\alpha = \lceil k^{2/3} \rceil$,
   and Group $G_\beta$ of size $k_\beta = k - k_\alpha$.
2. Assign a continuous arc $\widehat{AB}$ of length $\pi - 2\sqrt{\pi}k^{-1/3}$ to group $G_\alpha$ and
   remaining part of the perimeter denoted by $\widehat{BA}$ (of length $\pi + 2\sqrt{\pi}k^{-1/3}$)
   to group $G_\beta$.
3. Divide arcs $\widehat{AB}$ and $\widehat{BA}$ equally between members of groups. Each robot
   belonging to $G_\alpha$ is assigned an arc of length $a_\alpha = |\widehat{AB}|/k_\alpha$. Each robot
   from group $G_\beta$ receives an arc of length $a_\beta = |\widehat{BA}|/k_\beta$.
4. Each robot goes from the centre to the perimeter and explores an assigned
   arc. Extremal robots from group $G_\alpha$ when exploring the assigned arcs go
   towards each other (see Figure 5.6). All other robots explore assigned arcs
   is any direction. A robot that discovers the exit sends notification to all
   other robots using wireless communication.
5. Upon receiving a notification about the position of the discovered exit, a
   robot takes the shortest chord to the exit.
6. Robots from group $G_\beta$ after finishing exploration of their arcs start moving
   towards the centre.
```

Theorem 5.16. *Using Algorithm 10 with an input of* $k \geq 100$ *then it is possible to evacuate* k *robots from an unknown exit located in the perimeter of the disk in time* $3 + \frac{\pi}{k} + O(k^{-4/3})$, *in the model with wireless communication.*

Proof. Consider the evacuation time of the Algorithm 10. Note that since $k \geq 100$ then $k - \lceil k^{2/3} \rceil \geq \lceil k^{2/3} \rceil$ implying that $a_\alpha > a_\beta$. Thus robots from G_β finish exploration first and start going towards the centre while robots from G_α are still exploring (point 6. in the pseudocode). We will show an upper bound on evacuation time T of the algorithm. Consider two cases:

Case 1. *The exit is located within the arc* $\widehat{AB}$.

Consider the evacuation time T_β of robots from group G_β. Observe that since $\varepsilon > 1$, then $a_\alpha < 1$ thus the exit is discovered while robots from G_β are walking towards the centre (before they reach the centre). Robots from G_β start moving towards the centre at time $1 + a_\beta$. At some time t' satisfying $2 + a_\beta > t' > 1 + a_\beta$ the exit is discovered by a robot from group G_α. Consider a trajectory taken by a robot r from group G_β starting from time $1 + a_\beta$. If r would simply walk to the centre and then from the centre to the exit (location of the exit would be known by the time when r reaches the centre). The time would be not more than $3 + a_\beta$. By the triangle inequality the path taken by robot r acting according to the algorithm is shorter (see Figure 5.6). Thus the evacuation time T_β for robots belonging to team G_β is at most

$$\begin{aligned} T_\beta &\leq 3 + a_\beta \leq 3 + \frac{\pi + 2\sqrt{\pi}k^{-1/3}}{k - k_\alpha} \\ &= 3 + \frac{\pi + 2\sqrt{\pi}k^{-1/3}}{k} + \frac{(\pi + 2\sqrt{\pi}k^{-1/3})\lceil k^{2/3}\rceil}{k(k - \lceil k^{2/3}\rceil)} = 3 + \frac{\pi}{k} + O(k^{-4/3}). \end{aligned}$$

Consider now the evacuation time of robots from group G_α. Assume that the exit is discovered at time $1+x$ for some $0 \leq x \leq a_\alpha$. Since the extremal robots from group G_α are walking towards each other at the time moment $1 + x$ two arcs of length x has been explored starting from each endpoint of arc $\widehat{AB}$. Thus the distance on the perimeter between extremal unexplored points of arc $\widehat{AB}$ is $\pi - 2\sqrt{\pi}k^{-1/3} - 2x$. Hence the maximum length of a chord connecting two unexplored points of arc $\widehat{AB}$ in this moment is $2\sin((\pi - 2\sqrt{\pi}k^{-1/3} - 2x)/2)$. Therefore the time T_α until evacuation of all robots from group G_α is at most

$$\begin{aligned} T_\alpha &\leq \max_{0 \leq x \leq a_\alpha} \left\{ 1 + x + 2\sin\left(\frac{\pi - 2\sqrt{\pi}k^{-1/3} - 2x}{2}\right)\right\} \\ &= \max_{0 \leq x \leq a_\alpha} \left\{1 + x + 2\cos\left(\sqrt{\pi}k^{-1/3} + x\right)\right\}. \end{aligned}$$

The function $f(x) = 1 + x + 2\cos(\sqrt{\pi}k^{-1/3} + x)$ has derivative $f'(x) = 1 - 2\cos(\sqrt{\pi}k^{-1/3} + x)$. For $k \geq 100$ we have that $2\sqrt{\pi}k^{-1/3} + a_\alpha \leq \pi/6$. Thus $\cos(\sqrt{\pi}k^{-1/3} + x) \leq 1/2$ for all $x \in [0, a_\alpha]$, which implies that the function $f(x)$ is non-decreasing in the considered set. In order to find the maximum it is sufficient to consider its value at the extremal point a_α.

$$\begin{aligned} T_\alpha &\leq 1 + a_\alpha + 2\sin(\pi/2 - (\sqrt{\pi}k^{-1/3} + a_\alpha)) \\ &= 1 + \frac{\pi - 2\sqrt{\pi}k^{-1/3}}{\lceil k^{2/3}\rceil} + 2\cos\left(\sqrt{\pi}k^{-1/3} + \frac{\pi - 2\sqrt{\pi}k^{-1/3}}{\lceil k^{2/3}\rceil}\right) \end{aligned}$$

$$\begin{aligned}
&\leq\ 1+\frac{\pi-2\sqrt{\pi}k^{-1/3}}{\lceil k^{2/3}\rceil}+2-\left(\sqrt{\pi}k^{-1/3}+\frac{\pi-2\sqrt{\pi}k^{-1/3}}{\lceil k^{2/3}\rceil}\right)^2 \\
&\quad+\left(\sqrt{\pi}k^{-1/3}+\frac{\pi-2\sqrt{\pi}k^{-1/3}}{\lceil k^{2/3}\rceil}\right)^4/12 \\
&\leq\ 3+O(k^{-4/3})
\end{aligned}$$

Thus in this case the evacuation time $T \leq \max\{T_\alpha, T_\beta\} \leq 3+\frac{\pi}{k}+O(k^{-4/3})$.

Case 2. The exit is located within arc $\widehat{BA}$.

Each robot from group G_β explores an arc of length $(\pi+2\sqrt{\pi}k^{-1/3})/(k-k_\alpha)$. Thus time until the exit is discovered is at most $1+(\pi+2\sqrt{\pi}k^{-1/3})/(k-\lceil k^{2/3}\rceil)$. Since we are in the wireless communication model, each robot is notified immediately and needs additional time at most 2 to go to the exit. Thus the total evacuation time in this case is at most

$$\begin{aligned}
T &\leq 3+\frac{\pi+2\sqrt{\pi}k^{-1/3}}{k-k^{2/3}-1} \\
&= 3+\frac{\pi+2\sqrt{\pi}k^{-1/3}}{k}+\frac{(\pi+2\sqrt{\pi}k^{-1/3})(k^{2/3}+1)}{k(k-k^{2/3}-1)} \\
&= 3+\frac{\pi}{k}+O(k^{-4/3})
\end{aligned}$$

This completes the proof of Theorem 5.16. ■

Theorem 5.17. *It takes at least $3+\frac{\pi}{k}$ time in the worst case to evacuate $k \geq 2$ robots from an unknown exit located on the perimeter of the disk in the model with wireless communication.*

Proof. This is a direct consequence of Lemma 5.7 where $x = \pi/k$. ■

5.5 Conclusion

We studied the evacuation problem for k robots in a disk of unit radius and provided several algorithms in both local communication and wireless communication models for $k = 2$ and $k = 3$ robots. For the case of k robots we were able to give asymptotically tight bounds thus indicating a clear separation between the local communication and the wireless communication models. There are many interesting open questions. An interesting challenge would be to tighten our bounds or even determine optimal algorithms for $k = 2, 3$ robots. Another interesting class of problems is concerned with evacuation from more than one exit, or with robots having distinct maximal speeds. Finally, the geometric domain being considered, the starting positions of the robots, as well as the communication model provide challenging variants of the questions considered in this chapter.

Chapter 6

Conclusions

6.1 Overview

This thesis has presented a variety of solutions to various problems that can all be categorised as in the domain of Distributed Computing. The problems themselves have been concerned with an array of control problems for mobile robots in distributed settings. The models used take their inspiration from both network and geometric based environments.

The results presented in the preceding chapters are the result of published work carried out by the author, their supervisors and also several collaborators from different institutions. Below is a more detailed look at the conclusions that can be obtained from the main results presented in this thesis.

6.1.1 Robot Location Discovery

Chapter 3 Introduces a randomised distributed communication-less coordination mechanism for n uniform anonymous robots located on a circle with unit circumference with the goal to learn the positions of the other robots in the system as quickly as possible so that they are able to self organise into doing a more useful and involved task later on. It is assumed that the robots are located at arbitrary but distinct positions, unknown to other robots. The robots perform actions in synchronised rounds. At the start of each round an robot chooses the direction of its movement (*clockwise* or *anticlockwise*), and moves at unit speed during this round. robots are not allowed to overpass, i.e., when an robot collides with another it instantly starts moving with the same speed in the opposite direction. robots cannot leave marks on the ring, have zero vision and cannot exchange messages. However, on the conclusion of each round each robot has access to (some, not necessarily all) information regarding its trajectory during this round. This information can be processed and stored by the robot for further analysis.

The *Location Discovery Task* to be performed by each robot is to determine the initial position of every other robot and eventually to stop at its initial position, or proceed to another task, in a fully synchronised manner. The primary motivation was to study distributed systems where robots collect the minimum amount of information that is necessary to accomplish this location discovery task.

Our original result for this problem was a fully distributed randomised (Las Vegas type) algorithm, solving the *Location Discovery Task w.h.p.* in $O(n \log^2 n)$ rounds (assuming the robots collect sufficient information). Note that this result also holds if initially the robots do not know the value of n and they have no coherent sense of direction. We believe that our work in [68] is the first attempt to solve the distributed boundary patrolling problem in the geometric ring (circle) model. Furthermore, the proof technique of the concept of virtual "batons" that robots exchange with each other upon collision, we believe, is a novel and intriguing approach to analysing the motion of the robots in the system. To our knowledge this is the first time such an approach has been used to analyse such a system and it led to us discovering a rotation of robots positions at the end of each round. This in turn had a large impact on us designing and analysisng the resulting algorithm. This method has since been explored and built upon by [45] and [44].

However, Chapter 3 presents another fully distributed randomised (Las Vegas type) algorithm that can achieve success *w.h.p* significantly faster in $n + O(\log^2 n)$ rounds. It is also our conjecture that this new algorithm is in fact the optimal solution for this problem.

There are many applications to this work but the main principle that should be taken away from this in terms of practical applications is that even in the most limited distributed environments there can still exist an efficient method to enable collaboration on a larger goal, even if smaller tasks must first be achieved. For example, the work in Chapter 3 shows how a larger goal of the robots might be to efficently patrol the perimeter of the ring. However, given their limited capabilities each robot can employ the algorithms outlined in that chapter to learn the location of the other robots in the system.

When thinking of where the next step for the research presented in Chapter 3, where we had been looking at the case where the circumference of the ring was known to the robots, we believe that a natural question that follows is therefore whether one can solve the location discovery problem when it is the case that n is known but the circumference is not. This work that is presented here uses a randomised technique to achieve the results so it would be nice if we could create a deterministic solution to the problem. The model here would most likely be one where robots had their own unique identifiers. This problem could also be studied in a variety of different settings that could vary in complexity.

6.1.2 Evacuation Problem on the Line

After our work on limited robots in distributed settings it was a natural progression for us to move on to studying other settings and to see what sorts of limitations could be imposed on them and then what useful tasks can still be achieved. This is why we started to look at *group search problems*, or the *evacuation problem.* These results are presented in Chapter 4 and consider a problem in which k *MEs* located on the line perform search for a specific destination. The *MEs* are initially placed at the same point (origin) on the line L and the target is located at unknown distance d either to the left or to the right from the origin. All *MEs* must simultaneously occupy the destination, and the goal is to minimize the time necessary for this to happen. The problem with $k = 1$ is known as the *cow-path* problem, and the complexity of this problem is known to be $9d$ in the worst case (when the cow moves at unit speed); it is also known that this is the case for $k \geq 1$ unit-speed *MEs*. Presented in Chapter 4 is a clear argument for this claim by showing a rather counter-intuitive result. Namely independently from the number of *MEs*, group search cannot be performed faster than in time $9d$, where d is the distance between the origin and the destination. The case of $k = 2$ *MEs* with different speeds is also examined, showing a surprising result that the bound of $9d$ can be achieved when one *MEs* has unit speed, and the other *ME* moves with speed at least $\frac{1}{3}$. Finally the case where $k = 3$ *MEs* with one having a speed less than 1 is briefly looked at showing that a bound of $9d$ can yet again be achieved, but only if the slower *ME's* speed is at least $\frac{1}{5}$. This work can be seen to have its practical applications based heavily in the search and rescue space or indeed within military settings. Both spaces where we see more and more are employing the use of distributed robotics to aid in their tasks.

As stated already in Chapter 4 we have not provided a lower bound for the case when the speed of the slower robot(s) is not too small. Therefore, investigating this further would be a natural progression of our work and would be something that would have been included in this thesis had time allowed. Our work dealt with showing that in the new setting of the *evacuation problem* the optimal 9d bound is still obtainable. More generally, of course, there is also the impact the study of robots with multiple maximum speeds could have on rendezvous and search problems. One interesting change to the model shown in Chapter 4 could be to study problems where more than one evacuation point exists.

6.1.3 Evacuation Problem on the Disk

Following on from our work in Chapter 4 we could see that although the results were interesting that the setting was quite a simple one, being only a line. Therefore, it made sense to assume that any interesting results we obtained on the line would most likely become far more interesting if the same problem was considered in a more complex setting. In Chapter 5 k mobile robots inside a circular disk of unit radius are considered. The robots are required to evacuate the disk through an unknown exit point situated on its boundary. It is assumed all robots have the same (unit) maximal speed and start at the centre of the disk. The robots may communicate in order to inform each other about the presence (and its position) or the absence of an exit. The goal is for all the robots to evacuate through the exit in the minimum time possible.

Two models of communication between the robots were considered: In *non-wireless* (or *local*) *communication* model robots exchange information only when simultaneously located at the same point, and *wireless communication* in which robots can communicate between each other at any time.

The following question for different values of k have been studied: What is the optimal evacuation time for k robots? Algorithms are given here as well as lower bounds in both communication models for $k = 2$ and $k = 3$ thus indicating a difference in evacuation time between the two models. Almost-tight bounds are also obtained on the asymptotic relation between evacuation time and team size, for large k. Also in the local communication model it is shown that, a team of k robots can always evacuate in time $3+\frac{2\pi}{k}$, whereas at least $3+\frac{2\pi}{k}-O(k^{-2})$ time is sometimes required. In the wireless communication model, time $3+\frac{\pi}{k}+O(k^{-4/3})$ always suffices to complete evacuation, and at least $3+\frac{\pi}{k}$ is sometimes required. This shows a clear separation between the local and the wireless communication models.

We found that one of the remarkable points of interest for this problem was that when increasing the number of participating robots only slightly, and still when considering relatively small number of k, the compexity of the problem itself grew rapidly.

This work, as with the work shown in Chapter 4 on the line, can also be seen to have obvious applications in both a search and rescue and military setting.

As before when talking about the future research directions of the work done in Chapter 4, the next interesting point to consider for work done in Chapter 5 is to look at a model where there exists multiple evacuation points. This would indeed more accurately model a real world setting such as a search and rescue procedure performed on a collapsed building. Another, intriguing scenario would be if the robots found themselves at different starting locations from each other and indeed from the center of the disk itself. A natural continuation from this would also be to look at a geometric setting that was not the disk but perhaps some irregular polygon in the area that was covered as again this would more accurately model real world examples.

Our work on the disk has since been extended by [80] who are able to show improvements to both the upper and lower bounds for the evacuation of robots with face to face communication. They present an algorithm that provides an evacuation time that is at most 5.628 and show that any algorithm has evacuation time at least $3+\frac{\pi}{6}+\sqrt{3}\approx 5.255$.

6.2 Final Remarks

Over the course of this project we have felt that we have contributed to a new class navigation problems of autonomous robots being designed and explored within the field of Distribute Computing research. It is also felt that through this contribution we have gained a greater understanding of surrounding topics such as *Search and Rendezvous* problems and *Monitoring and Patrolling* problems. It is our hope that the work presented in this thesis and the papers that it draws from can also spark an interest in the community for *Evacuation Problems* as there is such a close relation to *Location Discovery* and *Rendezvous* based algorithms both in teams and as individuals. There has already been so much work done on this subject and by changing the model to represent that of an evacuation type problem it is likely that many more interesting questions can be proposed, especially when looking at cases where there are a multitude of maximum speeds for the robots present in the system.

Bibliography

[1] M. Abramowitz and I.A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* Dover, New York, ninth dover printing, tenth gpo printing edition, 1964.

[2] J.J Acevedo, B.C Arrue, J.M Diaz-Banez, I. Ventura, I. Maza, and A. Ollero. Decentralized strategy to ensure information propagation in area monitoring missions with a team of uavs under limited communications. In *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, pages 565–574, May 2013.

[3] J.J Acevedo, B.C. Arrue, I. Maza, and A.Ollero. Cooperative perimeter surveillance with a team of mobile robots under communication constraints. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 5067–5072, 2013.

[4] N. Agmon, S. Kraus, and G.A Kaminka. Multi-robot perimeter patrol in adversarial settings. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 2339–2345, 2008.

[5] S. Albers and M. Henzinge. Exploring unknown environments. *SIAM Journal on Computing*, 29(4):1164–1188, 2000.

[6] S. Albers, K. Kursawe, and S. Schuierer. Exploring unknown environments with obstacles. *Algorithmica*, 32(1):123–143, 2002.

[7] S. Alpern. *Rendezvous Search Games.* John Wiley and Sons, Inc., 2010.

[8] S. Alpern and S. Gal. *The Theory of Search Games and Rendezvous.* Kluwer Academic Publishing, Dordrecht, 2003.

[9] S. Alpern and T. Lidbetter. Mining coal or finding terrorists: The expanding search paradigm. *Operations Research*, 61(2):265–279, 2013.

[10] S. Alpern, V.J. Baston, and S. Essegaier. Rendezvous search on a graph. *Journal of Applied Probability*, 36(1):223–231, 1999.

[11] S. Alpern, R. Fokkink, r. Lindelauf, and G. Olsder. The "princess and monster" game on an interval. *SIAM Journal on Control and Optimization*, 47 (3):1178–1190, 2008.

[12] S. Alpern, R. Fokkink, L. Gąsieniec, R. Lindelauf, and V.S. Subrahmanian. *Search Theory, A Game Theoretic Perspective*. Springer, 2013.

[13] A. Anandkumar, A. Hassidim, and J. Kelner. Topology discovery of sparse random graphs with few participants. *Random Structures and Algorithms*, 43(1):16–48, 2013.

[14] E. J. Anderson. Mazes: Search games on unknown networks. *Networks*, 11 (4):393–397, 1981.

[15] H. Ando, I. Suzuki, and M. Yamashita. Formation and agreement problems for synchronous mobile robots with limited visibility. In *Intelligent Control, 1995., Proceedings of the 1995 IEEE International Symposium on*, pages 453–460, 1995.

[16] M.J Atallah. *Algorithms and Theory of Computation Handbook*. Chapman & Hall/CRC Applied Algorithms and Data Structures series. Taylor & Francis, 1998. ISBN 9781420049503.

[17] R.A. Baeza-Yates and R. Schott. Parallel searching in the plane. *Comput. Geom. Theory Appl.*, (3):143–154, Oct 1995.

[18] R.A. Baeza-Yates, J.C. Culberson, and G.J.E. Rawlins. Searching with uncertainty. In *SWAT 88: 1st Scandinavian workshop on algorithm theory*, pages 176–189, 1988.

[19] R.A. Baeza-Yates, J.C. Culberson, and G.J.E. Rawlins. Searching in the plane. *Information and Computation*, 106(2):234–252, 1991.

[20] E.B. Barbier, D. Moreno-Mateos, A.D. Rogers, J. Aronson, L. Pendleton, R. Danovaro, L. Henry, T. Morato, J. Ardron, and C.L. Van Dover. Ecology: Protect the deep sea. *Nature*, 505(7484):475–477, 2014.

[21] M. Baseggio, A. Cenedese, P. Merlo, and M. Pozziand L. Schenato. Distributed perimeter patrolling and tracking for camera networks. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 2093–2098, 2010.

[22] N. Baumann and M. Skutella. Earliest arrival flows with multiple sources. *Math. Oper. Res.*, 34(2):499–512, 2009.

[23] A. Beck. On the linear search problem. *Israel Journal of Mathematics*, 2(4): 221–228, 1964.

[24] M. Becker, F. Blatt, and H. Szczerbicka. A multi-agent flooding algorithm for search and rescue operations in unknown terrain. In Matthias Klusch, Matthias Thimm, and Marcin Paprzycki, editors, *Multiagent System Technologies*, volume 8076 of *Lecture Notes in Computer Science*, pages 19–28. Springer Berlin Heidelberg, 2013.

[25] R. Bellman. Minimization problem. In *Bull. Amer. Math. Soc*, page 270, 1956.

[26] R. Bellman. An optimal search problem. *SIAM Rev.*, 5:274, 1963.

[27] M.A. Bender and D. Slonim. The power of team exploration: Two robots can learn unlabeled directed graphs. In *35th Annual Symposium on Foundations of Computer Science (FOCS'94)*, pages 75–85, 1994.

[28] M.A Bender, A. Fernández, D. Ron, A. Sahai, and S. Vadhan. The power of a pebble: Exploring and mapping directed graphs. *Information and Computation*, 176(1):1 – 21, 2002.

[29] B. Bollobás. *Graph Theory, An Introductory course.* Springer-Verlag, New York, Heidelberg, Berlin, 1st edition, 1979. ISBN 0387903992, 9780387903996.

[30] P. Bose, J.L. De Carufel, and S. Durocher. Revisiting the problem of searching on a line. In *Algorithms – ESA 2013*, volume 8125 of *Lecture Notes in Computer Science*, pages 205–216. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-40449-8. doi: 10.1007/978-3-642-40450-4_18.

[31] W. Burgard, M. Moors, C. Stachniss, and F. Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21:376–386, 2005.

[32] J. Chalopin, P. Flocchini, B. Mans, and N. Santoro. Network exploration by silent and oblivious robots. In *36th International Workshop on Graph Theoretic Concepts in Computer Science (WG'10)*, pages 208–219, 2010.

[33] R.R. Choudhury, S. Bandyopadhyay, and K. Paul. A distributed mechanism for topology discovery in ad hoc wireless networks using mobile agents. In *Proceedings of the 1st ACM International Symposium on Mobile Ad Hoc Networking & Computing*, MobiHoc '00, pages 145–146. IEEE Press, 2000.

[34] M. Chrobak, L. Gąsieniec, T. Gorry, and R. Martin. Group search on the line. In *SOFSEM 2015: Theory and Practice of Computer Science*, volume 8939, pages 164–176. Springer Berlin Heidelberg, 2015. ISBN 978-3-662-46077-1.

[35] T.H. Chung, G.A. Hollinger, and V. Isler. Search and pursuit-evasion in mobile robotics. *Auton. Robots*, 31(4):299–316, 2011.

[36] V. Chvátal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B*, 18(1):39 – 41, 1975.

[37] M. Cieliebak, P. Flocchini, G. Prencipe, and N. Santoro. Solving the robots gathering problem. In *Automata, Languages and Programming*, volume 2719 of *Lecture Notes in Computer Science*, pages 1181–1196. Springer Berlin Heidelberg, 2003.

[38] CNET. Meet amazon's busiest employee – the kiva robot, 2014. URL `http://www.cnet.com/news/meet-amazons-busiest-employee-the-kiva-robot/`.

[39] R. Cohen and D. Peleg. Local spreading algorithms for autonomous robot systems. *Theor. Comput. Sci.*, 399(1-2):71–82, June 2008. ISSN 0304-3975.

[40] A. Collins, J. Czyzowicz, L. Gąsieniec, and A. Labourel. Tell me where i am so i can meet you sooner. In *Automata, Languages and Programming*, volume 6199 of *Lecture Notes in Computer Science*, pages 502–514. 2010.

[41] C. Cooper, A.M. Frieze, and T. Radzik. Multiple random walks and interacting particle systems. In *36th Internatilonal Colloquium on Automata, Languages and Programming (ICALP'09)*, pages 399–410, 2009.

[42] J. Czyzowicz, L. Gąsieniec, A. Kosowski, and E. Kranakis. Boundary patrolling by mobile agents with distinct maximal speeds. In *Algorithms – ESA 2011*, volume 6942 of *Lecture Notes in Computer Science*, pages 701–712. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-23718-8.

[43] J. Czyzowicz, A. Labourel, and A. Pelc. Optimality and competitiveness of exploring polygons by mobile robots. *Information and Computation*, 209(1): 74–88, 2011.

[44] J. Czyzowicz, L. Gąsieniec, A. Kosowski, E. Kranakis, O.M. Ponce, and E. Pacheco. Position discovery for a system of bouncing robots. In *Distributed Computing*, volume 7611, pages 341–355. 2012.

[45] J. Czyzowicz, E. Kranakis, and E. Pacheco. Localization for a system of colliding robots. In *Proceedings of the 40th International Conference on Automata, Languages, and Programming - Volume Part II*, ICALP'13, pages 508–519, 2013.

[46] J. Czyzowicz, L. Gąsieniec, T. Gorry, E. Kranakis, R. Martin, and D Pająk. Evacuating robots via unknown exit in a disk. In *Distributed Computing*, volume 8784 of *Lecture Notes in Computer Science*, pages 122–136. Springer Berlin Heidelberg, 2014. ISBN 978-3-662-45173-1.

[47] A. Dagan and S. Gal. Network search games, with arbitrary searcher starting point. *Networks*, 52(3):156–161, 2008.

[48] R. D'Andrea and P. Wurman. Future challenges of coordinating hundreds of autonomous vehicles in distribution facilities. In *Technologies for Practical Robot Applications, 2008. TePRA 2008. IEEE International Conference on*, pages 80–83, 2008.

[49] L.M. de Campos, J.M. Fernández-Luna, J.A. Gámez, and J.M. Puerta. Ant colony optimization for learning bayesian networks. *International Journal of Approximate Reasoning*, 31(3):291 – 311, 2002.

[50] N.D. Dendris, L.M. Kirousis, and D.M. Thilikos. Fugitive-search games on graphs and related parameters. *Theoretical Computer Science*, 172(1–2):233 – 254, 1997.

[51] X. Deng, T. Kameda, and C. Papadimitriou. How to learn an unknown environment. In *Proceedings of the 32Nd Annual Symposium on Foundations*

of Computer Science, SFCS '91, pages 298–303. IEEE Computer Society, 1991.

[52] X. Deng, T. Kameda, and C. Papadimitriou. How to learn an unknown environment i: the rectilinear case. *Journal of ACM*, 45(2):215–245, 1998.

[53] A. Dessmark and A. Pelc. Optimal graph exploration without good maps. *Theoretical Computer Science*, 326(1–3):343 – 362, 2004.

[54] S. Devismes, F. Petit, and S. Tixeuil. Optimal probabilistic ring exploration by semi-synchronous oblivious robots. *Theoretical Computer Science*, 498 (0):10 – 27, 2013.

[55] Y. Dieudonné and A. Pelc. Anonymous meeting in networks. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '13*, pages 737–747, 2013.

[56] K. Diks, P. Fraigniaud, E. Kranakis, and A. Pelc. Tree exploration with little memory. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '02, pages 588–597. Society for Industrial and Applied Mathematics, 2002.

[57] S. Dolev. *Self-stabilization.* MIT Press, 2000. ISBN 9780262041782.

[58] M. Dorigo and L.M. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *Evolutionary Computation, IEEE Transactions on*, 1(1):53–66, 1997.

[59] J. Du, E. Kranakis, O.M. Ponce, and S. Rajsbaum. Neighbor discovery in a sensor network with directional antennae. In Thomas Erlebach, Sotiris Nikoletseas, and Pekka Orponen, editors, *Algorithms for Sensor Systems*, volume 7111 of *Lecture Notes in Computer Science*, pages 57–71. Springer Berlin Heidelberg, 2012.

[60] A. Einstein. Erklarung der perihelionbewegung der merkur aus der allgemeinen relativitatstheorie. *Sitzungsber. preuss. Akad. Wiss., vol. 47, No. 2, pp. 831-839, 1915*, 47:831–839, 1915.

[61] Y. Elmaliach, N. Agmon, and G.A Kaminka. Multi-robot area patrol under frequency constraints. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 385–390, 2007.

[62] S. Fekete, C. Gray, and A. Kröller. Evacuation of rectilinear polygons. In *Combinatorial Optimization and Applications*, volume 6508 of *Lecture Notes in Computer Science*, pages 21–30. Springer Berlin Heidelberg, 2010.

[63] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Hard tasks for weak robots: The role of common knowledge in pattern formation by autonomous mobile robots. In *Algorithms and Computation*, volume 1741 of *Lecture Notes in Computer Science*, pages 93–102. Springer Berlin Heidelberg, 1999. ISBN 978-3-540-66916-6.

[64] P. Flocchini, p. Giuseppe, N. Santoro, and P. Widmayer. Pattern formation by autonomous robots without chirality. In *In Proc. SIROCCO*, pages 147–162, 2001.

[65] F.V. Fomin and D.M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theor. Comput. Sci.*, 399(3):236–245, June 2008.

[66] P. Fraigniaud, D. Ilcinkas, G. Peer, A. Pelc, and D. Peleg. Graph exploration by a finite automaton. *Theoretical Computer Science*, 345(2–3):331 – 344, 2005. Mathematical Foundations of Computer Science 2004 Mathematical Foundations of Computer Science 2004.

[67] P. Fraigniaud, L. Gąsieniec, D.R. Kowalski, and A. Pelc. Collective tree exploration. *Networks*, 48(3):116–177, 2006.

[68] T. Friedetzky, L. Gąsieniec, T. Gorry, and R. Martin. Observe and remain silent (communication-less agent location discovery). In *Mathematical Foundations of Computer Science 2012 - 37th International Symposium (MFCS'12)*, pages 407–418, 2012.

[69] J.G.M Fu and V.M.H Ang. Probabilistic ants (pants) in multi-agent patrolling. In *Advanced Intelligent Mechatronics, 2009. AIM 2009. IEEE/ASME International Conference on*, pages 1371–1376, 2009.

[70] S. Gal. Search games with mobile and immobile hider. *SIAM Journal on Control and Optimization*, 17(1):99–122, 1979.

[71] A.A. Galyaev and E.P. Maslov. Patrolling a barrier with a network of mobile sensors. *Journal of Computer and Systems Sciences International*, 52(2): 326–332, 2013.

[72] S.K. Ghosh and R. Klein. Online algorithms for searching and exploration in the plane. *Computer Science Review*, 4(4):189 – 201, 2010.

[73] L. Gąsieniec, E. Kranakis, D. Krizanc, and X. Zhang. Optimal memory rendezvous of anonymous mobile agents in a unidirectional ring. In *SOFSEM 2006: Theory and Practice of Computer Science*, volume 3831 of *Lecture Notes in Computer Science*, pages 282–292. Springer Berlin Heidelberg, 2006.

[74] B. Gluss. An alternative solution to the "lost at sea" problem. *Naval Research Logistics Quarterly*, 8(1):117–122, 1961.

[75] T. Gorry. Distributed Agent Location Discovery on the Ring. Master's thesis, The University of Liverpool, Department of Computer Science, Liverpool, UK, 2011.

[76] E. Guizzo. Three engineers, hundreds of robots, one warehouse. *Spectrum, IEEE*, 45(7):26–34, 2008.

[77] M. Hammar, B.J. Nilsson, and S. Schuierer. Parallel searching on m rays. *Computational Geometry*, 18(3):125 – 139, 2001.

[78] F. Hoffmann, C. Icking, R. Klein, and K. Kriegel. The polygon exploration problem. *SIAM J. Computing*, 31(2):577–600, 2001.

[79] J. Isbellr. Pursuit around a hole. *Naval Research Logistics Quarterly*, 14(4): 569–571, 1967.

[80] E. Kranakis L. Narayanan J. Opatrny J. Czyzowicz, K. Georgiou and B. Vogtenhuber. Evacuating robots from a disk using face-to-face communication. arXiv:1501.04985.

[81] A. Jeż and J. Łopuzański. On the two-dimensional cow search problem. *Information Processing Letters*, 109(11):543 – 547, 2009.

[82] M.Y. Kao, J.H. Reif, and S.R. Tate. Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem. In *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '93, pages 441–447, 1993.

[83] S. Kijima, M. Yamashita, and Y. Yamauchi. Private communication, 2013.

[84] R. Klasing, E. Markou, and A. Pelc. Gathering asynchronous oblivious mobile robots in a ring. In Tetsuo Asano, editor, *Algorithms and Computation*, volume 4288 of *Lecture Notes in Computer Science*, pages 744–753. Springer Berlin Heidelberg, 2008.

[85] J.M. Kleinberg. On-line search in a simple polygon. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '94, pages 8–15, 1994.

[86] E. Koutsoupias, C.H. Papadimitriou, and M. Yannakakis. Searching a fixed graph. In *Automata, Languages and Programming*, volume 1099 of *Lecture Notes in Computer Science*, pages 280–289. 1996.

[87] E. Kranakis, D. Krizanc, and E. Markou. *The Mobile Agent Rendezvous Problem in the Ring.* Morgan and Claypool Publishers, 1st edition, 2010. ISBN 1608451364, 9781608451364.

[88] J. Lategahn, M. Mulle, and C. Rohrig. Global localization of automated guided vehicles in wireless networks. In *Wireless Systems (IDAACS-SWS), 2012 IEEE 1st International Symposium on*, pages 7–12, 2012.

[89] H. Li and K P. Chong. Search on lines and graphs. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 5780–5785, 2009.

[90] T. Lidbetter. *Hide-and-seek and Other Search Games.* PhD thesis, London School of Economics, 2013.

[91] X. Liu and D. Gong. A comparative study of a-star algorithms for search and rescue in perfect maze. In *Electric Information and Control Engineering (ICEICE), 2011 International Conference on*, pages 24–27, 2011.

[92] N.A. Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers Inc., 1996. ISBN 1558603484.

[93] Wired Magazine. Autonomous robots invade retail warehouses, 2009. URL `http://www.wired.com/2009/01/retailrobots/`.

[94] J.P. Massias and G. Robin. Bornes effectives pour certaines fonctions concernant les nombres premiers. *Journal de théorie des nombres de Bordeaux*, 8(1):215–242, 1996.

[95] Y. Meng and J. Gan. A distributed swarm intelligence based algorithm for a cooperative multi-robot construction task. In *Swarm Intelligence Symposium, 2008. SIS 2008. IEEE*, pages 1–6, 2008.

[96] H. Minkowski and A. Sommerfeld. Raum und zeit. In *Das Relativitätsprinzip*, Fortschritte der Mathematischen Wissenschaften in Monographien, pages 54–71. Vieweg+Teubner Verlag, 1923. ISBN 978-3-663-19372-2. doi: 10.1007/978-3-663-19510-8_5. URL http://dx.doi.org/10.1007/978-3-663-19510-8_5.

[97] J.S.B. Mitchell. Geometric shortest paths and network optimization. In *Handbook of Computational Geometry*, pages 633 – 701. North-Holland, 2000.

[98] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995. ISBN 0-521-47465-5, 9780521474658.

[99] P.J. Nahin. *Chases and Escapes: The Mathematics of Pursuit and Evasion*. Princeton University Press, 2007.

[100] F. Ooshita, S. Kawai, H. Kakugawa, and T. Masuzawa. Randomized gathering of mobile agents in anonymous unidirectional ring networks. *Parallel and Distributed Systems, IEEE Transactions on*, 25(5):1289–1296, 2014.

[101] J. Ota. Multi-agent robot systems as distributed autonomous systems. *Advanced Engineering Informatics*, 20(1):59 – 70, 2006.

[102] C.H. Papadimitriou and M. Yannakakis. Shortest paths without a map. *Theoretical Computer Science*, 84(1):127 – 150, 1991.

[103] F. Pasqualetti, A. Franchi, and F. Bullo. On cooperative patrolling: Optimal trajectories, complexity analysis, and approximation algorithms. *Robotics, IEEE Transactions on*, 28(3):592–606, 2012.

[104] D Pająk. Private communication, 2014.

[105] D. Portugal and R.P. Rocha. Distributed multi-robot patrol: A scalable and fault-tolerant framework. *Robotics and Autonomous Systems*, 61(12):1572 – 1587, 2013.

[106] D. Portugal, M.S Couceiro, and R.P. Rocha. Applying bayesian learning to multi-robot patrol. In *Safety, Security, and Rescue Robotics (SSRR), 2013 IEEE International Symposium on*, pages 1–6, 2013.

[107] A.M. Reynolds and F. Bartumeus. Optimising the success of random destructive searches: Lévy walks can outperform ballistic motions. *Journal of Theoretical Biology*, 260(1):98 – 103, 2009.

[108] S. Sand, S. Zhang, M. Muhlegg, G. Falconi, C. Zhu, T. Kruger, and S. Nowak. Swarm exploration and navigation on mars. In *Localization and GNSS (ICL-GNSS), 2013 International Conference on*, pages 1–6, 2013.

[109] N. Santoro. *Design and Analysis of Distributed Algorithms (Wiley Series on Parallel and Distributed Computing).* Wiley-Interscience, 2006. ISBN 0471719978.

[110] K.S. Senthilkumar and K.K. Bharadwaj. Multi-robot exploration and terrain coverage in an unknown environment. *Robotics and Autonomous Systems*, 60(1):123 – 132, 2012.

[111] S.L. Smith and D. Rus. Multi-robot monitoring in dynamic environments with guaranteed currency of observations. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 514–521, Dec 2010.

[112] K. Sugihara and I. Suzuki. Distributed algorithms for formation of geometric patterns with many mobile robots. *Journal of Robotic Systems*, 13:127–139, 1996.

[113] I. Suzuki and M. Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM Journal on Computing*, 28:1347–1363, 1999.

[114] Kiva Systems. Kiva systems company website, 2015. URL `http://www.kivasystems.com`.

[115] C. Sze, K. New, A. Peng, and I. Rekleitis. Distributed coverage with multi-robot systems. In *International Conference on Robotics and Automation 2006 (ICRA'06)*, pages 2423–2429, 2006.

[116] T. Temple and E. Frazzoli. Whittle-indexability of the cow path problem. In *American Control Conference (ACC), 2010*, pages 4152–4158, 2010.

[117] D. Teodorovic and M. Dell'Orco. Bee colony optimization – a cooperative learning approach to complex transportation problems. In *10th EWGT Meeting and 16th Mini-EURO Conference*, 2005.

[118] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5):335–363, 2001.

[119] D. Volchenkov, J. Helbach, M. Tscherepanow, and S. Küheel. Exploration-exploitation trade-off in a treasure hunting game. *Electronic Notes in Theoretical Computer Science*, 299(0):101 – 121, 2013. Proceedings of the fourth International Workshop on Interactions between Computer Science and Biology (CS2Bio'13).

[120] J.L. Welch and H. Attiya. *Distributed Computing: Fundamentals, Simulations and Advanced Topics.* McGraw-Hill, Inc., 1998. ISBN 0-07-709352-6.

[121] L. Wischoff, A. Ebner, H. Rohling, M. Lott, and R. Halfmann. Sotis - a self-organizing traffic information system. In *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, volume 4, pages 2442–2446 vol.4, 2003.

[122] B. Yamauchi. Frontier-based exploration using multiple robots. In *Proceedings of the Second International Conference on Autonomous Agents*, AGENTS '98, pages 47–53, 1998.