Robot Evacuation in a Disk with Arbitrary Starting Positions

Honours Project COMP 4905

Student Name: Nicolas Perez
Student ID: 100978917
Supervisor: Evangelos Kranakis

April 17, 2019

Abstract

Imagine a scenario in which there are 2 robots which can move at most 1 unit speed are placed inside of a room (AKA a disk) whose perimeter is in the shape of a perfect circle of 1 unit radius. The 2 robots must find the exit along the perimeter and leave the room in the shortest amount of time. In this project, I will examine this problem from a mathematical perspective, considering a 'perfect' model for simplicity and considering various sub-scenarios which determine where they are initially placed inside of the disk and how much information they know about their initial placements. The robots can only find the exit once they come into contact with it, and both robots can communicate anything they know or have learnt about their environment to the other robots. This includes but is not limited to their trajectory, their GPS coordinates, the location of the perimeter of the disk (if it is known) and the exit to the disk (again, if it is known).
Using knowledge from various pieces of work on similar variations of this problem as well as some of my own original work, I will show algorithms for multiple variations of the problem which are all proven to be at-most 1 unit of time off from optimal solutions. I will also discuss a modified model for the problem as well as demonstrate methods for getting the expected run time of evacuation algorithm which both might be useful for future work on the evacuation problem.

Key words and phrases. Arbitrary Location, Circle, Communication, Disk, Evacuation, Mobile Robot, Wireless.

Contents

1 Introduction **4**
- 1.1 Motivation . . . 4
- 1.2 Model . . . 4
- 1.3 Related Work . . . 5
- 1.4 Outline and Results of the Project . . . 6

2 One Robot at the Center, the Robots Know the Center of the Disk **9**
- 2.1 Upper Bound . . . 9
- 2.2 Lower Bound . . . 11
- 2.3 Optimality . . . 12

3 One Robot at the Center, the Robots Don't Know the Center of the Disk **13**
- 3.1 Upper Bound . . . 13
- 3.2 Comparison of Algorithm A and Algorithm B . . . 16
- 3.3 Lower Bound . . . 17

4 Both Robots at the Same Arbitrary Location **18**
- 4.1 Upper Bound . . . 18
- 4.2 Lower Bound . . . 21

5 Both Robots at Different Arbitrary Locations **23**
- 5.1 Upper Bound . . . 23
- 5.2 Lower Bound . . . 26

6 Modified Model **30**
- 6.1 Both Robots at the Same Arbitrary Location . . . 30

7 Randomized Algorithm Simulations **34**
- 7.1 Both Robots at the Center, the Robots Know the Center of the Disk . . . 35
- 7.2 Both Robots at the Same Random Location, the Robots Don't Know the Center of the Disk . . . 36
- 7.3 Both Robots at the Same Random Location, the Robots Know the Center of the Disk . . . 39

8 Conclusion **41**

A Lemma A.1 **43**

B Lemma B.1 **47**

C Lemma C.1 **48**

D Code For Simulations **49**

1 Introduction

1.1 Motivation

The robot evacuation problem has been studied a lot with many different variations, but has not been studied very much with variations of the robots starting in arbitrary positions. This project will discuss a few different scenarios in an attempt to get a better understanding of the added complexities of having the robots start in arbitrary positions. Analyzing upper bounds in this variant of the problem is very interesting since depending on the algorithm used, the robots may have a different initial placement which yields the worst case run time. On top of the original intricacies involved when the robots start in a known position, a good understanding of the compromises between finding the perimeter quickly and not having one of the robots be too far away from the perimeter of the disk when it is found is neccessary. This makes finding good lower bounds more difficult compared to when the robots start in a known position.

1.2 Model

Two robots are placed in a disk-shaped room of radius 1, the robots can move at maximum speed 1 and the exit to the room is some arbitrary point on the perimeter of the disk (the term disk and room can be used interchangeably). The walls of the room, better described as a single wall,

forms a perfect circle (popularly called a disk in the field of computer science). The goal is for the robots to find the exit and evacuate the room in the lowest amount of time. Throughout the project, if an algorithm is described and does not explicitly state the speed of the robots at a given step, then it should be assumed the robots are travelling at maximum speed 1.

The robots can not see where the exit is or see where the perimeter of the room is, but they can detect them when in contact with them. The robots also always know each other's coordinates. With no delay, the robots can communicate with each other wirelessly anything they compute which is based on the instructions they are given and/or any information they gather. This then allows the robots to change their movement based on any computable information given what they know about each of their positions and the room. It also includes them being able to measure the angles they are traveling at with respect to each other, the line intersecting their current locations, and instruct one another the trajectory needed to evacuate the disk more quickly given any acquired knowledge such as finding the perimeter of or exit to the disk. The robots perfectly compute information and with no delay, meaning as soon as a robot reaches the perimeter both robots know instantly where they are with respect to the disk.

This type of model is given to simplify the mathematics behind the generic problem of robots evacuating a circular-shaped room. Varius factors found in a real life scenario such as the time it takes for the robots to turn, the time it takes for them to compute information etc. would make the problem much harder to be discussed concisely and with mathematics. In the abstract I refer to this as a 'perfect' model of the problem.

1.3 Related Work

Currently, there is a known algorithm for two robots starting at the center of the disk to evacuate in worst-case $1 + \frac{2\pi}{3} + \sqrt{3} \approx 4.83$ time (this is called an upper bound)[1]. It is also shown that the best possible algorithm for evacuating in this scenario should have a worst-case time of ≈ 4.83 (this is

called a lower bound). Different versions of the problem (scenarios) in this project will be discussed which differ based on how the robots are initially placed in the disk and how much information they are given about where they are placed.

Other variations of the evacuation from a disk problem exist, such as evacuating using face-to-face communication and evacuating with one of the robots being faulty [2, 3]. The problem of evacuating robots from equilateral triangles and squares has also been studied [4]. Another modification to the problem where only a single robot called the 'queen' needs to evacuate from a disk and recieves help from 'servant' robots which aren't required to evacuate the disk [5]

1.4 Outline and Results of the Project

In each of the sections 2, 3, 4 and 5 a different variation of the evacuation problem will be presented with an upper bound(s) and a lower bound proof. Section 6 will describe a modified model that restricts how the robots can move and will show that the algorithms described in section 4 and 5 are optimal under the modified model. Section 7 will discuss 3 different algorithms each for a different evacuation scenario and their expected run times. Section 8 will be a conclusion including a summary of the project and suggested further work on the evacuation problem with arbitrarily placed robots. The appendix at the very end of this document will contain proofs for lemmas used throughout the project.

Summary of Results for Upper Bounds and Lower Bounds		
Model	Upper Bound	Lower Bound
One Robot at the Center, the Robots Know the Center of The Disk	$= 1 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$ $\approx 4.83 - y/2$ Theorem 2.1	$= 1 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$ $\approx 4.83 - \frac{y}{2}$ Theorem 2.3
One Robot at the Center, the Robots Don't Know the Center of The Disk	$\begin{cases} \sim 4.83 & y < \frac{2}{3} \\ \sim 5.83 - \frac{3y}{2} & \frac{2}{3} \leq y \end{cases}$ Theorem 3.3	$= 1 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$ $\approx 4.83 - \frac{y}{2}$ Theorem 3.4
Both Robots at the Same Arbitrary Location	$= 2 + \frac{2\pi}{3} + \sqrt{3}$ ≈ 5.83 Theorem 4.1, 4.2	$= 1 + \frac{2\pi}{3} + \sqrt{3}$ ≈ 4.83 Theorem 4.3
Both Robots Placed in Arbitrary Positions	$= 2 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$ $\approx 5.83 - \frac{y}{2}$ Theorem 5.1, 5.2	$= 1 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$ $\approx 4.83 - \frac{y}{2}$ Theorem 5.3

Table 1: A table summarizing the upper bounds and lower bounds of each scenario where y is the distance the robots start from each other.

Summary of Results for Randomized Algorithms		
Model	Simulated Average Run Time	Mathematically Calculated Average Run Time
Both Robots at the Center, the Robots Know the Center of the Disk	3.8459 ± 0.0068 Section 7.1	$= 1 + \frac{\pi}{2} + \frac{4}{\pi}$ ≈ 3.84 Theorem 7.1
Both Robots at the Same Arbitrary Location, the Robots Don't Know the Center of the Disk	3.6887 ± 0.0076 Section 7.2	$\frac{8}{3\pi} + \frac{\pi}{2} + \frac{4}{\pi}$ ≈ 3.69 Theorem 7.2
Both Robots at the Same Arbitrary Location, the Robots Know the Center of the Disk	3.1819 ± 0.0069 Section 7.3	$\frac{1}{3} + \frac{\pi}{2} + \frac{4}{\pi}$ ≈ 3.18 Theorem 7.3

Table 2: A table summarizing the simulated average run time of the aglorithm used for each scenario using a 95% confidence interval and 100000 trials for each algorithm.

2 One Robot at the Center, the Robots Know the Center of the Disk

The first scenario in this project that will be discussed is the one in which one robot is at the center of the disk and the other one is placed in some arbitrary position inside the disk. Both robots know which robot is at the center of the disk as well, so they can each compute their shortest paths to reach the perimeter of the disk. As mentioned in the introduction, there is a known optimal solution already for when both robots are placed in the center of the disk and they know they are at the center. The proof in this project for an optimal algorithm when one robot is placed in an arbitrary position is very similar.

2.1 Upper Bound

Theorem 2.1 *There exists an algorithm that takes* $1 + x + 2\sin\left(x + \frac{y}{2}\right)$ *time for the robots to evacuate, where* x *is the time the robots have been exploring the perimeter simultaneously and* y *is the distance they start apart from each other.*

Proof. An even shorter run time can be given if one of the robots finds the exit to the disk before the other one reaches the perimeter which is proven in Lemma A.1. For purposes of only examining worst case scenarios, the run time for that case will not be examined. First the algorithm will be stated, then the time complexity will be analyzed.

Shortest Path to the Perimeter Algorithm
1. The robots orient themselves by computing the line intersecting their starting positions, and then both travel along that line in the direction that gives the randomly placed robot the shortest path to the perimeter of the disk.
2. Once robot r_1 reaches the perimeter of the disk it starts exploring the perimeter CCW and once robot r_2 reaches the perimeter it starts exploring the perimeter CW.
3. Once a robot finds the exit of the room, the other robot is notified of the location to the exit and then travels the shortest path to the exit.

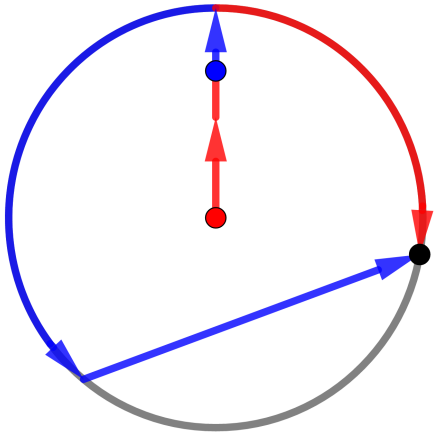

Figure 1: An example of a worst case run time for when $y = 0.7$. The red and blue dots are the starting positions of the robots r_2 and r_1 respectively, and the red and blue arrows indicate the paths taken. The black dot is the exit to the disk.

The run time for this algorithm is then:

$$\begin{aligned} f(x) &= 1 + x + 2\sin\left(\frac{2x+y}{2}\right) \\ &= 1 + x + 2\sin\left(x + \frac{y}{2}\right) \end{aligned}$$

Where y is the distance the two robots start apart from each other and x is the amount of time the robot that started at the center of the circle spends along the perimeter. It takes time 1 for robot r_2 to reach the perimeter of the disk and then it spends time x along the perimeter. The term with the sinusoidal function is to compute the chord one of the robots must travel once the exit is found on the perimeter, and both robots are on the perimeter. This then gives all three terms for the run time. Since the robots can figure out the orientation with respect to each other, the run time of this algorithm would be the same for if the robot that was randomly placed

could only be placed along the vertical line intersecting the center of the disk.

■

2.2 Lower Bound

The lower bound for this scenario makes use of a theorem from a published paper, [1], which states:

Theorem 2.2 *Consider a perimeter of a disk whose subset of total length $u + \epsilon > 0$ has not been explored for some $\epsilon > 0$ and $\pi \geq u > 0$. Then there exist two unexplored boundary points between which the distance along the perimeter is at least u.*

Theorem 2.2 will be used to help prove theorem 2.3. Theorem 2.3 follows the same structure as a lower bound proof found in the same paper, [1], as theorem 2.2.

Theorem 2.3 *For any x satisfying $\frac{\pi - y}{2} \leq x < \pi - \frac{y}{2}$ and $y \leq x$ it takes at least $1 + x + 2\sin(\pi - x - \frac{y}{2})$ time for both robots to evacuate from the disk given one robot starts at the center, one is randomly placed and they both know the center of the circle.*

Proof. Given an algorithm for the robots to evacuate, it takes at least time $1 - y$ for one robot and time 1 for the other robot to reach the perimeter of the disk. While both robots are exploring on the perimeter at time $1 + x$, at most $2x + y$ of the perimeter can be explored. This then means that there would be at least $2\pi - 2x - y$ of the perimeter unexplored and $\pi \geq 2\pi - 2x - y > 0$. Using theorem 2.2 and setting ϵ to approach 0, there would then be a chord of at least $2\sin(\pi - x - \frac{y}{2})$ length at time $1 + x$ that has both of its endpoints in an unexplored part of the perimeter. If the exit was located in the end of the chord that is explored last, then one of the robots would have to travel at least the length of that chord to evacuate

once the exit is found. This means any given algorithm takes at least $1 + x + 2\sin(\pi - x - \frac{y}{2})$ time to evacuate both robots.
■

2.3 Optimality

The algorithm described in section 2.1 is optimal since it has the same run time as the lower bound:

$$\begin{aligned} f(x) &= 1 + x + 2\sin\left(x + \frac{y}{2}\right) \\ &= 1 + x + 2\sin\left(\pi - x - \frac{y}{2}\right) \end{aligned}$$

Lemma A.1 proves that the worst case run time for when a robot finds the exit before both of the robots reach point Z is always less than the worst case for when they find the exit and both robots are on the perimeter. Therefore it is an optimal algorithm.

Using Lemma A.1 the worst case run time of the algorithm is:

$$\frac{(1-y)+1}{2} + \frac{2\pi}{3} + \sqrt{3} = \frac{2-y}{2} + \frac{2\pi}{3} + \sqrt{3} = 1 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$$

3 One Robot at the Center, the Robots Don't Know the Center of the Disk

One scenario involves one robot being placed at the center of the disk and the other robot being placed in some arbitrary position in the disk. The robots can still communicate each other's initial starting positions, and they know that one of them is at the center but don't know which robot it is. They also make use of computing the line intersecting each other's positions in the algorithms described in this section. Two different algorithms will be used depending on how far apart the robots initially start from each other to provide a better upper bound on the run time. One algorithm involves the robots moving in opposite directions from each other to find the perimeter and the other algorithm involves the robots moving in straight lines so that they will intersect in time 1.

3.1 Upper Bound

Theorem 3.1 *There is an algorithm for when one robot is at the center and the other one is placed randomly, and they don't know where the center is that runs in at most* ≈ 4.83 *time.*

Proof. How the algorithm works, is each robot travels in a straight line at a trajectory so that the robots meet in exactly 1 unit of time. Since one robot is located at the center of the disk, they will reach the perimeter of the disk at the same time that they meet. They can do this by first calculating the distance y between each other, and then each moving at an angle α computed by $cos^{-1}(\frac{y}{2})$ with respect to the line intersecting their initial starting positions. If they happen to be placed in the same spot, they then pick an arbitrary direction and move to the perimeter together in 1 unit of time.

Algorithm A
1. The robots compute the paths needed to travel along to meet at some point on the perimeter in exactly 1 unit of time. 2. Robot r_2 starts exploring the perimeter CCW and r_1 starts exploring the perimeter CW. 3. Once a robot finds the exit of the room, the other robot is notified of the location of the exit and then travels the shortest path to the exit.

The run time for this algorithm is identical to if they both started at the center together and knew they were both at the center[1]. The max run time is:

$$1 + \frac{2\pi}{3} + \sqrt{3} \approx 4.83 \tag{1}$$

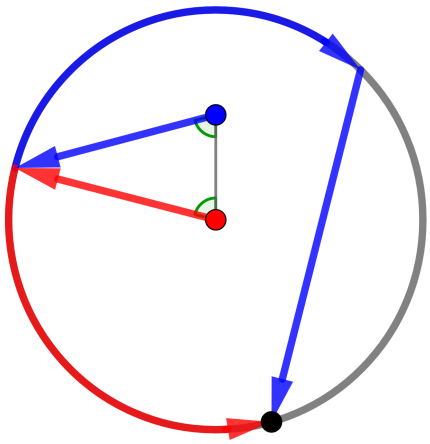

Figure 2: An example of a worst case run time for when $y = 0.5$. The blue and red dots are the starting positions of the robots r_1 and r_2 respectively, and the red and blue arrows indicate the paths taken. The black dot is the exit to the disk. The green segments indicate the angle α with respect to the grey line intersecting the starting positions of the robots.

■

Theorem 3.2 *There is an algorithm for when one robot is at the center and the other one is placed randomly, and they don't know where the center is that runs in at most* $2 - \frac{3y}{2} + \frac{2\pi}{3} + \sqrt{3}$ *time.*

Proof. First the algorithm will be stated, then the time complexity will be analyzed.

Algorithm B
1. The robots orient themselves by computing the line intersecting their starting positions, and then both travel in opposite directions away from each other.
2. Robot r_1 reaches some point Z along the perimeter and starts exploring the perimeter CCW and notifies r_2 of the location of Z.
3. r_2 takes the shortest path to Z and starts exploring the perimeter CW once it reaches Z.
4. Once a robot finds the exit of the room, the other robot is notified of the location of the exit and then travels the shortest path to the exit.

Algorithm B has a worst case run time based on how far apart the robots start. Ignoring one of the two cases when a robot finds the exit before both robots reach the perimeter, the runtime is given by the following equation:

$$\begin{aligned} f(x) &= 1 + 2(1-y) + x + 2\sin\left(\frac{2x+1+1-y}{2}\right) \\ &= 3 - 2y + x + 2\sin\left(x + 1 - \frac{y}{2}\right) \end{aligned}$$

Where x is the amount of time the last robot to reach point Z spends along the perimeter and y is the distance the robots start apart from each other. The first robot takes $1 - y$ time to reach Z and the second robot takes $1 + 2(1 - y)$ time to reach Z. Using Lemma A.1 the max run time is then

given by:

$$\frac{(1-y)+(1+2(1-y))}{2}+\frac{2\pi}{3}+\sqrt{3}=2-\frac{3y}{2}+\frac{2\pi}{3}+\sqrt{3} \qquad (2)$$

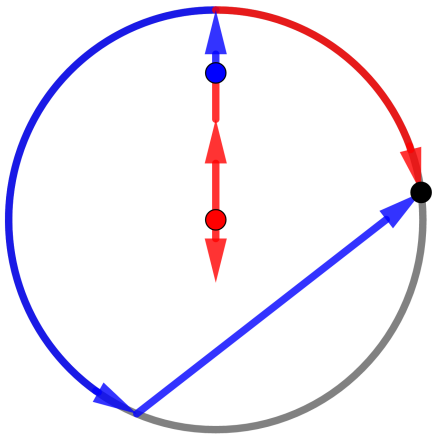

Figure 3: An example of a worst case run time for when $y = 0.7$. The blue and red dots are the starting positions of the robots r_1 and r_2 respectively, and the red and blue arrows indicate the paths taken. The black dot is the exit to the disk.

■

3.2 Comparison of Algorithm A and Algorithm B

Theorem 3.3 *For any $y > \frac{2}{3}$ Algorithm B has a smaller worst-case run time than Algorithm A.*

Proof. This can be shown by taking equations (2) and (1) which then can be put in the inequality (2) $<$ (1) and simplified to imply $y > \frac{2}{3}$:

$$
\begin{aligned}
2 - \frac{3y}{2} + \frac{2\pi}{3} + \sqrt{3} &< 1 + \frac{2\pi}{3} + \sqrt{3} \\
2 - \frac{3y}{2} &< 1 \\
\frac{-3y}{2} &< -1 \\
\frac{3y}{2} &> 1 \\
y &> \frac{2}{3}
\end{aligned}
$$

■

3.3 Lower Bound

Theorem 3.4 *The lower bound proof for this problem follows the same proof for when one robot is placed randomly but both robots know the location of the center of the circle. See theorem 2.3.*

The lower bound is then:

$$1 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$$

In the worst-case, any optimal algorithm should take at least this much time for both robots to evacuate.

4 Both Robots at the Same Arbitrary Location

Another evacuation problem to solve is when both robots are placed in the same arbitrary location inside the circle, and they do not know where they are inside the disk. This means they do not know where the shortest path is to the perimeter. For the algorithms described in this section the robots make use of computing the trajectories they are travelling in with respect to each other.

Given both robots start in the same arbitrary location and don't know where they are placed in the disk, two different algorithms have been found that both have the worst-case run time of $2 + \frac{2\pi}{3} + \sqrt{3}$.

4.1 Upper Bound

Theorem 4.1 *Given both robots start in the same arbitrary location and don't know where they are placed in the disk, there is an algorithm where the robots travel in the same direction while searching for the perimeter that has a worst-case run time of* $2 + \frac{2\pi}{3} + \sqrt{3}$.

Proof. First the algorithm will be stated, then the time complexity will be analyzed.

Algorithm C

---

1. The robots pick an arbitrary direction to travel in and both travel in that direction, along a straight line at speed 1.
2. Both robots reach some point Z along the perimeter and r_1 starts exploring the perimeter CCW and r_2 starts exploring the perimeter CW.
3. Once a robot finds the exit of the room, the other robot is notified of the location to the exit and then travels the shortest path to the exit.

Algorithm C requires both robots to travel together in a straight line in an arbitrary direction until they reach the perimeter. The worst-case is when both robots are placed right next to the perimeter of the circle, but not close enough to detect it, and then travel in the opposite direction of the part of the perimeter closest to them. This results in both robots travelling the diameter of the circle before beginning to explore the perimeter. Using Lemma A.1, the worst-case run time of this algorithm is:

$$\frac{2+2}{2} + \frac{2\pi}{3} + \sqrt{3} = 2 + \frac{2\pi}{3} + \sqrt{3}$$

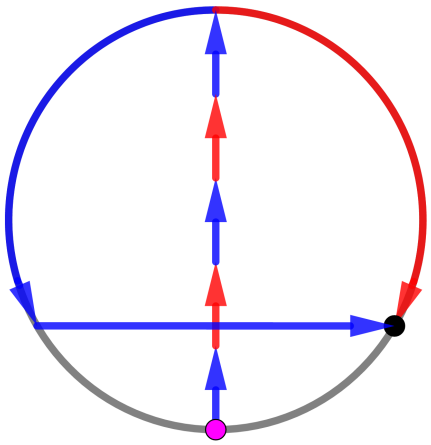

Figure 4: An example of a worst case run time of Algorithm C. The pink coloured dot is the starting positions of the robots, and the red and blue arrows indicate the paths taken by r_2 and r_1 respectively. The black dot is the exit to the disk.

■

Theorem 4.2 *Given both robots start in the same arbitrary location and don't know where they are placed in the disk, there is an algorithm where the robots*

travel in opposite directions while searching for the perimeter that has a worst-case run time of $2 + \frac{2\pi}{3} + \sqrt{3}$.

Proof. First the algorithm will be stated, then the time complexity will be analyzed.

Algorithm D
1. The robots agree upon some arbitrary way to travel in opposite directions away from each other.
2. Robot r_1 reaches some point Z along the perimeter and starts exploring the perimeter CCW and notifies r_2 of the location of Z.
3. r_2 takes the shortest path to Z and starts exploring the perimeter CW once it reaches Z.
4. Once a robot finds the exit of the room, the other robot is notified of the location to the exit and then travels the shortest path to the exit.

Algorithm D requires both robots to travel antipodally until one detects the perimeter at some point Z. The worst-case is when both robots are placed in the ceneter of the circle, resulting in one robot taking time 1 to reach Z and the other robot taking time 3 to reach the point Z. Using Lemma A.1, the worst-case run time of this algorithm is:

$$\frac{1+3}{2} + \frac{2\pi}{3} + \sqrt{3} = 2 + \frac{2\pi}{3} + \sqrt{3}$$

Combining results from unpublished work as of April 2019 by Jarda Opatrny, [6], a better worst case run time of ≈ 5.502 can be given. Jarda shows an algorithm for when the robots both start along the perimeter of the disk which can be used once the robots both reach the perimeter to give a lower upper bound.

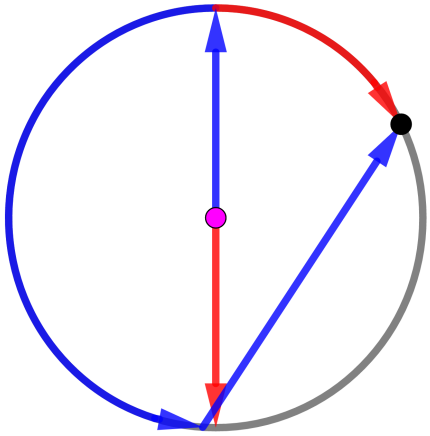

Figure 5: An example of a worst case run time of Algorithm D. The pink coloured dot is the starting positions of the robots, and the red and blue arrows indicate the paths taken by r_2 and r_1 respectively. The black dot is the exit to the disk.

■

4.2 Lower Bound

Theorem 4.3 *It takes at least* $1 + \frac{2\pi}{3} + \sqrt{3}$ *time for the robots to evacuate given they both start in the same arbitrary location and don't know where they are placed in the disk.*

Proof. The proof for the lower bound is similar to the lower bound proof for when one robot is placed at an arbitrary location in the disk and another robot is placed at the center of the disk and they both know where the center of the disk is (see theorem 2.3). The only thing that has to be changed is y must be set to 0. It is important to note that the placement of the robots is at the center of the disk. The worst-case placement, given the structure of the proof used, is when the robots are both placed at the center of the circle since in any other position the robots could reach the

perimeter in a shorter amount of time and yield a lower lower bound.
■

5 Both Robots at Different Arbitrary Locations

Another scenario for the robots to evacuate is when both robots are placed at arbitrary locations in the disk and don't know where they are placed inside the disk. This means they could be placed any amount of distance y from each other such that $0 \leq y \leq 2$.

Given both robots start in arbitrary locations and don't know where they are placed in the disk, there are two algorithms that both give a worst-case run time of $2 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$ where y is the distance the robots start apart from each other.

5.1 Upper Bound

Theorem 5.1 *Given both robots start in arbitrary locations and don't know where they are placed in the disk, there is an algorithm where the robots move in the same direction while searching for the perimeter that has a worst-case run time of* $2 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$ *where y is the distance the robots start apart from each other.*

Proof. First the algorithm will be stated, then the time complexity will be analyzed.

Algorithm E
1. The robots orient themselves by computing the line intersecting their starting positions, and then both travel along that line in the same arbitrary direction towards some unknown point Z on the perimeter. 2. Once robot r_1 reaches Z it starts exploring the perimeter CCW and once robot r_2 reaches Z it starts exploring the perimeter CW. 3. Once a robot finds the exit of the room, the other robot is notified of the location of the exit and then travels the shortest path to the exit.

In this algorithm, the robots decide to travel in one of the two directions on the line intersecting their initial positions. In the worst case, both robots are placed along a line that intersects the disk's center and at least

one robot is placed almost touching the perimeter and picks the direction to travel in so that at least one robot travels distance 2 to reach Z. Worst-case, a robot must travel distance 2 to reach the perimeter and the other $2 - y$. Using Lemma A.1, this then equals a worst-case run time of:

$$\frac{(2-y)+2}{2} + \frac{2\pi}{3} + \sqrt{3} = 2 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$$

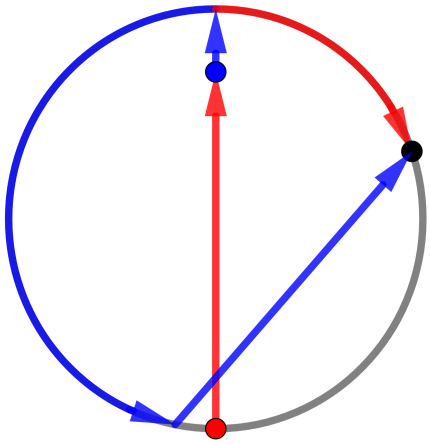

Figure 6: An example of a worst case run time of Algorithm E for when $y = 1.7$. The blue and red dots are the starting positions of the robots r_1 and r_2 respectively, and the blue and red arrows indicate the paths taken. The black dot is the exit to the disk.

■

Theorem 5.2 *Given both robots start in arbitrary locations and don't know where they are placed in the disk, there is an algorithm where the robots move in opposite directions while searching for the perimeter that has a worst-case run time of* $2 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$ *where y is the distance the robots start apart from each other.*

Proof. First the algorithm will be stated, then the time complexity will be analyzed.

Algorithm F
1. The robots orient themselves by computing the line intersecting their starting positions, and then both travel in opposite directions away from each other.
2. Robot r_1 reaches some point Z along the perimeter and starts exploring the perimeter CCW and notifies r_2 of the location of Z.
3. r_2 takes the shortest path to Z and starts exploring the perimeter CW once it reaches Z.
4. Once a robot finds the exit of the room, the other robot is notified of the location to the exit and then travels the shortest path to the exit.

Algorithm F requires both robots move antipodally along the straight line that intersects their initial starting positions. The worst case run time for this algorithm is when both robots are placed along a straight line that intersects the center of the disk and are at equal distance on opposite sides of the center of the disk. Let y be the distance the robots are placed from each other. Using Lemma A.1, the worst-case run time is then given by:

$$\frac{(1-\frac{y}{2})+(2+1-\frac{y}{2})}{2}+\frac{2\pi}{3}+\sqrt{3}=2-\frac{y}{2}+\frac{2\pi}{3}+\sqrt{3}$$

Combining results from unpublished work as of April 2019 by Jarda Opatrny, [6], a better worst case run time of $\approx 5.502-\frac{y}{2}$ can be given. Jarda shows an algorithm for when the robots both start along the perimeter of the disk which can be used once the robots both reach the perimeter to give a lower upper bound.

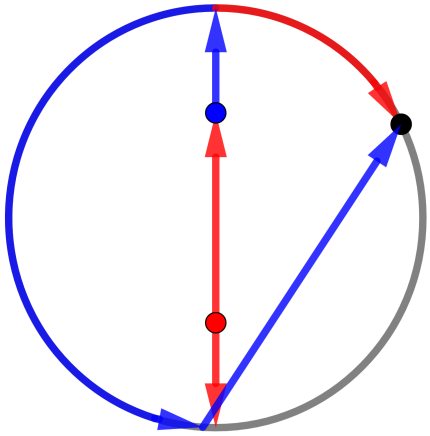

Figure 7: An example of a worst case run time of Algorithm F for when $y = 1$. The blue and red dots are the starting positions of the robots r_1 and r_2 respectively, and the blue and red arrows indicate the paths taken. The black dot is the exit to the disk.

■

5.2 Lower Bound

Theorem 5.3 *Given both robots start in arbitrary locations and don't where they are placed in the disk, it takes at least* $1 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$ *time for the robots to evacuate.*

Proof. The lower bound for this scenario is similar to the first lower bound proof, theorem 2.3, presented in this project. There are a few differences, and the structure of the proof determines the worst-case placement of the robots in terms of their summed shortest paths to the perimeter. Let y be the distance between the two robots' intial placements in the disk. For a given y, the robots' placement that requires the largest summed time for both robots to reach the perimeter is when the robots are placed along a line that intersects the center of the disk, and both robots are not on the same side of the center of the disk. The line drawn from one robots' initial starting position to the other robot's initial starting position must intersect

the center of the disk. This makes it so that the summed distance each robot has to take to reach the perimeter of the disk is at least $a + b = 2 - y$ where a and b are the times each robot takes to reach the perimeter.

There are two cases to consider. First a lower bound for when the exit is found while both robots are on the perimeter will be analyzed. Then it will be shown that a lower bound for when the exit is found while only one robot is on the perimeter can not be more than the lower bound for when the exist is found while both robots are on the perimeter.

Given an algorithm for the robots to evacuate, it takes at least time a for the first robot to reach the perimeter of the disk and time b for the other robot. While both robots are exploring on the perimeter at time $b + x$, at most $2x + b - a$ of the perimeter can be explored. This then means that there would be at least $2\pi - 2x - b + a$ of the perimeter unexplored and $\pi \geq 2\pi - 2x - b + a > 0$. Using theorem 2.2 and setting ϵ to approach 0, there would then be a chord of at least $2\sin(\pi - x - \frac{b-a}{2})$ length at time $b + x$ that has both of its endpoints in an unexplored part of the perimeter. If the exit was located in the end of the chord that is explored last, then one of the robots would have to travel at least the length of that chord to evacuate once the exit is found. This means any given algorithm takes at least $b + x + 2\sin(\pi - x - \frac{b-a}{2})$ time to evacuate both robots.

The lower bound is then the maximum of the above equation. Taking the derivative, the value of x required for the maximum can be found:

$$\frac{d}{dx}\left(b + x + 2\sin\left(x + \frac{b-a}{2}\right)\right) = 0$$
$$1 + 2\cos\left(x + \frac{b-a}{2}\right) = 0$$
$$\cos\left(x + \frac{b-a}{2}\right) = \frac{-1}{2}$$
$$x + \frac{b-a}{2} = \frac{2\pi}{3}$$
$$x = \frac{2\pi}{3} - \frac{b-a}{2}$$

Substituting the value for x in to the original equation gives the lower bound:

$$= b + \frac{2\pi}{3} - \frac{b-a}{2} + 2\sin\left(\frac{2\pi}{3} - \frac{b-a}{2} + \frac{b-a}{2}\right)$$
$$= \frac{a+b}{2} + \frac{2\pi}{3} + \sqrt{3}$$
$$= 1 - \frac{y}{2} + \frac{2\pi}{3} + \sqrt{3}$$

In some cases one of the robots will find the exit to the perimeter before the other robot reaches the perimeter. In the worst-case for this scenario, $a < b$ and $b = 1$ and the perimeter is found right before the second robot to reach the perimeter reaches the perimeter. The second robot would have to travel at most distance 2 to evacuate. This then results in a total run time of $1 + 2 = 3$ to evacuate. No lower bound can be higher than this for this case. This is less than the lower bound for when a robot finds the exit while both of them are on the perimeter and $y = 1$:

$$3 < \frac{1}{2} + \frac{2\pi}{3} + \sqrt{3}$$

So the lower bound first described, $1 - \frac{\gamma}{2} + \frac{2\pi}{3} + \sqrt{3}$, is still currently the highest known lower bound.

■

6 Modified Model

If the robots had some additional rules on how they can travel it can be shown that the two algorithms described are optimal. These rules are the following axioms:

1. Each robot travels at maximum speed 1 at all times
2. Until the perimeter of the disk is found, the robots must travel in straight lines and can not change the direction they travel in unless the perimeter of the disk is found
3. Once a robot reaches the perimeter, it picks either to travel clockwise or counterclockwise along the perimeter and does not change directions
4. The robots most travel in opposite directions to each other along the perimeter of the disk
5. Once the perimeter is found by one of the robots, the other robot must choose to take the shortest path to the point of the perimeter that that was first discovered
6. A robot can only leave the perimeter of the disk if it is notified of the exit to the disk
7. If a robot finds the exit, it must tell the other robot where the exit is and exit the disk, and the other robot must immediately take the shortest path to the exit

6.1 Both Robots at the Same Arbitrary Location

Theorem 6.1 *For covering all of the rules in the modified model, a an optimal algorithm exists which takes at most* $2 + \frac{2\pi}{3} + \sqrt{3}$ *time for when the robots are placed in the same arbitrary location in the disk.*

Proof. The optimal algorithm can be given by describing all possible algorithms given this rules listed above. The algorithms can be described by a single algorithm, which depend on some angle α, where $0 \leq \alpha \leq \frac{\pi}{2}$:

Algorithm CD
1. The robots determine an arbitrary orientation (we can say they always move north for example) and represent their orientation as an "orientation vector" pointing in some direciton. 2. The robots begin travelling at an angle of α and $-\alpha$ to the orientation vector and continue to travel in straight lines to the perimeter of the disk. 3. Robot r_1 reaches some point Z along the perimeter and starts exploring the perimeter CCW and notifies r_2 of the location of Z. 4. r_2 takes the shortest path to Z and starts exploring the perimeter CW once it reaches Z. 5. Once a robot finds the exit of the room, the other robot is notified of the location of the exit and then travels the shortest path to the exit.

Depending on the value of α used, a different worst case placement of the robots is given. Two different types of worst case placements exist: One where the robots are placed right next to the perimeter of the disk and reach the perimeter of the disk at the same time and one where the robots are not right next to the perimeter of the disk and reach the periemter of the disk at the same time and at distance 2 away from each other. A worst case placement requires the robots to travel the furthest combined distance to reach point Z (due to Lemma A.1), so that means the robots would reach the perimeter at the same time and arbitrarily decide which robot should start exploring the perimeter and which one should turn to reach point Z in a worst case placement. The worst case placement for the robots given α can be determined by placing the robots in an arbitrary position in the circle such that the robots would reach the perimeter at the same time and then moving the robots in the opposite direction of the orientation vector until the triangle the robots' paths to reach point Z form has one side length equaling the diameter of the disk or all three vertices of the triangle touch the disks' perimeter. This then leads to two types of worst cases which will be described below.

For when both robots reach the perimeter of the circle at the same time and at distance 2 away from each other the shortest amount of time they can reach the perimeter is in time 1. After reaching the perimeter at the same time, one robot has to double back and reach point Z in time 2. Us-

ing Lemma A.1, this gives a worst case run time of at least:

$$\frac{1+3}{2} + \frac{2\pi}{3} + \sqrt{3} = 2 + \frac{2\pi}{3} + \sqrt{3}$$

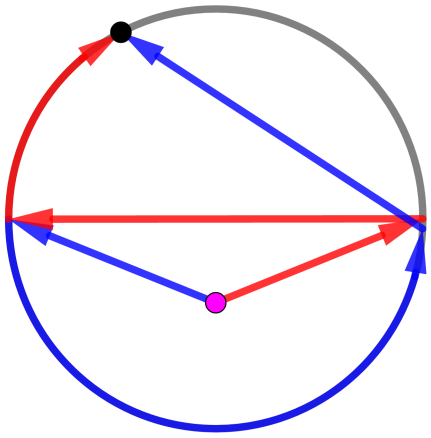

Figure 8: An example of a worst case run time ofAlgorithm CD for when $\alpha = 1.19$. The pink coloured dot is the starting positions of the robots and the blue and red arrows indicate the paths taken. The black dot is the exit of the disk.

The other worst case is when both robots are placed almost touching the perimeter of the circle. Both robots travel and reach the perimeter of the circle at the same time and then one of either r_2 or r_1 travels to reach point Z such that the paths the robots took to reach point Z forms a triangle containing the center of the circle and with all vertices touching the perimeter. Using Lemma C.1 this means the robots would travel at least a combined distance of 4 to each reach point Z. Using Lemma A.1 this would equal a run time of at least:

$$\frac{4}{2} + \frac{2\pi}{3} + \sqrt{3} = 2 + \frac{2\pi}{3} + \sqrt{3}$$

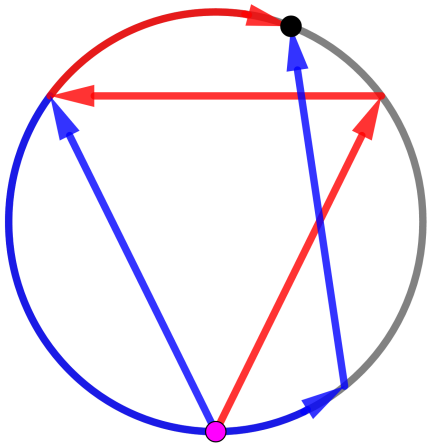

Figure 9: An example of a worst case run time of Algorithm CD for when $\alpha = 0.46$. The pink coloured dot is the starting positions of the robots and the blue and red arrows indicate the paths taken. The black dot is the exit of the disk.

An optimal angle α for Algorithm CD is $\alpha = \frac{\pi}{2}$. This is the same as Algorithm D explained in Theorem 4.2. ■

7 Randomized Algorithm Simulations

Three different robot evacuation scenarios were simulated using python and a library for python called pygame. See Appendix D on the last page of this project for a link to an online repository containing the code. Pygame was used to create an animation of the robots evaucating while the time elapsed was displayed on the screen. Python's math libraries were used to calculate time elapsed at significant moments during the simulation of the algorithms: when the robots reach the perimeter, when one finds the exit and when both robots are done evaucating. This allows the simulations to be checked for accuracy. After each screen refresh in the simulation the time elapsed was updated by a fixed amount and the robots were each moved by a fixed distance. If the calculated time elapsed at significant moments using python's math libraries is the same as the time elapsed calculated by updating the time elapsed incrementally after each screen refresh, then this means the simulation is accurate. No inaccuracy was found in the simulations.

The simulations were created for the main purpose of getting a good idea of what the expected run time of each algorithm should be. The expected run time can precisely be calculated using mathematical proofs. For each of the algorithms simulated, the exit was placed in a random location each time the robots had to evacuate which results in an average run time given any number of 'trials' greater than 0. A trial is a single run of a robot evacuation with the exit placed randomly along the perimeter and the robots randomly placed somewhere in the disk (unless the scenario requires the robots to be placed in a fixed position). All points along the perimeter have an equal probability of being chosen to be where the exit is. If the robots are randomly placed, all points inside the disk have an equal probability of being chosen to be where the robots are placed. The following 3 subsections describe each algorithm that was simulated (along with its scenario) and the assosciated run time given by a 95% confidence interval on 100000 trials. After testing the simulations for inaccuracies, simulations for getting the average run time used only the code which used python's math libraries for calculating the run time for both robots to evacuate. Pygame was not used in the total of 300000 trials to save com-

putational time for rendering the robots.

A mathematical proof will also be presented for each simulation, showing what the precise expected value should be. It was found that the simulations and mathematical proofs yielded the same expected run times. Each algorithm can be treated as a continuous random variable X. Taking the definite integral of the probability density function $g(x)$ times the value of the random variable $f(x)$ both as functions of x for all possible values of x will give the expected value of the random variable X. More formally, it can be written as:

$$E(X) = \int_{\min_x}^{\max_x} g(x)f(x)dx$$

A probability density function can be thought of as a function which describes the likelihood of a certain event happening given the value of x relative to all other values of x. A probability density function should always have a definite integral equal to 1 for all possible values of x.

7.1 Both Robots at the Center, the Robots Know the Center of the Disk

Algorithm C, as was explained in theorem 4.1, was simulated for the scenario in which the robots start at the center of the disk and know they are at the center of the disk. This is an optimal worst-case algorithm [1]. The 'arbitrary' orientation (as described in the first step of Algorithm C) that was simulated was for the robots to always travel in the y axis, and towards the 'top' of the disk as illustrated in figure 4. For example, in a real-world setting this could mean the robots always move north, although this doesn't affect the run time. With 100000 trials and a 95% confidence interval, a run time of 3.8459 ± 0.0068 was given.

Theorem 7.1 *There exists an algorithm for when the robots start at the center of the disk and know the center of the disk that has an expected run time of* $1 + \frac{\pi}{2} + \frac{4}{\pi} \approx 3.84$.

Proof. The expected run time for the algorithm explained in theorem 4.1 for this scenario is entirely dependent on where the exit is randomly placed, since the robots start in the same position every time. The exit has a uniform probability for being placed anywhere along the perimeter. The run time of the algorithm is symmetric such that if the exit is place at an angle of x along the perimeter with respect to where the robots first reach the perimeter, then the same run time will be given for if the exit was placed at an angle of $-x$. For simplicity, only values of $0 \leq x \leq \pi$ will be considered which will still give the same expected run time due to symmetry. The random variable in this case is the run time of the algorithm, who's value can be expressed as the function $1 + x + 2\sin(\frac{x}{2})$ where x is the amount of time the robots spend along the perimeter before one of them finds the exit. The probability density function represents the relative probability for each possible value of x in the range of $0 \leq x \leq \pi$. The probability density function represents a uniform distribution, so it should then be $g(x) = \frac{1}{\pi}$ since $\int_0^\pi \frac{1}{\pi} dx = 1$.

The expected run time can be given by the following definite integral:

$$\begin{aligned}
E(X) &= \int_0^\pi \frac{1}{\pi}\left(1 + x + 2\sin\left(\frac{x}{2}\right)\right) dx \\
&= \frac{1}{\pi}\left(x + x^2 - 4\cos\left(\frac{x}{2}\right)\right)\Big|_0^\pi \\
&= \frac{1}{\pi}\left(\left(\pi + \frac{\pi^2}{2} - 4\cos\left(\frac{\pi}{2}\right)\right) - \left(0 + \frac{0^2}{2} - 4\cos\left(\frac{0}{2}\right)\right)\right) \\
&= 1 + \frac{\pi}{2} + \frac{4}{\pi}
\end{aligned}$$

■

7.2 Both Robots at the Same Random Location, the Robots Don't Know the Center of the Disk

The same algorithm as referenced in the beginning of section 7.1 was simulated but for the robots starting at a random location inside the disk. Algorithm C was used and with the robots always travlling in the same direction to reach the perimeter. For simplicity, this can be seen as the robots

orienting themselves by always travelling 'north' or along the y axis and in the positive direction to reach the perimeter. With 100000 trials and a 95% confidence interval, a run time of 3.6887 ± 0.0076 was given.

Theorem 7.2 *There exists an algorithm for when the robots both start at a random location in the disk and don't know the center of the disk that has an expected run time of* $\frac{8}{3\pi} + \frac{\pi}{2} + \frac{4}{\pi} \approx 3.69$.

Proof. The expected run time for Algorithm C in this scenario can be broken up into two expectations due to linearity of expectation. One for the expected amunt of time for the robots to reach the perimeter, and one for the expected amount of time for the robots to evacuate once they have reached the perimeter. Adding these two expectations together will give the expected time of Algorithm C. The proof for the expected amount of time the robots need to evacuate once they have reached the perimeter follows the same proof in theorem 7.1 but with the time 1 for the robots to reach the perimeter subtracted. Where x is the distance along the perimeter the robots have to travel before finding the exit, this then gives:

$$\begin{aligned} E(X) &= \int_0^{\pi} \frac{1}{\pi}\left(x + 2\sin\left(\frac{x}{2}\right)\right) dx \\ &= \frac{1}{\pi}\left(x^2 - 4\cos\left(\frac{x}{2}\right)\right)\Big|_0^{\pi} \\ &= \frac{1}{\pi}\left(\left(\frac{\pi^2}{2} - 4\cos\left(\frac{\pi}{2}\right)\right) - \left(\frac{0^2}{2} - 4\cos\left(\frac{0}{2}\right)\right)\right) \\ &= \frac{\pi}{2} + \frac{4}{\pi} \end{aligned} \tag{3}$$

The expected amount of time for the robots to reach the perimeter depends on where they are placed inside the disk. To model the problem better for analyzing the expected run time using a continuous random variable, the center of the disk will be treated as the coordinate (0,0) on the cartesian plane. The robots then always travel in the positive direction along the y-axis to reach the perimeter. The probability density function can then represent the relative likelihood for all possible values of the x

coordinate for the robots' random placement. Each unique x coordinate can be thought of as belonging to a chord of the disk which is parallel to the y axis and the length of each of these chords can be given by the equation $2\sqrt{1-x^2}$. The probability density function can then be expressed as $g(x) = \frac{2\sqrt{1-x^2}}{\pi}$ since $\int_{-1}^{1} \frac{2\sqrt{1-x^2}}{\pi} dx = 1$. Given the robots are placed along a chord parallel to the y axis, it is easy to see that the expected amount of time for the robots to reach the perimeter is half the length of the chord. The value for the random variable can then be $f(x) = \sqrt{1-x^2}$. The expected time for the robots to reach the perimeter of the disk can be described by the following definite integral:

$$\begin{aligned}
E(X) &= \int_{-1}^{1} \frac{2\sqrt{1-x^2}}{\pi} \left(\sqrt{1-x^2}\right) dx \\
&= \int_{-1}^{1} \frac{2-2x^2}{\pi} dx \\
&= \frac{2x - \frac{2x^3}{3}}{\pi}\Big|_{-1}^{1} \\
&= \frac{2(1) - \frac{2(1^3)}{3}}{\pi} - \left(\frac{2(-1) - \frac{2(-1^3)}{3}}{\pi}\right) \\
&= \frac{4 - \frac{4}{3}}{\pi} \\
&= \frac{8}{3\pi}
\end{aligned} \tag{4}$$

Adding equations (4) and (3) together gives the expected run time of Algorithm C, for when the robots start at the center of the disk and don't know the center of the disk:

$$\frac{8}{3\pi} + \frac{\pi}{2} + \frac{4}{\pi} \approx 3.69$$

■

7.3 Both Robots at the Same Random Location, the Robots Know the Center of the Disk

There is a known optimal evacuation algorithm, which is unpublished as of April 2019 [7], for the scenario of two robots starting in the same arbitrary position in the disk and they both know where the center of the disk is. The algorithm was simulated and is as follows:

Algorithm G

1. The robots calculate and take the shortest path to the perimeter.
2. Both robots reach some point Z along the perimeter and r_1 starts exploring the perimeter CCW and r_2 starts exploring the perimeter CW.
3. Once a robot finds the exit of the room, the other robot is notified of the location to the exit and then travels the shortest path to the exit.

With 100000 trials and a 95% confidence interval, a run time of 3.1819 ± 0.0069 was given.

Theorem 7.3 *There exists an algorithm, Algorithm G, for when the robots start at the center of the disk and know the center of the disk that has an expected run time of* $\frac{1}{3} + \frac{\pi}{2} + \frac{4}{\pi} \approx 3.18$.

Proof. The expected run time for Algorithm G in this scenario can be broken up into two expectations due to linearity of expectation. One for the expected amunt of time for the robots to reach the perimeter, and one for the expected amount of time for the robots to evacuate once they have reached the perimeter. Adding these two expectations together will give the expected time of Algorithm G. The proof for the expected amount of time the robots need to evacuate once they have reached the perimeter follows the same proof in theorem 7.2 (see equation (3) in theorem 7.2).

The expected amount of time for the robots to reach the perimeter depends on where they are placed inside the disk. The robots are more likely to be placed near the perimeter of the disk than they are near the center, since there is more surface area closer to the perimeter than there is closer to the center. For example, a robot is twice as likely to be placed at distance 0.8 from the center of the disk than it is at distance 0.4 since the circumfirence of a circle increases linearly as a function of the length of the radius. Circumference $= 2\pi r$. The probability density function can be $g(x) = \frac{2\pi x}{\pi} = 2x$ since $\int_0^1 2x dx = 1$. This probability density function $g(x) = 2x$ represents the relative likelihood between all random placements of the robots in the disk where x is the distance the robots are from the center of the disk. The value given to the random variable X in this case would be the distance the robots are from the perimeter, which can be expressed as $f(x) = 1 - x$. The expected time for the robots to reach the perimeter of the disk can be described by the following definite integral:

$$\begin{aligned} E(X) &= \int_0^1 2x\,(1-x)\,dx \\ &= \int_0^1 2x - 2x^2 dx \\ &= x^2 - \frac{2x^3}{3}\Big|_0^1 \\ &= 1^2 - \frac{2(1^3)}{3} - \left(0^2 - \frac{2(0^3)}{3}\right) \\ &= 1 - \frac{2}{3} \\ &= \frac{1}{3} \end{aligned} \tag{5}$$

Adding equations (5) and (1) together gives the expected run time of Algorithm G, for when the robots start at the center of the disk and know the center of the disk:

$$\frac{1}{3} + \frac{\pi}{2} + \frac{4}{\pi} \approx 3.18$$

■

8 Conclusion

It has been shown that algorithms for multiple variations of the problem were all proven to be at-most 1 unit of time off from optimal solutions. It seems to intuitively make sense that an optimal evacuation algorithm for when both robots start at arbitrary locations and don't know where they are placed inside the disk should follow the second axiom listed in Section 6. It is: "Until the perimeter of the disk is found, the robots must travel in straight lines and can not change the direction they travel in unless the perimeter of the disk is found". If this were the case, then the problem would be able to be reduced to the sub-problem for evacuating two robots where they both start on the perimeter of the disk assuming that in the worst case both robots reach the perimeter at the same time. Further work focusing on proving or disproving this is recommended. Methods for calculating the expected run time of algorithms were also shown, which could be good to understand for working on variations of the evacuation problem which may possibly be studied in the future which involve expected run times.

References

[1] Jurek Czyzowicz, Leszek Gąsieniec, Thomas Gorry, Evangelos Kranakis, Russell Martin, and Dominik Pajak. Evacuating robots via unknown exit in a disk. In *International Symposium on Distributed Computing*, pages 122–136. Springer, 2014.

[2] Jurek Czyzowicz, Konstantinos Georgiou, Evangelos Kranakis, Lata Narayanan, Jaroslav Opatrny, and Birgit Vogtenhuber. Evacuating robots from a disk using face-to-face communication. In *International Conference on Algorithms and Complexity*, pages 140–152. Springer, 2015.

[3] Jurek Czyzowicz, Konstantinos Georgiou, Maxime Godon, Evangelos Kranakis, Danny Krizanc, Wojciech Rytter, and Michał Włodarczyk. Evacuation from a disc in the presence of a faulty robot. In *International Colloquium on Structural Information and Communication Complexity*, pages 158–173. Springer, 2017.

[4] Jurek Czyzowicz, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, and S Shende. Wireless autonomous robot evacuation from equilateral triangles and squares. In *International Conference on Ad-Hoc Networks and Wireless*, pages 181–194. Springer, 2015.

[5] Jurek Czyzowicz, Konstantinos Georgiou, Ryan Killick, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, and Sunil Shende. Priority evacuation from a disk using mobile robots. In *International Colloquium on Structural Information and Communication Complexity*, pages 392–407. Springer, 2018.

[6] Jarda Opatrny. Two agents evacuating from a disk starting from two arbitrary points along the perimeter, the wireless communication model. In *Unpublished as of April 2019*.

[7] Evangelos Kranakis. Optimal evacuation of two robots from a disk in the wireless model. In *Unpublished as of April 2019*.

A Lemma A.1

Lemma A.1 *There exists a class of algorithms where the the two robots both start exploring the perimeter of the disk in opposite directions at some point Z and get to point Z in arbitrary amounts of time. Additionally, if the first robot to reach point Z finds the exit before the other robot reaches point Z, the second robot to reach point Z takes the shortest path to the exit instead of reaching Z. Given each of the two robots get to Z in arbitrary amounts of time a and b respectively where* $a \leq b \leq 3.65$*, and the last robot to reach Z takes the optimal path to Z right when Z is reached by the first robot, then the optimal runtime for this class of algorithms is* $\frac{a+b}{2} + \frac{2\pi}{3} + \sqrt{3}$.

Proof. There are two possible cases that can occur for this class of algorithms: The exit is found and one robot hasn't reached Z, or the exit is found after both robots have reached point Z. It will be shown that there is a known optimal algorithm for the case when the exit is found and both robots are on the perimeter of the circle (they have reached point Z). It will also be shown that the lower bound for the case when the exit is found and one robot hasn't reached Z can not have a greater run time than the optimal algorithm for the case for when the exit is found while both robots have reached Z. The first case that will be shown is when the exit is found and both robots have reached Z.

The robots reach the point Z on the perimeter of the disk at different times, in a time for one robot and b time for the other robot. There exists an algorithm for the robots to explore the perimeter of the disk in opposite directions and at speed 1 as soon as each robot reaches point Z which will give the above worst-case runtime:

Algorithm H
1. The robots travel in some arbitrary manner, and after time a, robot r_1 reaches some point Z on the perimeter of the disk. 2. Once robot r_1 reaches point Z along the perimeter, it starts exploring the perimeter CCW and notifies r_2 of the location of Z. 3. r_2 takes the shortest path to reach Z in time b and starts exploring the perimeter CW once it reaches Z. If r_1 finds the exit before r_2 reaches Z, then r_2 takes the shortest path to the exit instead of having to reach Z first. 4. Once a robot finds the exit of the room, the other robot is notified of the location of the exit and then travels the shortest path to the exit.

Assuming the exit is found after both robots have reached Z, the run time can be given by the time it takes for the last robot to reach the perimeter, plus the amount of time that robot spends on the perimeter, plus the length of the chord between the two robots when the exit is found. Note that since one robot always explores $b - a$ more of the perimeter than the other robot while they are both on the perimeter, $b - a$ must be added to the length of the arc the two robots form.

The run time of the algorithm in this case is:

$$f(x) = b + x + 2\sin\left(\frac{2x + (b-a)}{2}\right) = b + x + 2\sin\left(x + \frac{b-a}{2}\right)$$

The worst-case is the maximum of the above equation. Taking the derivative, the value of x required for the maximum can be found:

$$\begin{aligned}
\frac{d}{dx}\left(b + x + 2\sin\left(x + \frac{b-a}{2}\right)\right) &= 0 \\
1 + 2\cos\left(x + \frac{b-a}{2}\right) &= 0 \\
\cos\left(x + \frac{b-a}{2}\right) &= \frac{-1}{2} \\
x + \frac{b-a}{2} &= \frac{2\pi}{3} \\
x &= \frac{2\pi}{3} - \frac{b-a}{2}
\end{aligned}$$

Substituting the value for x in to the original equation gives the lower bound:

$$\begin{aligned}
&= b + \frac{2\pi}{3} - \frac{b-a}{2} + 2\sin\left(\frac{2\pi}{3}\right) \\
&= \frac{2b}{2} - \frac{b-a}{2} + \frac{2\pi}{3} + \sqrt{3} \\
&= \frac{a+b}{2} + \frac{2\pi}{3} + \sqrt{3}
\end{aligned}$$

The proof for it being the lowest possible worst-case follows the same lower bound proof first shown in this project (see theorem 2.3) but with a slight alteration. The proof is as follows:

It takes time b for both robots to have reached the point Z of the circle. While both robots are exploring on the perimeter at time $b + x$, at most $2x + b - a$ of the perimeter can be explored. This then means that there would be at least $2\pi - 2x - b + a$ of the perimeter unexplored and $\pi \geq 2\pi - 2x - b + a > 0$. Using theorem 2.2 and setting ϵ to approach 0, there would then be a chord of at least $2\sin(\pi - x - \frac{b+a}{2})$ length at time

$b + x$ that has both of its endpoints in an unexplored part of the perimeter. If the exit was located in the end of the chord that is explored last, then one of the robots would have to travel at least the length of that chord to evacuate once the exit is found. This means any given algorithm takes at least $b + x + 2\sin(\pi - x - \frac{b+a}{2})$ time to evacuate both robots.

The run time of Algorithm H previously described is the same as the lower bound so it an optimal algorithm. It is important to note is that it is an optimal algorithm given the restriction that the robots each go to point Z in time a and b and the exit is found after both robots have reached point Z.

The run time for the one other case where one robot finds the exit to the disk before both robots have reached the perimeter can happen only if $a \neq b$. The second robot takes the shortest and optimal path to Z once it is found, the highest lower bound (an optimal algorithm) for this case must be less than equal to the time it takes for the second robot to reach Z plus the length of the chord from Z to the exit. A simple proof can show this: the second robot takes the shortest path to the exit as soon as it is found, this shortest path may or may not include passing through the point Z. The worst case can be given by the following equation:

$$b + 2\sin\left(\frac{\pi}{2}\right) = b + 2$$

The largest a chord can be is length 2. So $b + 2$ is given. Now it must be shown this is less than the optimal algorithm for the case when both robots are on the perimeter and the exit is found.

$$
\begin{aligned}
b+2 &< \frac{a+b}{2} + \frac{2\pi}{3} + \sqrt{3} \\
b+2 &< \frac{b}{2} + \frac{2\pi}{3} + \sqrt{3} \\
\frac{b}{2} &< \frac{2\pi}{3} + \sqrt{3} - 2 \\
b &< \frac{4\pi}{3} + 2\sqrt{3} - 4 \\
b &< 3.65
\end{aligned}
$$

■

B Lemma B.1

Lemma B.1 *Given a chord of length c_j and its assosciated arc of length $a_j \leq \pi$ in a circle of radius 1 then $\frac{c_j}{a_j} \geq \frac{2}{\pi}$.*

Proof. This can be proven by showing that for all values of $0 < x \leq \pi$ the fraction $\frac{2\sin(\frac{x}{2})}{x}$ where $2\sin(\frac{x}{2}) = c_j$ and $a_j = x$ is greater than or equal to $\frac{2}{\pi}$. Taking the derivative of $\frac{2\sin(\frac{x}{2})}{x}$ gives $\frac{x\cos(\frac{x}{2}) - 2\sin(\frac{x}{2})}{x^2}$. Only in the interval $0 < x \leq \pi$ if it is true that $\frac{x\cos(\frac{x}{2}) - 2\sin(\frac{x}{2})}{x^2} < 0$ then $\frac{2\sin(\frac{x}{2})}{x}$ will have only one minimum when $x = \pi$.

It is easy to see $x^2 > 0$ in the interval $0 < x \leq \pi$. So if in the interval $0 < x \leq \pi$ it is the case that $x\cos(\frac{x}{2}) - 2\sin(\frac{x}{2}) < 0$ then $\frac{x\cos(\frac{x}{2}) - 2\sin(\frac{x}{2})}{x^2} < 0$.

Subsituting $x = 0$ into $x\cos(\frac{x}{2}) - 2\sin(\frac{x}{2})$ gives $0\cos(\frac{0}{2}) - 2\sin(\frac{0}{2}) = 0$. For in the in the interval $0 < x \leq \pi$, if it can be shown that the derivative of $x\cos(\frac{x}{2}) - 2\sin(\frac{x}{2}) < 0$ then it will follow that $x\cos(\frac{x}{2}) - 2\sin(\frac{x}{2}) < 0$. Taking the derivative of $x\cos(\frac{x}{2}) - 2\sin(\frac{x}{2})$ gives $\cos(\frac{x}{2}) - \frac{x\sin(\frac{x}{2})}{2} + \cos(\frac{x}{2}) = -\frac{x\sin(\frac{x}{2})}{2}$. Looking only in the interval $0 < x \leq \pi$, it can be seen

that $-\frac{x\sin(\frac{x}{2})}{2} < 0$ since $\sin(x) > 0$, and applying a horizontal stretch by a factor of 2 gives $\sin(\frac{x}{2}) > 0$, followed by a vertical compression by a factor of 2 gives $\frac{\sin(\frac{x}{2})}{2} > 0$. Finally, multiplying by $-x$ makes $-\frac{x\sin(\frac{x}{2})}{2} < 0$ in the interval $0 < x \leq \pi$.

Therefore proving that for all values of $0 < x \leq \pi$ the equation $\frac{2\sin(\frac{x}{2})}{x}$ has a minimum at $x = \pi$ and as a result $\frac{c_j}{a_j} \geq \frac{2}{\pi}$ since $\frac{2\sin(\frac{\pi}{2})}{\pi} = \frac{2}{\pi}$. ■

C Lemma C.1

Lemma C.1 *Any polygon who's vertices touch the perimeter of a circle with radius 1 and contains the center of the circle will have at least a perimeter of 4.*

Proof. Each side of the polygon contained by the circle will also be a chord of the circle. The entire circle's perimeter should then be composed of all the arcs assosciated with each side of the polygon.

Due to Lemma B.1, it can be shown that a polygon with i sides that contains the center of the circle and has vertices touching the perimeter of the circle will have at least a perimeter of 4. Each chord of length c_j and its assosciated arc of length a_j gives $\frac{c_j}{a_j} \geq \frac{2}{\pi}$ which then gives $c_j \geq \frac{2}{\pi}a_j$. Summing all chord lengths gives the following:

$$\begin{aligned} c_1 + c_2 + \cdots + c_i &\geq \frac{2}{\pi}a_1 + \frac{2}{\pi}a_2 + \cdots + \frac{2}{\pi}a_i \\ &= \frac{2}{\pi}(a_1 + a_2 + \cdots + a_i) \\ &= \frac{2}{\pi}(2\pi) \\ &= 4 \end{aligned}$$

Thus $c_1 + c_2 + \cdots + c_i \geq 4$, proving the Lemma. ■

D Code For Simulations

Click here for a link to an online repository containing all the code used for simulating the randomized algorithms.

If that doesn't work, copy and paste this into a browser:
https://github.com/NickPerezCarletonUniversity/SimulatingEvacuatingTwoRobotsInADiskViaUnknownExitWithTheRobotsStartingAtArbitraryLocations