Evacuation of Equilateral Triangles by Mobile Agents of Limited Communication Range

Iman Bagheri

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Computer Science at
Concordia University
Montréal, Québec, Canada

December 2019

Concordia University
School of Graduate Studies

This is to certify that the thesis prepared

By: **Iman Bagheri**

Entitled: **Evacuation of Equilateral Triangles by Mobile Agents of Limited Communication Range**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

______________________ Chair
Dr. Aiman Hanna

______________________ Examiner
Dr. Thomas Fevens

______________________ Examiner
Dr. Denis Pankratov

______________________ Supervisor
Dr. Lata Narayanan

Approved by ______________________
Dr. Leila Kosseim
Graduate Program Director

December 10, 2019 ______________________
Dr. Amir Asif, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Evacuation of Equilateral Triangles by Mobile Agents of Limited Communication Range

Iman Bagheri

We consider the problem of evacuating $k \geq 2$ mobile agents from a unit-sided equilateral triangle through an exit located at an unknown location on the perimeter of the triangle. The agents are initially located at the centroid of the triangle. An agent can move at speed at most one, and finds the exit only when it reaches the point where the exit is located. The agents can collaborate in the search for the exit. The goal of the *evacuation problem* is to minimize the evacuation time, defined as the worst-case time for *all* the agents to reach the exit.

Two models of communication between agents have been studied before; *non-wireless* or *face-to-face communication* model and *wireless communication* model. In the former model, agents can exchange information about the location of the exit only if they are at the same point at the same time, whereas in the latter model, the agents can send and receive information about the exit at any time regardless of their positions in the domain. In this thesis, we propose a new and more realistic communication model: agents can communicate with other agents at distance at most r with $0 \leq r \leq 1$.

We propose and analyze several algorithms for the problem of evacuation by $k \geq 2$ agents in this model; our results indicate that the best strategy to be used varies depending on the values of r and k. For two agents, we give five strategies, the last of which achieves the best performance among all the five strategies for all sub-ranges of r in the range $0 < r \leq 1$. We also show a lower bound on the evacuation time of two agents for any $r < 0.336$. For $k > 2$ agents, we study three strategies for evacuation: in the first strategy, called X3C, agents explore all three sides of the triangle before connecting to exchange information; in the second strategy, called X1C, agents explore a single side of the triangle before connecting; in the third strategy, called CXP, the

agents travel to the perimeter to locations in which they are connected, and explore it while always staying connected. For 3 or 4 agents, we show that X3C works better than X1C for small values of r, while X1C works better for larger values of r. Finally, we show that for any r, evacuation of $k = 6 + 2\lceil(\frac{1}{r} - 1)\rceil$ agents can be done using the CXP strategy in time $1 + \sqrt{3}/3$, which is optimal in terms of time, and asymptotically optimal in terms of the number of agents.

Acknowledgments

First and foremost, I thank my parents for their unconditional love and support throughout my life. You are the ones that I share my happiness in joyful days, and the shoulders that I lean on when things get rough. This thesis is, of course, dedicated to you.

I am highly indebted to my supervisor, Professor Lata Narayanan, as this thesis would have not been completed without her exceptional guidance and charismatic presence. Also, my utmost thanks to Professor Jaroslav Opatrny for his many contributions to this project, including our co-authored publication [4].

My special thanks to Concordia university for providing me with an extraordinary academic experience together with the necessary funding to complete my studies.

I also want to express my gratitude toward my toughest undergraduate faculty member at Shahid Beheshti University, Dr. Ghavameddin Razavizadeh. It was only years after when I realized that his toughness came from his love and care for his students. He not only taught us the most difficult subject in my field, but also was the best exemplar of professional ethics.

I wish to thank my friend Majid, who never missed any chance to help people around him. I would also like to express my heartfelt gratitude to my sister and brother, who have continuously encouraged me to pursue this academic journey. I consider myself very lucky to be surrounded by my loving and supporting friends, Dr. Saam Iranmanesh, Dr. Mohammad Asghari and Hamid Reza Behizadi, who have never given up on supporting me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In Computer Science, *search theory* tackles the problem in which a *Searcher* with a limited amount of information about the environment wishes to minimize the time required to find a hidden prize [3]. Traditionally the hidden item was considered stationary, meaning an adversary chooses a location for the item before the execution of the algorithm. The best known example of these types of problems is the *Cow Path Problem* [6], in which a cow (or an agent) searches an infinite fence represented by a line, in order to find a hole, the hidden prize. An extended version of the cow path problem is proposed in [29] where the prize is hidden in a star Q consisting of M unbounded rays originating from a single point O, as shown in Figure 1.

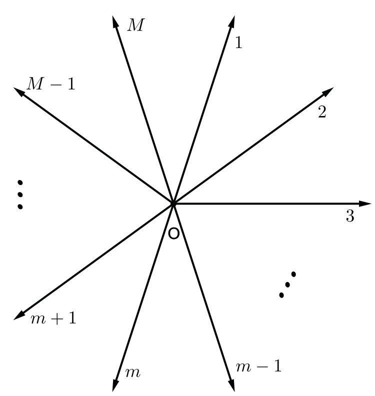


Figure 1: Star search proposed in [29]

Another classical problem is the *The Princess and the Monster* problem proposed in [31]. In this problem the Princess and the Monster are both in a dark room and the Princess is aware of the fixed speed of the Monster although she can move with arbitrary speed and she tries to avoid the Monster. This problem has some major differences with the *cow-path* problem; in addition to the searching domain being a two-dimensional region vs. a single line in cow-path problem, the motivation of this algorithm is for the Princess to avoid being caught by the Monster.

Another search problem, called *The Flaming Datum*, is studied in [11] which has military applications. Suppose at time 0 a submarine has fired a torpedo at a ship, hence revealing its location. The ships sends out a distress call and a sonar carrying helicopter arrives on the scene at time t. Unaware of the trajectory of the submarine, the helicopter should use its sonar to locate the submarine. The advantage that the submarine has is that up to time t, it can move in an arbitrary direction unknown to the helicopter. On the other hand, the speed of the helicopter works in its favor.

A subcategory of search algorithms is *group search* algorithms in which the number of searchers is greater than one. In this class, the search environment is divided into several parts and then each part is assigned to a searcher so that they collaboratively explore the whole area. In this case, the execution time of the algorithm with k searchers can potentially be reduced up to $1/k$ of the time needed for the algorithm with only one agent, though getting the improvement of $1/k$ is rarely achieved in real scenarios.

Evacuation Problems can be seen as another type of the search problems. We will first give a definition of the the evacuation problem in the following section, and later on, will determine the specific problem that is going to be studied in this thesis.

1.1 Evacuation Problem

Consider a situation in which multiple agents are trapped in an enclosed domain. The agents are aware of the existence of a single or multiple exit points but the whereabouts of these points are unknown to the agents. Their initial position and the domain is also known to the agents. The goal is to *evacuate* all the agents from this domain through those exit points in the shortest time possible because of unfortunate circumstances e.g. earthquake or fire. The agents have the ability to communicate with each other and to send the location of the found exit points to the other agents. This problem is known as *Evacuation Problem*. The main difference between evacuation and group search problems is that in the search problem, the execution of the algorithm is finished once the target is found by *any* of the agents, whereas in evacuation problems, the algorithm is finished when *all* the agents or some *specific agents* exit the domain.

To describe an evacuation problem, we need to specify the following aspects of the model [19]:

Search Domain: The search domain is the perimeter of an enclosed area in which all the agents are trapped in it. There exists one or more exit points located on the boundary, through which the agents can exit from that area. The domains that have been studied so far are 2-D geometric shapes such as disk, triangle and square.

Agents and Their Capabilities: The agents are assumed to know the geometry of the search domain, they have the same sence of orientation, and they know each others trajectories. They can move with at most unit speed. In terms of computational power, both finite-automata agents, and Turing-machine agents have been studied. In the latter model, agents can, for example, calculate the shortest path to a point in which they can communicate with another agent in negligible time, if it is necessary. An agent can recognize the exit only when it reaches it. We may assume that all agents are reliable, or that some fraction are faulty; both crash-faulty and Byzantine-faulty agents have been studied. Finally, agents are capable of communicating with each other, using a specified communication model. We describe below the communication models that have been studied so far.

The main distinction between the evacuation problems is the communication method of the agents. Agents in the *face-to-face* model studied in [13, 17, 10] can only communicate if they are at the same point at the same time. In contrast, agents in the *wireless* model studied in [19, 20] are capable of sending a message to any location inside the domain at any time. Both these models are unrealistic; in practice agents are likely to communicate using a wireless transmitter/receiver with limited communication range. In this thesis we introduce a new model, where agents are capable of communicating with each other, *only if they are close enough to each other.*

In general, an evacuation algorithm consists of segments of a line or a curve that each agent traverses. The sequence of these segments for each agent is called a *trajectory* for the agent. To specify an evacuation algorithm, first we need to specify these trajectories for each agent. The trajectories are defined at time $t = 0$ and typically all the agents are aware of the trajectories of the other agents, and they typically know what to do if they find the exit or they get notified about the location of the exit by another agent.

1.2 Problem Definition

In this thesis we consider $k \geq 2$ non-faulty agents that are initially located at the centroid of an equilateral triangle with side 1. Each agent has a wireless transmitter/receiver and they can communicate with another agent only when their distance is at most r. The objective of the algorithm is to evacuate all the agents through the single exit point located on the boundary of this triangle. Our goal is to minimize the time needed for all agents to exit through the exit point. At time $t = 0$ the agents start to follow their assigned trajectories with maximum speed 1 and they find the exit only when they reach it, meaning the exit cannot be recognized from some distance.

1.3 Preliminaries

We denote the triangle by T, with O as centroid and A, B and C as vertices. For any segment on the plane connecting point P to Q, we denote it with PQ and its length by $|PQ|$. The *height* of the triangle is denoted by h and $y = h/3$. We denote the time that point x is seen for the first time by any of the agents by t_x.

By $E_{\mathcal{A}}(k, r, x)$ we mean the evacuation time of algorithm $\mathcal{A}$ with k agents, $k \geq 2$ and communication range of r, $0 \leq r \leq 1$, if the exit is located at x. The worst-case evacuation time of algorithm $\mathcal{A}$ with k agents and communication range of r is given by $E_{\mathcal{A}}(k, r)$, in other words:

$$E_{\mathcal{A}}(k, r) = \sup_{x \in \mathcal{P}(T)} \left(E_{\mathcal{A}}(k, r, x)\right)$$

where $\mathcal{P}(T)$ is the perimeter of T. We denote the optimal evacuation time by k agents by $E^*(k, r)$, that is:

$$E^*(k, r) = \inf_{\mathcal{A}} E_{\mathcal{A}}(k, r)$$

Critical Points

In our algorithms, we divide the exploration trajectory of agents into few segments. For each segment, we try to find the point that has the maximum evacuation time, we call these points *critical points*. In order to minimize the maximum evacuation time, after identifying these critical points in the triangle, we introduce algorithms to decrease the evacuation time for these points.

Extended Meeting Points

Due to the ability of transmitting a message, one agent can place itself between two other agents, in order to act as a relay and enable the two other agents to communicate. We call these positions *extended meeting points*. More formally if we consider the positions of all the agents as $P_1,\ P_2,\ ...\ P_k$, those positions can be considered as extended meeting points if there exists at least one connected graph with vertices $P_1,\ P_2,\ ...\ P_k$, such that the length of each edge is at most r.

Effective Communication Range

By communication range we mean the maximum distance in which two agents can send and receive data through their transmitters. As stated before, this range can be further improved if a third agent is place between the two, acting as a relay agent. By *effective communication range* we mean the maximum distance across which an agent can transmit some data with the help of relaying agents.

1.4 Our Contributions

We now give a summary of our results in this section. The results will be discussed in depth in the following chapters.

1. We propose a new communication model for agents in search and evacuation problems. This model is more realistic than the previously proposed wireless and face-to-face communication models.

2. For evacuation by two agents, we first modify the *Equal Travel with Detour* algorithm in [13] for some values of r. We describe a new kind of detour,

resulting in the $\mathcal{A}_4$ algorithm which can be used for any value of r, and improves the evacuation time over the algorithm using the old detour. We further improve the $\mathcal{A}_4$ algorithm by utilizing two detours instead of one. Our new algorithm also improves the evacuation time of two agents in face-to-face model from 2.3367 [13] to 2.2861. Finally, we give a lower bound for the evacuation time by 2 agents with communication range $r \leq 3y - 0.5 \approx 0.3660$.

3. For evacuation by three agents, we give three different strategies, each giving a better evacuation time than the other two strategies for some range of r.

4. For evacuation by four agents, we give two different strategies, each of them achieving a better evacuation time than the other for some range of r.

5. It is known that evacuation requires time at least $1 + 2y$ even in the wireless model, regardless of the number of agents. For any r, we give upper and lower bounds of $k = 6 + 2\lceil 1/r - 1 \rceil$ and $k = 1/r + 1$ respectively, for k, the number of agents required to achieve this optimal time.

1.5 Outline of Thesis

We describe related work in Chapter 2. In Chapter 3, we describe our upper and lower bounds for evacuation by two agents with communication range r. In Chapter 4, we describe our algorithms for 3 agents. The case of 4 and more agents is described in Chapter 5. We conclude with discussion and open problems in Chapter 6.

Chapter 2

Related Work

In this chapter, we first discuss the previous research carried out on search problems. This is followed by the previous work done on evacuation problems for both face-to-face and wireless models.

In search and evacuation problems, the agents are looking for a target (or multiple targets) in a discrete domain, such as a graph, or the continuous domain, such as a line. The main goal is to minimize the worst-case time needed to find the target utilizing a single or multiple number of searchers. We therefore often speak of an adversary that will do his best to hide the target in some place that is hardest for the searchers to find it. As mentioned in Chapter 1, in search problems, we minimize the worst-case time for *any* of the agents to find the target, while in evacuation, we are interested in minimizing the time for *all* the agents to reach the target.

2.1 Single-Agent Search Algorithms

In classical search algorithms it is assumed that there only exists a single agent, responsible for searching the whole domain. A few of these problems are discussed in the following.

2.1.1 Cow Path Problem

Search games were introduced with the classical *cow-path* problem [6, 7]. The search domain is an unbounded line and there is a treasure which is located at some point on the line. The sole purpose of the searcher is to find this treasure in the least time possible. A doubling zigzag algorithm was introduced in [8] in which the agent moves to the right for distance 2^{2i} from it's initial location in phase $2i$, and moves to the left for distance 2^{2i+1} in phase $2i+1$ where $i \geq 0$. The authors proved the competitive ratio of 9 for this algorithm and showed that this algorithm is optimal.

The cow path problem may differ from the reality, i.e. the speed of the agent may vary for each part of the trajectory. This may be caused by a slope in the terrain or perhaps the agent may have to slow down when it is searching for the exit. The authors of [25] take these possibilities into consideration and study the following models:

Beacon model where agent moves away from its original point of start with unit speed and moves toward the start point with speed $s \geq 1$.

Exploration history model where the agent searches the unexplored regions with unit speed, and already explored regions with speed $s \geq 1$.

Constant acceleration models where the agent can accelerate its speed at constant rate in some conditions, or move with speed one otherwise.

Tailwind model in which the agent moves left with unit speed and moves right with speed $s \geq 1$.

For the first three models above, the authors give *expanding-distance* strategies, wherein the algorithm works in phases. In Phase i, the robot moves distance d_i from the origin in the opposite direction to the previous phase, where $d_i = rd_{i-1}$, and r is called the *expansion ratio* of the algorithm. For the doubling strategy described previously, $r = 2$. The authors proved that the doubling zig-zag strategy is optimal for the beacon model. For the exploration history model, an asymptotically optimal distance-based zigzag strategy with expansion ratio of $r = \sqrt{2/(s+1)}$ is given [25].

Two models of *constant acceleration* are studied. In the first model, the agent's speed increases at a constant rate c until the next turn. Upper and lower bounds of 6.36 and 11.1 respectively are given for the doubling strategy. In the second model, the agent moves with unit speed going uphill and with constant acceleration going downhill. The competitive ratio in the second model is dependent on the type of the terrain. The authors showed that while the competitive ratio of the doubling strategy is unbounded if the agent is located on an inclined line or on top of a symmetric hill, the competitive ratio is constant for the same strategy if the agent is located at the bottom of a symmetric valley [25].

For the tailwind model, the authors analyzed an *expanding-time* zig-zag search strategy in which the agent moves to the right for r^{2i} time in phase $2i$, and αr^{2i+1} in phase $2i+1$ where $i \geq 0$. This is shown to work better than the distance-based doubling strategy.

Many variations of this problem have been studied; randomized algorithm for cow-path problem in [33], a ring instead of a line in [40], and searching for a line, parallel to one of the x or y axes in [32].

2.1.2 Continuous High-Low Search

The *high-low search* is a game played on an interval or a discrete subset of it. The hider chooses a point H on this interval, and for each guess that the searcher makes, the hider reveals whether the guess was higher than H or lower.

The continuous version of this search game was introduced in [5] where the hider picks a point H from the unit interval of $Q = [0, 1]$, and the searcher should announce a guess $g_i \in Q$ and then it is revealed that the H is greater than g_i or otherwise. The goal of the algorithm is to minimize the "sum of errors" cost function, which is equal to $c = \sum_{i=1}^{\infty} |g_i - H|$. Note that $c = 1$ for the naive *halving* strategy in which the searcher always chooses the mid-point of the interval. Baston and Bostcok proved a lower bound of $v^- = 0.6$ for the problem and gave an algorithm with upper bound of $v^+ = 0.628$ [5]. It is demonstrated in [2] that there exists an optimal strategy for this game , i.e. that $v^- = v^+$.

2.1.3 Infiltration Games

Assume at time $t = 0$ an infiltrator enters a set Q through a known point and heads to the *sensitive zone* $B \subset Q$. Searcher is responsible for defending this zone by capturing the infiltrator before it reaches B. A discrete version of this problem is introduced in [30] where Q is an array of $n+1$ cells and B is located on the last cell. The probability of capture is $1-\lambda$ if the defender and the infiltrator are both located in the same cell and zero otherwise.

Lalley considered this game under the simplifying assumption that the infiltrator can stay at the *safe zone* outside of Q for unlimited time and he should reach the target by time t, whereas t is known for both players [35]. Assuming both the defender and the infiltrator can move with speed 1, Lalley presented optimal algorithms for each player. It is obvious that the probability of a successful infiltration is at most λ if the defender does not move throughout the course of the game. The optimal infiltration strategy (called *Admiral Farragut* by Lalley) is to wait in the safe zone until time τ where τ is uniformly distributed over the integers $1 \leq \tau \leq t-n$, and then go full-speed toward the target reaching it at maximum time t. The best approach for the defender (called *orderly fallback* by Lalley) is to stay at cell 1 for the first time unit, go toward the target for ξ_1 time unit; rest for 1 time unit then move toward the target for ξ_2 time unit and so on, where $\sum \xi_i = n-1$. It is shown that if $t \to \infty$ then the probability of a successful infiltration approaches λ [35].

2.1.4 Cop and Robber Games

The game of *cop and robber* was introduced in [38]. In this game which is played on a connected graph G by two players, the first player, the cop, chooses a vertex on G to begin. Likewise the second player, the robber, chooses another vertex for his starting point. At each time step, the cop moves and then it is the robber's turn to move. A move consists of either moving to an adjacent vertex or staying at the same vertex. The movement of each player is observed by the other player. The cop wins if he catches the robber in a finite number of steps, otherwise the robber wins. It is obvious that if the cop has a winning strategy, he should catch the robber in maximum $n(n-1)+1$ time steps [1], where n is the number of vertices in G.

The graphs in which the robber has a winning strategy, are called the *robber-win* graphs and the graphs in which the cops have a winning strategy, are called *cop-win* graphs [38]. The purpose of this research is to determine which graphs are cop-win and which are robber-win. Let p and d be adjacent vertices. Vertex p is a pitfall if all neighbors of p are also neighbors of d. It is shown that G is a cop-win graph if we could reduce G to a single vertex by removing the pitfalls (with their corresponding edges) one after the other without any specific order [1].

2.2 Group Search Algorithms

As opposed to the single-agent search algorithms, group search algorithms utilize more than one agent in order to find the hidden treasure or treasures. Since the process is parallelized and distributed between agents, the total time decreases. However, other issues emerge, such as how agents communicate, what happens in a case of failure, and also load balancing. The goal of the search algorithms is generally to minimize the required time to find the treasures. Reducing the required number of searchers could be another goal.

2.2.1 Faulty Agents in the Cow Path Problem

Consider the problem of group search on a line. It is clear that with two agents, search can be done optimally in time d where d is the distance from the origin to the exit. However, the assumption that the system is flawless is not completely practical as every system is susceptible to malfunction. Hence variations of group search problems where a few of the agents are faulty have been studied. In [23] the *Cow Path* problem is revisited with n agents out of which f are faulty. The faulty agents can traverse the line but they cannot detect the target. Such a type of fault is called a *crash fault*. Note that it is not known in advance which agents are faulty, in fact the faulty agents themselves don't know that they are faulty. The goal in [23] is to get a lower search time for the worst case, i.e. to decrease the time in which at least $f + 1$ agents have visited the target. They gave an algorithm with competitive ratio of 5.24 for three agents out of which one is faulty, and an optimal competitive ratio of 3, for the case $n = 2f + 1$ when n tends to ∞.

The Byzantine Generals Problem is a classical problem in distributed computing introduced in [36] where all the loyal generals should reach the same agreement on a plan ignoring the traitor generals. Since then, agents that are not just faulty but *malicious liars* have been considered in many distributed computing problems [34, 12, 21]. A similar idea is used in [24] for the classical cow-path problem where all the agents have wireless communication capabilities and Byzantine agents can not only stay silent while visiting the exit, but also they can lie about the exit and send every agent the location of a non-exit point as the exit. The goal of search with Byzantine agents is to minimize the time needed for all the non-faulty agents to be certain that the correct location of the exit has been found. The authors gave a competitive ratio of 3, 2 and 1 for four, five and six agents, respectively out of which 1 is Byzantine. They also conjectured that the competitive ratio cannot be improved for 3 agents including 1 Byzantine from the crash-fault case. Later, the lower bound for 3 agents including 1 Byzantine was improved from 3.93 [24] to $\frac{8}{3}\sqrt[3]{4} \approx 5.23$ [34], thus showing that the algorithm in [23] is optimal for 3 agents, one of which is either crash-faulty or Byzantine-faulty.

In the previous forms of faulty agents, we assumed that the faulty agents start malfunctioning right at the beginning of the execution of the algorithm. In contrast [26] looks at the problem where we have infinite number of boxes and the searches should open them one by one to find the hidden prize. It is possible for some of the agents to stop functioning at some point in the middle of execution of the algorithm and the other searchers should recognize the breakdown of these agents.

2.2.2 Cops and Robbers Games

A variation of the *cop and robber* game described earlier is where there is more than one cop in the game. The motivation is to identify $c(G)$, the minimum number of cops in a graph G to catch the robber. It is shown that if graph G does not have any cycle with length of 3 or 4, $c(G)$ should be at least the maximum degree of G [1]. Also $c(G) \leq 3$ for any planar graphs [1]. Many variations of these games have been studied quite extensively; see [28, 37, 9].

2.3 Evacuation Problems

The evacuation problem was first introduced in [16], looking at the evacuation of multiple agents from a disk. The evacuation algorithms generally proceed in two phases: first, agents have to search the perimeter looking for the exit, and then in the second phase they communicate and move to the exit. Though we can say that the first phase is a group search problem, we should look at the problem as a whole, since an optimal algorithm for the first phase does not necessarily guarantee a best solution for the whole problem.

The evacuation problems that we are interested in, differs from the evacuation problem studied in [27], where the location of the exit is known to the evacuators, and the domain is a collection of cells, where at most one person can occupy a cell at any time. In contrast, in our problem, the location of the exit is unknown to the evacuators, the domain is the perimeter of a polygon, and a single point can be occupied with multiple agents.

We will briefly discuss the research done on the evacuation problems so far categorizing them based on the search domain.

2.3.1 Disk

Evacuation from a disk with unit radius is studied in [16] for two, three as well as a very large number of agents, for both face-to-face and wireless models. The authors were able to give a tight bound for a large number of agents for both models, as well as for two agents in the wireless model.

For evacuating two agents, the algorithm for both models are identical up to the time that the exit is found. Both agents move to an arbitrary point A on the perimeter and from there, one agent starts searching clockwise while the other agent starts searching counter-clockwise. In the face-to-face model, when the exit is found by one of the agents, it calculates the shortest path to intercept the other agent and then the two agents go back to the exit together. In the wireless model, the location of the exit is broadcasted and the second agent moves immediately to the exit. It is shown that the worst placement of the exit in this specific algorithm for face-to-face and wireless models are at distance 1.91 and 2.09 from point A, respectively.

The notion of using *detours* into the region, in order to improve evacuation time, was first introduced in [17] for the case of two agents in the face-to-face model, which is later used in other evacuation problems with different domains. The two agents at some point during the execution of the algorithm leave the boundary of the disk and move to a specific point inside the disk. If the exit is not found by either of the agents, they move back to the same point on the boundary and continue searching the unexplored part of the disk, otherwise, they go to the exit. With the use of a detour, the evacuation time was improved from 5.74 in [16] to 5.628 in [17]. Also the lower bound was improved from 5.199 to 5.255.

In [15] the evacuation problem with two wireless agents is studied with multiple exits and slightly different settings. The perimeter of the disk is 1 and agents are initially located on the boundary at distance L from each other. The agents are aware of the distance between each pair of exits, but they don't know their own initial positions. The paper gives an upper bound of $\frac{3}{4} \cdot D + \frac{L}{2}$ and a lower bound of $\frac{3}{4} \cdot D - \frac{L}{2}$ for arbitrary values of L and D where L is the distance between the two agents and D is the length of the longer of the two arcs between two consecutive exits.

Evacuation from a disk with two exits located at distance d from each other is revisited in [39] for two agents in both wireless and face-to-face models. It is shown that it is better for the agents to move to two different points on the perimeter of the disk with distance d apart from each other for the wireless model, though for the face-to-face model, the value of d determines whether it is better for the agents to move to the same point on the perimeter or they should move to different points with distance d [39].

The case of three wireless agents with at most one of them being faulty are studied in [21]. The authors consider two fault models, *crash* fault, in which the agent cannot recognize the exit, or it is unable to transmit the location of the exit to the others; and *Byzantine* faults in which the agents may or may not recognize the exit, but may be malicious, and can transmit false data to the other agents. For the first fault model, they proposed a trajectory for agents where two of them go to the same point on the perimeter of the disk and start exploring in opposite directions; and the third agent moves to a point on the perimeter with arc-distance of approximately 2.966 from one of the agents and starts moving toward the closest one. Hence the crash fault model can be solved in time ≈ 6.309 [21]. In the algorithm for the Byzantine model, the agents go to opposite points on the perimeter, and start exploring in the same direction. The difference is that they continue exploring their share of the perimeter, even if they receive a message that exit is found. After the whole disk is explored, then they

- continue exploring if no exit location is reported,
- they go directly to the exit if only one exit location is reported
- they go to the nearest reported exit location and if they don't find any exit there, they move to the second reported exit.

It is shown that the maximum evacuation time for the above algorithm s equal to $1 + \frac{4\pi}{3} + \sqrt{3} \approx 6.921$ [21].

2.3.2 Square

In [19], the evacuation problem in unit sided squares for wireless model is studied. For the two-agent problem, the trajectories are shown in Figure 2. Both agents start exploring in opposite directions from some point with distance x from the nearest vertex. It is shown that the best value for x is equal to $\frac{1}{2} - \frac{1}{\sqrt{12}}$ which yields a tight bound of $\frac{3}{2} + \frac{\sqrt{3}}{2}$ for evacuating two agents. The evacuation time is maximized if the exit is placed at either points C or E. For more than two agents, the improvement

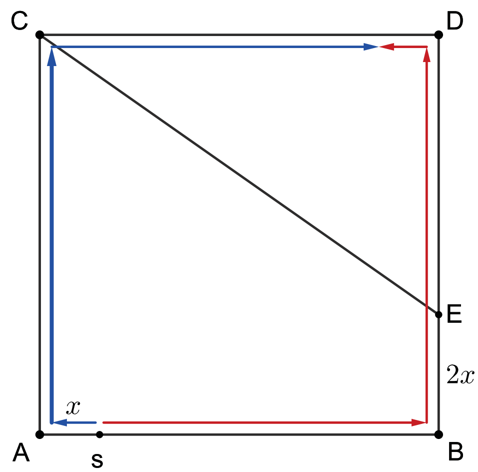


Figure 2: Evacuation of two agents from a square [19]

depends on the starting point. It is shown in [19] that no improvement can be made by using more that two agents if the starting point is one of the corners. For $k \geq 2$ agents starting from the center of the square, the evacuation time is at most $\frac{3\sqrt{2}}{2} + \frac{4}{k}$ and no algorithm can do better than $\frac{3\sqrt{2}}{2}$.

Furthermore, the authors in [14] showed that in the face-to-face model, two agents can evacuate the square in 3.4645 and any algorithm requires time at least 3.118. They also gave evacuation times 3.1786 and 2.6646 for three and four agents, respectively.

2.3.3 Triangle

Evacuation of multiple agents from an equilateral triangle with sides of length one is studied in [13] and [19]. Two models of communication, wireless and face-to-face, are studied and lower and upper bounds are stated for both of the models.

Wireless model

Evacuation of multiple agents with wireless communication capabilities in a triangle with unit sides is studied in [19]. In the case of two agents, it is shown that if the agents are located on the perimeter of the triangle, then $x + \frac{3}{2}$ is a tight bound for evacuating agents where x is the distance of the agents to the closest midpoint of an edge. Hence placing the agents initially on the midpoint of an edge results in minimum evacuation time. When the initial location of the agents is in the interior of the triangle, the results are the same, i.e. $x + \frac{3}{2}$ is a tight bound and x is the distance of the agents to the closest midpoint of an edge [19]. Therefore if the agents start at the centroid of the triangle, evacuation time will be $\frac{1}{2\sqrt{3}} + \frac{3}{2}$. The trajectories of both agents are shown in Figure 3.

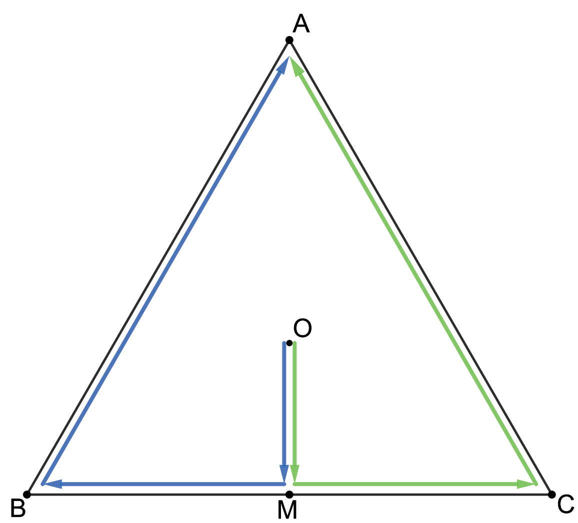


Figure 3: Evacuation of two agents from a triangle [19]

For the case of $k > 2$ agents, the same approach is used. One of the edges is divided in to k segments and each agent explores one segment and then moves to the third vertex. The trajectories of three and four agents are illustrated in Figure 4.

Face-to-face model

Evacuation in an equilateral triangle in the face-to-face model is first studied in [13]. It is shown that for any number of agents in the face-to-face model, the evacuation time is at least $\sqrt{3}$, and for two agents, the lower bound is improved to $1 + \frac{2}{\sqrt{3}}$.

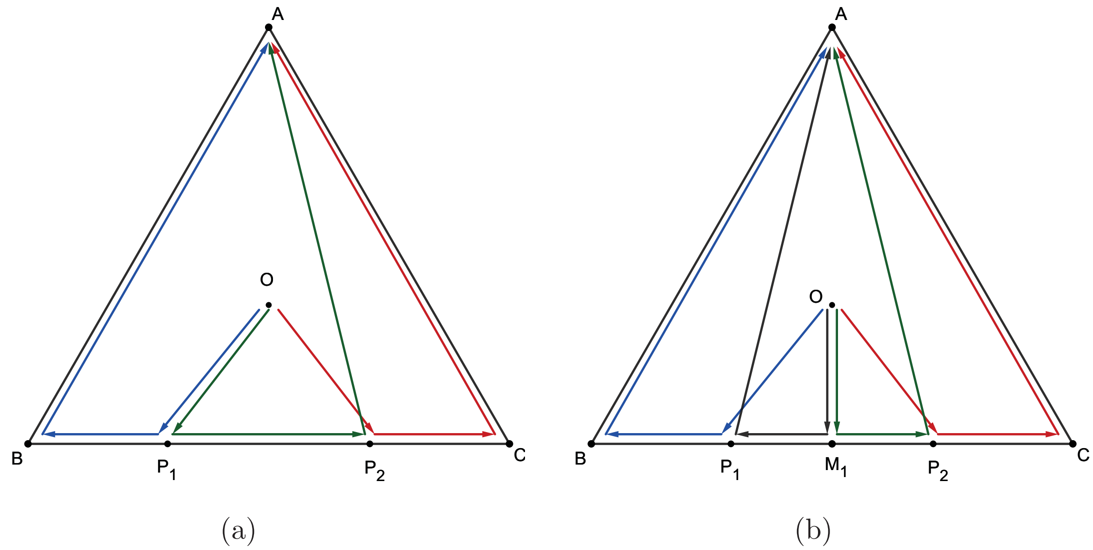


Figure 4: Evacuating (a)three and (b)four agents with wireless transmitters in [19].

For three and more agents *Equal-Travel Early-Meeting Algorithms* [13] are proposed. In these algorithms with k agents, the perimeter of the triangle is divided in to $k+1$ segments and each segment is assigned to be explored by an agent. After these segments are explored, agents move to the interior of the triangle in order to exchange the information. These segments are chosen in such a way that all agents get to the meeting point at the same time. If the exit is not found until then, then they move to the last segment together.

Improved algorithms for 2 agents are given in [13]. Instead of travelling to a meeting point, agents make a detour and move to the interior of the triangle. At the end of the detour, if the agents are not intercepted by the other agent, they know that the exit is not found and return to the perimeter of the triangle to resume the exploration. The trajectories of both agents are shown in Figure 5.

In theory any number of detours can be implemented for evacuating two agents; however it is shown in the same research that the improvement achieved by the third and subsequent detours is insignificant. The achieved evacuation time of two agents with *Equal-Travel with Detour strategy* is 2.3367.

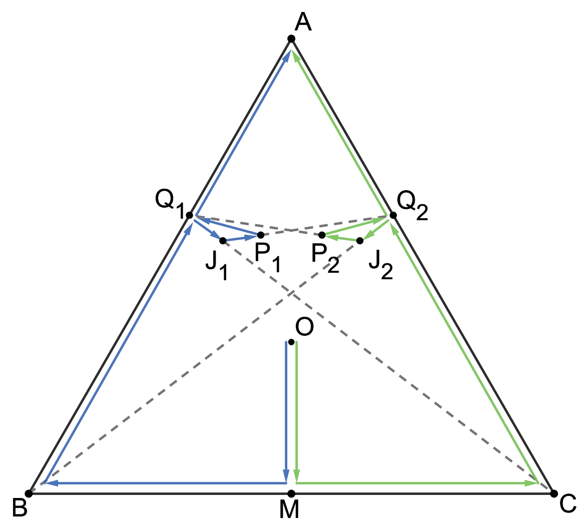


Figure 5: Evacuation of two agents from a triangle with one detour for each agent in [13]

Chapter 3

Evacuating Two Agents

In this chapter we give upper and lower bounds on the evacuation time for two agents in an equilateral triangle of side 1. We divide the triangle into two halves by a vertical line through A, as shown in Figure 6. In our algorithms, the first agent R_1 is responsible for searching for the exit in the left half and the second agent R_2 is responsible for the right half. All the trajectories presented in this chapter are *symmetric* with respect to line AM, where M is the midpoint of edge BC. Therefore, without loss of generality, in the analysis of algorithms throughout this chapter, we

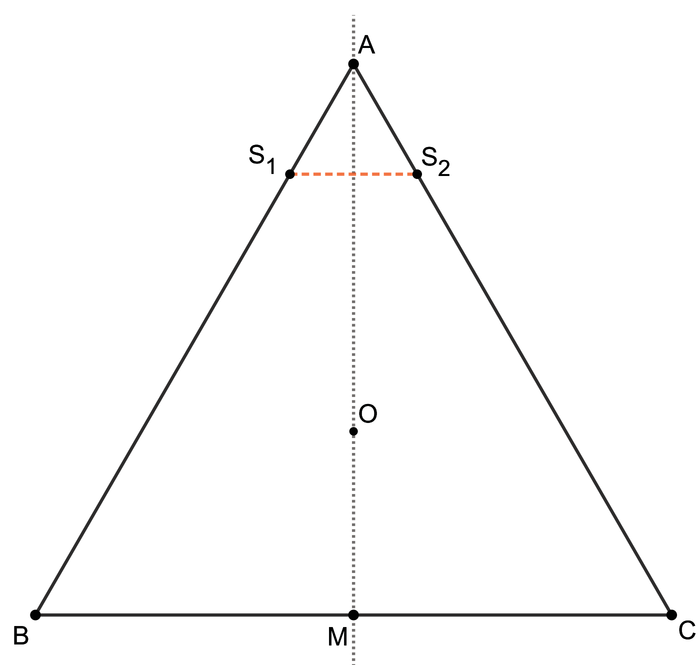


Figure 6: Search domain is an equilateral triangle with sides 1

assume that the exit is located on the right half of the triangle.

Let S_1 and S_2 be points on the sides AB and AC at distance r from A, shown in Figure 6. This makes an equilateral triangle ΔS_1AS_2 at the top of T with side r. If the agents do not find the exit outside ΔS_1AS_2 and enter this smaller equilateral triangle, they are always in communication range with each other and the evacuation time for all the algorithms described in this chapter, is independent of the exit position and will always be $t_{S_1} + r$. Recall that t_{S_1} is the time that S_1 is visited by *any* of the agents for the first time.

In the algorithms in the *face-to-face* models, if one agent finds the exit, it moves to a point, in a way that both of the agents will be at the same point at the same time. However in our model, since agents have the ability to communicate within a certain range, there is no need for both of them to be at the same point to communicate, an agent can calculate a path to point p so that when it reaches point p, the other agent is at distance at most r from the first agent. By r-interception we mean the action of calculating the point p and getting there so that the two agents can communicate with each other.

Recall that the best known algorithm described in [13] for the face-to-face model, evacuates two agents in time 2.3367 and employs two detours. In [19], an optimal algorithm for wireless model with evacuation time of $3/2 + y \approx 1.78867$ is described. Hence if the agents are capable of communication within a certain range $r > 0$, it is clear that:

$$1.78867 \leq E^*(2, r) \leq 2.3367$$

Algorithm 1 describes the behavior of both agents in T. The main idea is that each agent moves along a pre-specified trajectory looking for the exit, and as soon as one of them finds the exit, it r-intercepts the other agent and they both move to the exit. Throughout this chapter, we use the same evacuation algorithm with different trajectories.

3.1 A Simple Evacuation Trajectory

The trajectories of both agents are shown in Figure 7. The trajectory of R_1 is shown in blue, from the centroid to M to vertex B and then to vertex A. The green trajectory is for R_2, from the centroid to M to C and then to A. Whichever agent finds the

Algorithm 1 Evacuation Algorithm for Two Agents

function EXPLORATION
    found← false
    while not<found> and not<msg_recd > **do**
        move along the predetermined trajectory
    end while
    ACTION
end function
function ACTION
    if found **then**
        P ← current location
        if the other agent is not in communication range **then**
            calculate the closest point U, where the other agent can be r-intercepted
            go to U
        end if
        send(P) to the other agent
    end if
    go to P and exit
end function

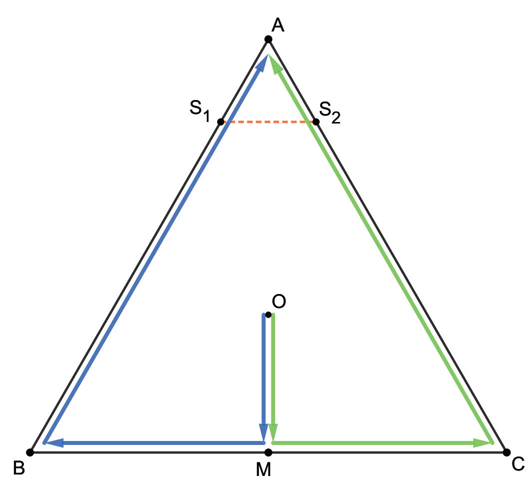


Figure 7: Trajectory of Agents in Simple Algorithm

exit, it will abort its exploration phase and attempt to r-intercept the other agent by possibly going in the interior of the triangle. After informing the other agent, both agents return to the exit point.

We are going to describe the trajectory of each agent as shown in Trajectory 1. The sequence shows the order of points visited by each agent, whereas the first point is the starting point and the last point of the sequence is the last point visited if the whereabouts of the exit is still unknown to them.

Trajectory 1.

$$R_1 \text{ follows the trajectory :} < O, M, B, A >$$

$$R_2 \text{ follows the trajectory :} < O, M, C, A >$$

By $\mathcal{A}_1$, we mean Algorithm 1 with respect to Trajectory 1. In our analysis we use the following lemma from [10] in order to determine the critical point of some segments in T.

Lemma 3.1. *[10] Suppose R_1 and R_2 with $r = 0$ are looking for an exit on lines L_1 and L_2 respectively, see Figure 8. Wlog assume the exit is found by R_2. Let point N be the exit point and Q where R_1 is intercepted and S be the line connecting N and Q. Denote the angle between L_2 and S by β and angle between L_1 and S by γ. Assuming the function describing the trajectory of R_2 is differentiable at N, if $2cos\beta + cos\gamma \neq 1$ then there exists another exit point that yields a larger evacuation time than placing the exit at N.*

More precisely if $2cos\beta + cos\gamma < 1$, then shifting the exit in the direction of the movement of R_2 yields a larger evacuation time, and if $2cos\beta + cos\gamma > 1$, then shifting the exit in the opposite direction of the movement of R_2 yields a larger evacuation time.

For the proof that Lemma 3.1 still holds when agents at distance at most r can communicate, we need the following lemma from [10]. Note that Lemmas 2.1, 2.2, and 2.6 and some definitions from [10] have been combined to give Lemma **??** below.

Lemma 3.2. • *[10] Set $z := 1 - 2\cos\beta - \cos\gamma$. If $2cos\beta + cos\gamma < 1$ and $\beta \leq \pi/2$, then $|DE| \geq 1 - \varepsilon/2 + z \cdot \varepsilon/2$, for some ε with $0 < \varepsilon \leq 0.5$ where D is the*

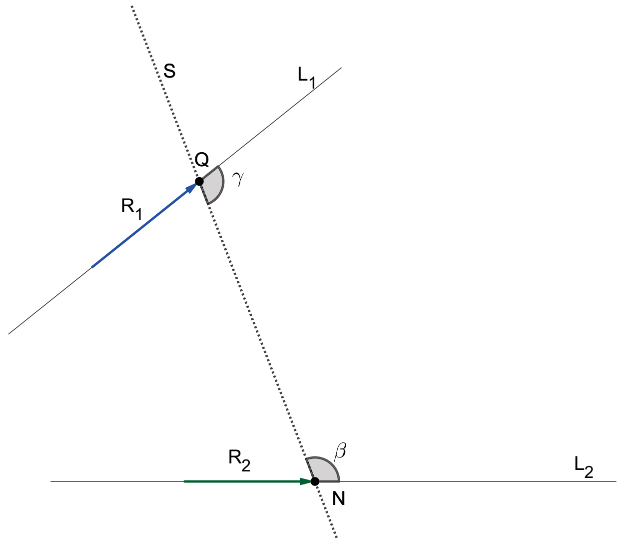


Figure 8: Lemma 3.1

point that R_2 would reach in distance ε from N if it continued on L_2 and E is the point that R_1 would reach in distance $\varepsilon/2$ from Q if it continued on L_1 (see Figure 9b) .

- *[10] Set $z := 1 - 2\cos\beta - \cos\gamma$. For all F on QE, if $2cos\beta + cos\gamma < 1$ and $\beta \le \pi/2$, then $\varepsilon + |DF| \ge 1 + |QF| + z. \cdot \varepsilon/2$, where the definitions of ε and points D and E are the same as above.*

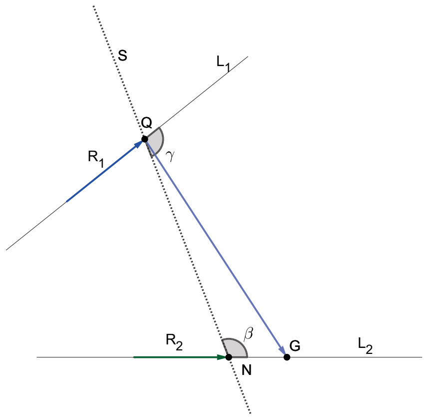
- *[10] Set $z := 2\cos\beta + \cos\gamma - 1$. If $2cos\beta + cos\gamma > 1$, then $|DE| \ge 1 - \varepsilon/2 + z \cdot \varepsilon/4$, for some ε with $0 < \varepsilon \le 0.5$ for some ε with $0 < \varepsilon \le 0.5$ where D is the point on L_2 that R_2 reached in distance ε from N before arriving at N and E is the point that R_1 reached in distance $\varepsilon/2$ from Q before arriving at Q (see Figure 9c) .*

Lemma 3.3. *Lemma 3.1 still holds for $r > 0$.*

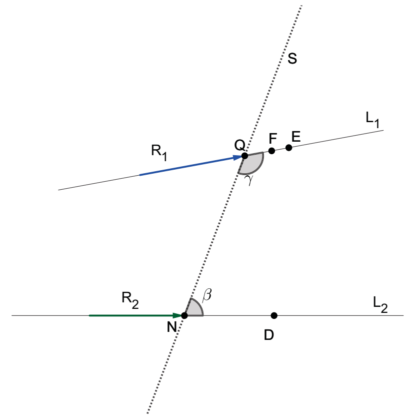
The proof is very similar to the proof in [10], however we will discuss it in depth for completeness.

The Case $2cos\beta + cos\gamma < 1$

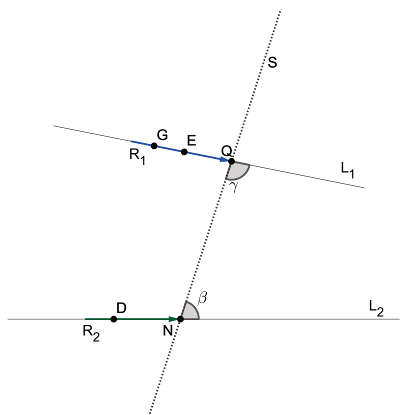
For simplicity we assume that $|QN| = 1$, which can be generalized to any value. First assume that $\beta > \pi/2$, see Figure 9a. Let t be the time that R_2 is at N. Clearly R_1



(a) The case $\beta > \pi/2$



(b) The case $2cos\beta + cos\gamma < 1$



(c) The case $2cos\beta + cos\gamma > 1$

Figure 9: The settings of agents in different cases based on β and γ

is at point Q at time $t' = t + 1 - r$, and the evacuation time is $t' + 1 = t + 2 - r$. If we shift the exit to point G, since R_1 cannot be r-intercepted sooner than t', then R_1 still will be at point Q at time t'. The evacuation time is at least $t' + |QG|$ and because $\beta > \pi/2$, we know that $|QG| > |QN| = 1$. We conclude that shifting the exit toward the direction of R_2 results in larger evacuation time.

Now we consider the case $\beta < \pi/2$. Let D be the point that R_2 will reach in distance ε from N, if it does not find the exit at N. Let E be the point that R_1 will reach in distance $\varepsilon/2$ from Q, if it continues moving on L_1, see Figure 9b. Set $z = 1 - 2cos\beta - cos\gamma$. Clearly $0 < z \leq 2$.

Lemma 3.4. *If $2cos\beta + cos\gamma < 1$ and $\beta \leq \pi/2$, placing the exit at D will result in larger evacuation time than placing the exit at N.*

Proof. Recall that the evacuation time if the exit is located at N is $t + 2 - r$. By placing the exit at D, R_2 will find the exit at time $t + \varepsilon$. We claim R_1 is picked up at time later than $t + 1 + \varepsilon/2 - r$. Suppose not, meaning R_1 will be picked up at most $t + 1 + \varepsilon/2 - r$. Agent R_1 should be picked up on some point F on QE at time $t+1+|QF|-r$. R_2 will go directly to point F after finding the exit and will r-intercept R_1 at time $t + \varepsilon + |DF| - r$ which by Lemma 3.2 is larger than $t + 1 + |QF| - r$. A contradiction and we conclude that R_1 is picked up at time later than $t+1+\varepsilon/2-r$.

Agent R_1 is at E at time $t+1+\varepsilon/2-r$ and if it travels directly to the exit point from E, the evacuation time will be $t + 1 + \varepsilon/2 - r + |DE|$, by Lemma 3.2 we have $|DE| > 1 - \varepsilon/2$. So the evacuation time if the exit is located at D, will be

$$t + 1 + \varepsilon/2 - r + |DE| > t + 1 + \varepsilon/2 - r + 1 - \varepsilon/2 = t + 2 + \varepsilon/2 - r$$

which is larger than the evacuation time if the exit is located at N. □

The Case $2cos\beta + cos\gamma > 1$

Let this time D be the point with distance ε from N which R_2 reaches before arriving at N. Similarly E is the point that R_1 reaches at distance $\varepsilon/2$ from Q before arriving at point Q, see Figure 9c. We use an analogous version of Lemma 3.2 for the case of $2cos\beta + cos\gamma > 1$.

Lemma 3.5. *If $2cos\beta+cos\gamma > 1$, placing the exit at D will result in larger evacuation time than placing the exit at N.*

Proof. Let $t = 0$ be the time that R_2 is at point D. Then R_2 will reach point N at time ε and if the exit is located at N, agent R_1 will be r-intercepted at time $\varepsilon+1-r$ and evacuation time will be $2 + \varepsilon - r$.

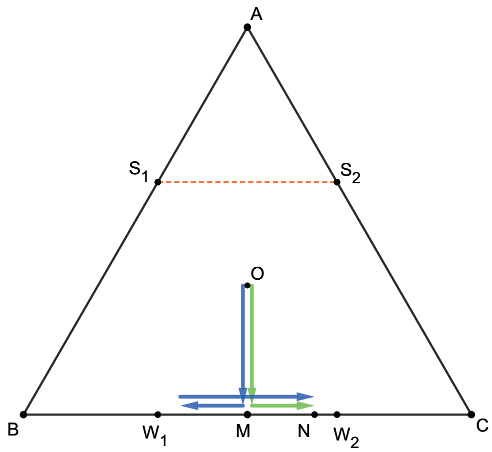
Now we place the exit at D. We claim R_1 will be r-intercepted after time $1+\varepsilon/2-r$. For contradiction, lets say otherwise meaning R_1 is r-intercepted at some point G at time $t' \leq 1 + \varepsilon/2 - r$. If R_1does not get notified, it will be at point E at time $1 + \varepsilon/2 - r$. So $|EG| \leq 1 + \varepsilon/2 - r - t'$. On the other hand, R_1 is r-intercepted when at G, so $|DG| \leq t' + r$. According to triangle inequality we should have $|DE| < |DG| + |EG| \leq 1 + \varepsilon/2$ which is a contradiction to Lemma 3.2 and our claim that R_1 will be r-intercepted after time $1 + \varepsilon/2 - r$ holds.

Therefore R_1 is r-intercepted at point E at time after $1 + \varepsilon/2 - r$ and by Lemma 3.2 we have $|DE| > 1 + \varepsilon/2$, so placing the exit at D results in evacuation time of

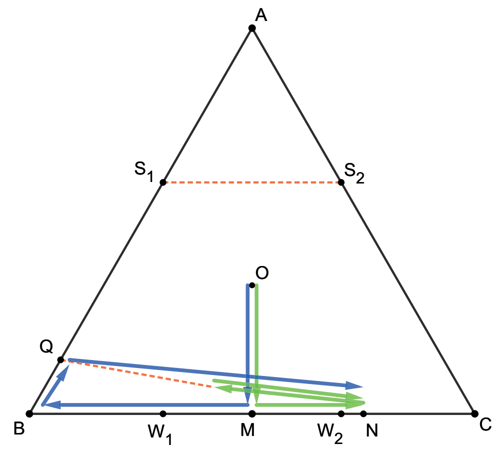
$t_e > \varepsilon/2 + 1 - r + 1 - \varepsilon/2 = 2 + \varepsilon - r$, which is larger than the evacuation time if the exit is located at N.

□

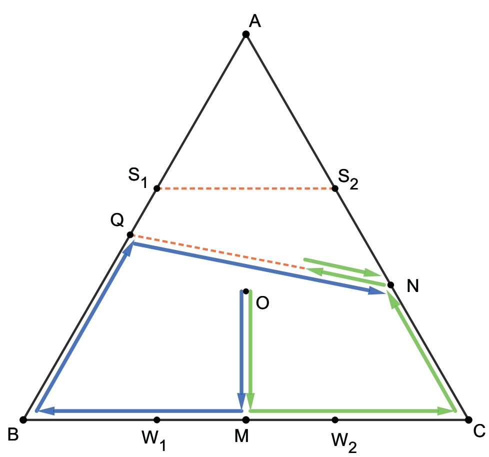
For the analysis of algorithm $\mathcal{A}_1$, we assume the exit is found by R_2. Then we give a critical point for each segment of the trajectory, and later we show that the maximum evacuation time is when the exit is located at point C.



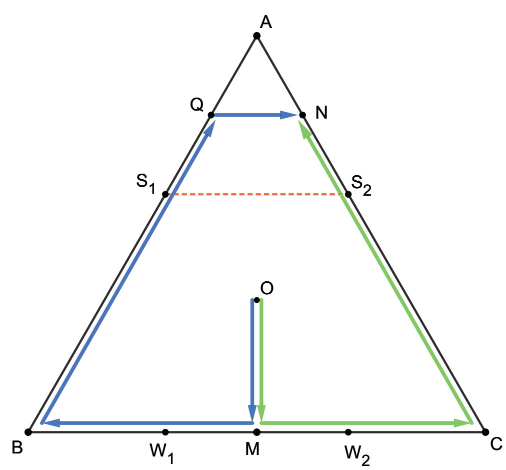
(a) Exit is located on segment MW_2



(b) Exit is located on segment W_2C



(c) Exit is located on segment CS_2



(d) Exit is located on segment S_2A

Figure 10: Trajectory of agents based on the position of the exit point

Lemma 3.6. *On segment MW_2 (see Figure 10a) where $|MW_2| = {}^{r}/_{2}$, point W_2 is the critical point and the evacuation time for this point is at most $y + 1.5$.*

Proof. Assume the exit is located at some point, N, on segment MW_2. Since the agents travel with the same speed, when R_2 reaches the exit, agent R_1 is located at the same distance from M as R_2 is from M. The distance between the two agents is $2|MN| \leq r$, meaning they are in communication range. The evacuation time is $y + 3|MN|$ and this value is maximized in W_2 and is equal to $y + \frac{3}{2}r \leq y + 1.5$ □

Lemma 3.7. *On segment W_2C (see Figure 10b), vertex C is the critical point.*

Proof. Suppose the exit is located at some point N on segment W_2C . Then R_1 will be r-intercepted while travelling on edge BA. Hence according to Lemma 3.3, since $\beta > \pi/2$, then N could not be the point that yields the maximum evacuation time since there exists another point to the right of N, which yields a larger evacuation time. We conclude that C is the critical point in this segment. □

Lemma 3.8. *On segment CS_2 (see Figure 10c), vertex C is the critical point.*

Proof. Let the exit be at some point N on segment CS_2 and the interception point of R_1 be Q on side BA. We know that $\beta + \gamma = \frac{2\pi}{3}$, and for all points on segment CS_2, the angle β is between 0 and $\pi/3$. Then we get $2cos(\beta) + cos(\gamma) = 2cos(\beta) - cos(\pi/3 + \beta)$ which is strictly greater than one. Hence based on Lemma 3.3 placing the exit at a point closer to C will result in higher evacuation time. We can conclude that in this segment, vertex C is the critical point. □

Lemma 3.9. *If the exit is located on segment S_2A (see Figure 10d), then the evacuation time is independent of exit location and is equal to $y + 1.5$.*

Proof. Since in this segment, the two agents are in communication range and the first agent that finds the exit sends its location to the other agent immediately and exits. Suppose R_2 find the exit at N, then R_1 will get notified when it is at Q (see Figure 10d). We conclude that the evacuation time is $y + 0.5 + |BQ| + |QN| = y + 0.5 + |BQ| + |QA| = y + 1.5$. □

Lemma 3.10. *The evacuation time when the exit is at C equals $y + 0.5 + r + \frac{2(1-r^2)}{2r+1}$.*

Proof. When the exit is located at C, the evacuation time will be $t = y + 0.5 + |BQ| + |QC|$ where Q is the point that R_1 is r-intercepted. Since both agents travel equal distances at the point of r-interception, we get $|BQ| = |QC| - r$. On the other hand by using the Cosine Rule we have $|QC| = \sqrt{BQ^2 + 1 - BQ}$. By solving for $|BQ|$ and substituting in the expression for the evacuation time t, we obtain $t = y + 0.5 + r + \frac{2(1-r^2)}{2r+1}$.

□

We now give the evacuation time for $E_{\mathcal{A}_1}$.

Theorem 3.11. *The worst-case evacuation time obtained by $\mathcal{A}_1$ is achieved at point C, and thus $E_{\mathcal{A}_1} = y + 0.5 + r + \frac{2(1-r^2)}{2r+1}$.*

Proof. Observe that $r + \frac{2(1-r^2)}{2r+1} : r \in [0, 1]$ is a decreasing function with maximum and minimum of 2 and 1 respectively. Therefore $y+0.5+r+\frac{2(1-r^2)}{2r+1} \geq y+0.5+1 = y+1.5$, so based on Lemmas 3.6 and 3.9, this evacuation time is also larger than maximum evacuation time of segments MW_2 and S_2A. We conclude that placing the exit at point C results in maximum evacuation time. The bound then follows from Lemma 3.10. □

3.2 Evacuating 2 Agents from a Triangle with Detour

In Theorem 3.11 we showed that placing the exit at point C causes the maximum evacuation time. In this section we give a modified version of *Equal-Travel with Detour* algorithm [13] in order to decrease the maximum evacuation time. The idea of a detour is that R_1 at some point will abandon exploring for the exit and move into the triangle, in order to facilitate the evacuation, if the exit is located on some segments of T. But if the agent realizes that the exit was not found in that segment, it returns to the same point on the boundary where it left off and resumes exploring.

3.2.1 Equal Travel with One Detour

We fix two points Q_1 and Q_2 on the sides AB and AC respectively. The exact location of these points will be specified later. Note that $|BQ_1| = |CQ_2|$ Point J_1 is on segment Q_1C, such that it satisfies the following equation:

$$|BQ_1| + |Q_1J_1| = |CJ_1| - r$$

Point P_1 is located on segment J_1Q_2, where P_1 satisfies the equation below:

$$|Q_1J_1| + |J_1P_1| = |Q_2P_1| - r$$

Points J_2 and P_2 are located symmetrically with those of points J_1 and P_1 respectively, with respect to line AM. The trajectory of both agents are described in Trajectory 2 and is shown in Figure 11.

Trajectory 2.

$$R_1 :< O, M, B, Q_1, J_1, P_1, Q_1, A >$$

$$R_2 :< O, M, C, Q_2, J_2, P_2, Q_2, A >$$

When R_1 reaches point P_1, if it does not get notified by the other agent, then with no need of communication, it realizes that the exit is not found by R_2 and terminates the detour by returning to point Q_1 where it started the detour in the first place and resumes exploring the boundary to find the exit point. We denote by $\mathcal{A}_2$, Algorithm 1 with respect to Trajectory 2.

We now show that R_1 will be r-intercepted before reaching point P_1 if R_2 finds the exit before reaching Q_2. Note that W_3 is a point on segment MC such that if the exit is located at W_3, then R_1 will be r-intercepted at point Q_1, more formally

$$|MB| + |BQ_1| = |MW_3| + |W_3Q_1| - r$$

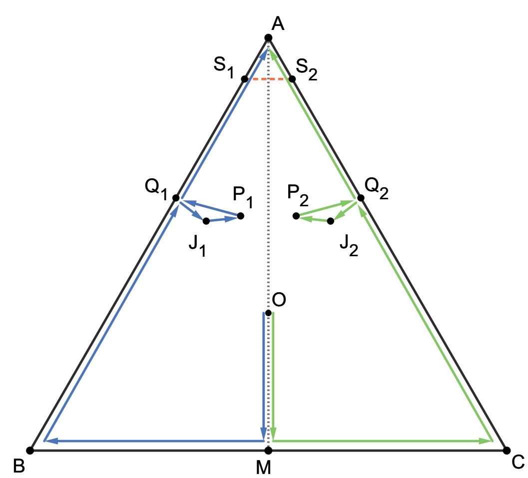
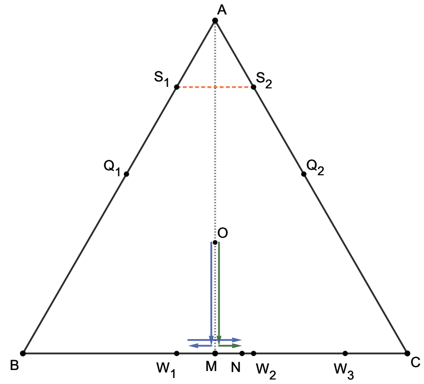
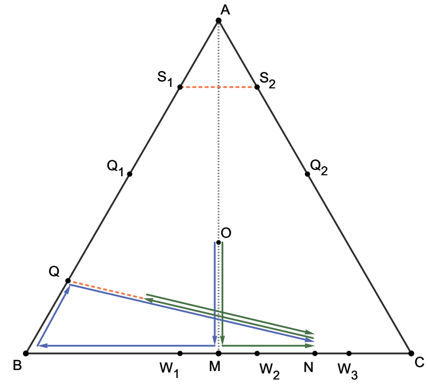


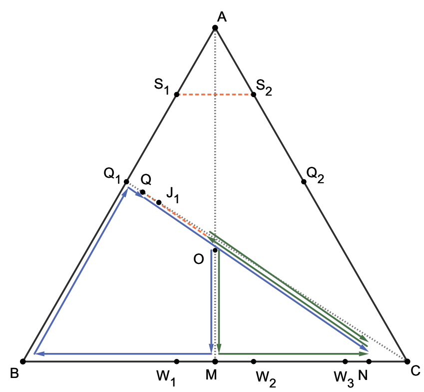
Figure 11: Trajectory of agents in One-Detour Model



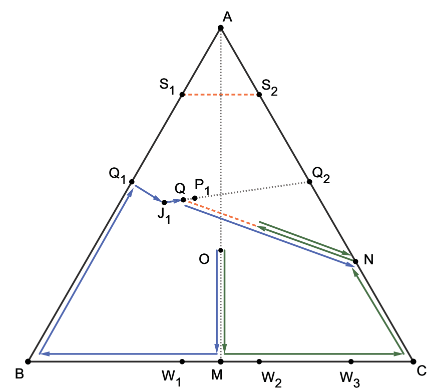
(a) Exit is located on segment MW_2



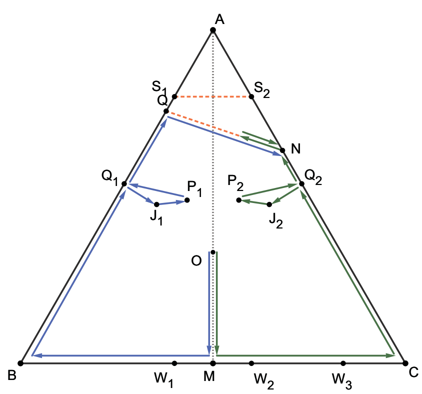
(b) Exit is located on segment W_2W_3



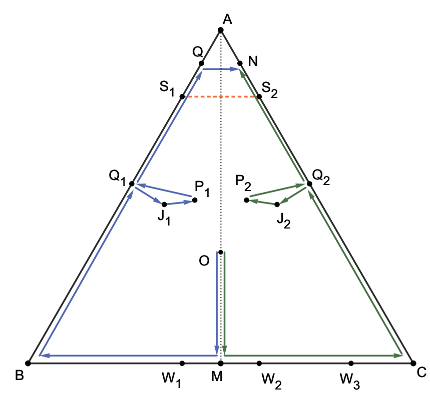
(c) Exit is located on segment W_3C



(d) Exit is located on segment CQ_2



(e) Exit is located on segment Q_2S_2



(f) Exit is located on segment S_2A

Figure 12: Trajectory of agents based on the position of the exit point in One Detour algorithm

Lemma 3.12. *Assume the exit is located at some point N on segment W_3C, then R_1 will be r-intercepted while moving on segment Q_1J_1.*

Proof. Note that by the definition of point W_3, agent R_1 cannot be r-intercepted before reaching point Q_1. Also by triangle inequality, in ΔJ_1NC, see Figure 12c we have $|NJ_1| < |NC| + |CJ_1|$, hence R_1 is going to be r-intercepted at point J_1 at the latest in this case. □

Lemma 3.13. *Suppose the exit is located at some point, N, on segment CQ_2, then R_1 will be r-intercepted while moving on segment J_1P_1.*

Proof. We know that if the exit is located at Q_2, then R_1 will be r-intercepted when it is at point P_1. In order to show that if the exit is before Q_2, agent R_1 can be intercepted before reaching P_1 it is enough to prove $|CN| + |NP_1| - r \leq |CQ_2| + |Q_2P_1| - r$. For the purpose of contradiction suppose not, meaning $|CN| + |NP_1| - r > |CQ_2| + |Q_2P_1| - r = |CN| + |NQ_2| + |Q_2P_1| - r$ and we get $|NP_1| > |NQ_2| + |Q_2P_1|$ which according to the triangle inequality is impossible. Hence a contradiction. □

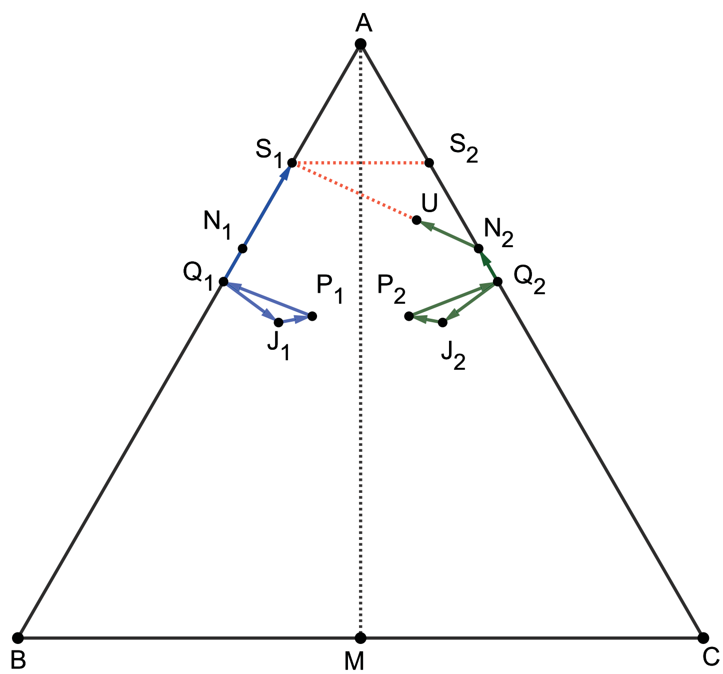


Figure 13: If the exit is located after Q_2 and before S_2, then R_1 will be r-intercepted before it reaches point S_1.

Lemma 3.14. *Suppose the exit is located between Q_2 and S_2 as in Figure 13. Then R_1 can be r-intercepted before reaching S_1, assuming $r > 0$.*

Proof. We have to show that if R_2 travels directly toward S_1 from some point N_2 between Q_2 and S_2 to a point U at distance r from S_1, it takes less time to be in communication range with R_1 than going to point S_2. In other words we need to show that $|N_2U| < |N_1S_1|$, considering N_1 is symmetric point to N_2 regarding line AM.

For contradiction, assume $|N_2U| \geq |N_1S_1|$. By the definition we have $|N_2U| = |N_2S_1| - r$, substituting this value to assumption, and adding r to both sides of the inequality, we get $|N_2S_1| \geq |N_1S_1| + r = |N_1N_2|$. Since ΔN_1AN_2 is an equilateral triangle, then the distance of any two points (excluding the edges) is strictly smaller than the length of a side of triangle, meaning $|N_2S_1| < |N_1N_2|$ which is a contradiction.

This means if R_2 moves toward point S_1, it will be r apart from S_1 sooner than R_1 reaches S_1. So it can use this time to move towards another point that is slightly before S_1 and communicate with R_1. □

Wlog let us assume that R_2 finds the exit at some point N on the perimeter of T, so Figure 12 shows the trajectory of R_2 assuming it finds the exit on 6 different segments of the triangle. Recall that $|MW_2| = r/2$ and W_3 is a fixed point so that if the exit is located there, R_1 will be r-intercepted exactly at point Q_1 right before starting its detour. For calculating this point, we have $0.5+|BQ_1| = |MW_3|+|W_3Q_1|-r$. Also by Cosine Rule we have $|W_3Q_1| = \sqrt{|BQ_1|^2 + (0.5 + |MW_3|)^2 - |BQ_1|^2 \cdot (0.5 + |MW_3|)}$. By solving these two equation we get $|MW_3| = \frac{2r \cdot |BQ_1| + r^2 + 1.5|BQ_1| + r}{|BQ_1| + 2r + 2}$.

We now split the trajectory of R_2 into 6 segments and show the critical point for each segment.

Lemma 3.15. *On segment MW_2 (see Figure 12a), point W_2 is the critical point and the evacuation time for this point is less than or equal to $y + 1.5$.*

Proof. Same as in Lemma 3.6, the evacuation time is $y+3|MN| \leq y+\frac{3}{2}r \leq y+1.5$ □

Lemma 3.16. *On segment W_2W_3 (see Figure 12b), point W_3 is the critical point.*

Proof. For any arbitrary point on segment W_2W_3 (except the two endpoints), based on Lemma 3.12, R_1 will be r-intercepted at some point Q while moving on segment Q_1J_1, and according to Lemma 3.3, since $\beta > \pi/2$, there exists another point in the direction of movement of R_2 such that placing the exit at that point will result in a higher evacuation time. We conclude that in segment W_2W_3, point W_3 is the critical point. □

Lemma 3.17. *On segment W_3C (see Figure 12c), point C is the critical point.*

Proof. Since $\beta > \pi/2$, the proof is analogous to the proof of the previous lemma. □

Lemma 3.18. *On segment CQ_2 (see Figure 12d), point C is the critical point.*

Proof. Suppose the exit is located at some point N, then based on Lemma 3.13 R_1 will be r-intercepted while moving on segment Q_1J_1 at some point Q. Note that $\angle QQ_2N > \pi/6$, hence $\beta+\gamma < 5\pi/6$. On the other hand we have $\angle QQ_2N+\beta+\gamma = \pi$. It is easy to see that $2cos(\beta) + cos(\gamma)$ is always greater than 1. Based on Lemma 3.3, there exists another point in the opposite direction of R_2 which yields a larger evacuation time if the exit is located there. □

Lemma 3.19. *On segment Q_2S_2 (see Figure 12e), the point right after Q_2 is the critical point.*

Proof. Let the exit be at an arbitrary point N on segment Q_2S_2 and the interception point be Q on side BA. We know that $\beta+\gamma = \frac{2\pi}{3}$, and for all points on segment CS_2, the angle β is between 0 and $\pi/3$. Then we get $2cos(\beta)+cos(\gamma) = 2cos(\beta)-cos(\frac{\pi}{3}+\beta)$ which is strictly greater than one. Hence based on Lemma 3.3, moving the exit point in the opposite direction of the movement of R_2 will result in higher evacuation time. We can conclude that in this segment, the point right after Q_2 is the critical point. □

Lemma 3.20. *On segment S_2A (see Figure 12f), the evacuation time is independent of the location of the exit point and is equal to $t_{S_1} + r$.*

Proof. Let the exit be at some point N on this segment and point Q be the position of R_1 when the exit is found by R_2. Recall that in this segment, the agents are in communication range and as soon as R_2 finds the exit, it will send the location to R_1. Since both agent are at the same distance to A, if the exit is found before A, agent R_1 will move to the exit on a line parallel to edge BC making a smaller equilateral triangle at top of T. It is obvious that $QN = QA$ end evacuation time will be $t_{S_1} + S_1Q + QN = t_{S_1} + S_1A = t_{S_1} + r$. □

We will now give a value for $E_{\mathcal{A}_2}(2, r)$.

r	BQ_1	Evac Time
0.10	0.61227	2.27422
0.20	0.53870	2.19427
0.30	0.47134	2.12651
0.40	0.40616	2.06593
0.50	0.34140	2.01050
0.60	0.27608	1.95926
0.70	0.20960	1.91169
> 0.7374049	N/A	N/A

Table 1: Evacuation of 2 Agents using only one detour

Theorem 3.21. *Let*

$$t_1 = y + 0.5 + |BQ_1| + |Q_1C| \text{ and}$$

$$t_2 = y + 0.5 + |BQ_1| + |Q_1J_1| + |J_1P_1| + |P_1Q_1| + |Q_1Z| + |ZQ_2|$$

whereas Z is a point that if the exit is located right after Q_2, agent R_1 will be r-intercepted at Z. Then $E_{\mathcal{A}_2}(2, r) = max\{t_1, t_2\}$.

Proof. Based on Lemmas 3.15 to 3.18, point C is a critical point on segments MC and CQ_2. We also know that the point after Q_2 is a critical point on segment Q_2A based on Lemmas 3.19 and 3.20, therefore the maximum evacuation time is achieved when the exit is located at either of these points. □

Placing the exit at point C will result in evacuation time of $t_1 = y + |MB| + |BQ_1| + |Q_1C|$. On the other hand, evacuation time if the exit is located right after point Q_2 is $t_2 = y + |MB| + |BQ_1| + |Q_1J_1| + |J_1P_1| + |P_1Q_1| + |Q_1Z| + |ZQ_2|$. By Theorem 3.21 we have $E_{\mathcal{A}_2} = max(t_1, t_2)$. By increasing $|BQ_1|$, time t_1 increases and Similarly, decreasing length of BQ_1, increases t_2. Best value for $|BQ_1|$ is obtained when equation $t_1 = t_2$ is satisfied. Because of the complexity of the equations, the final result is in the form of $f(r, BQ_1) = g(r, BQ_1)$ and BQ_1 cannot be shown as a function of r. The results for different values of r are shown in Table 1.

3.2.2 No Detour for Large Values of r for $\mathcal{A}_2$

The placement of point Q_1 is a critical part of the algorithm, meaning it should yield an optimal result. Moving the point higher toward A, and time t_1 increases while moving it toward point B, will result in increment of t_2. Hence Q_1 is a unique point placed between B and A.

Consider a large r, in this specific example we assume $r = 0.9$. We know that $|Q_1J_1| = 1/2*(\sqrt{|BQ_1|^2 + 1 - |BQ_1|} - |BQ_1| - r)$. If we draw the graph of $|Q_1J_1|$ for that specific r, Figure 14, we realize that $|Q_1J_1|$ would not be valid if $|BQ_1|$ is greater that some specific value. Considering what was mentioned about the uniqueness of point Q_1, making a detour does not improve the evacuation time for large values of r.

In other words, if we have a big r, when R_1 reaches the point Q_1 that it is optimal to make a detour, that point is so close to point C, so that both agents could communicate with each other directly and no detour is needed. We formalize this in the lemma below:

Lemma 3.22. *Adding a detour for* $r > 0.7374049$ *increases the maximum evacuation time.*

Proof. Values r and $|Q_1J_1|$ have negative correlation. Increasing r will result in decrement of $|Q_1J_1|$ up until a point that $|Q_1J_1|$ is equal to zero. At this point we would have $r = \sqrt{|BQ_1|^2 + 1 - |BQ_1|} - |BQ_1|$. By substituting this value in $f(r, |BQ_1|) = g(r, |BQ_1|)$ and solving that we get the values of 0.1843512 and 0.7374049 for $|BQ_1|$ and r respectively. If we increase r, we get negative value for Q_1J_1 which is invalid. □

3.2.3 Equal Travel with Two or More Detours

We now show that for some values of r, further improvement in evacuation time can be achieved by making more detours. Let's skip the execution of $\mathcal{A}_2$ up until the point that R_1 and R_2 reach vertices B and C respectively, assuming non of the agents have found the exit so far. The problem which remains will be a triangle with two unexplored sides of length 1, call this $\mathcal{P}_1$. Now consider the moment that the two agents finish their detour and get back to points Q_1 and Q_2. Call the remaining problem $\mathcal{P}_2$. It is obvious that $\mathcal{P}_2$ is similar to problem $\mathcal{P}_1$ with the only difference of

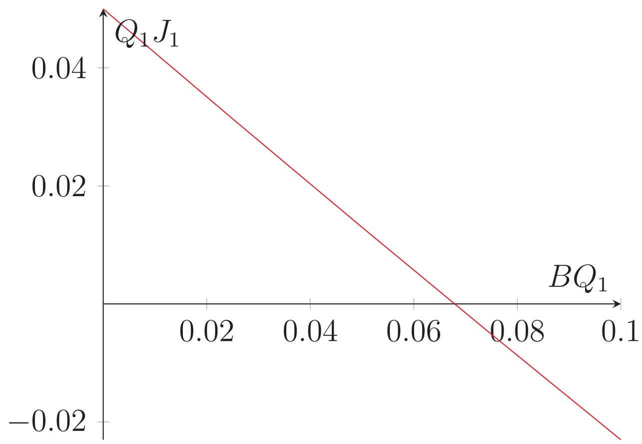


Figure 14: Plot $Q_1J_1 = f(BQ_1)$ where $r = 0.9$

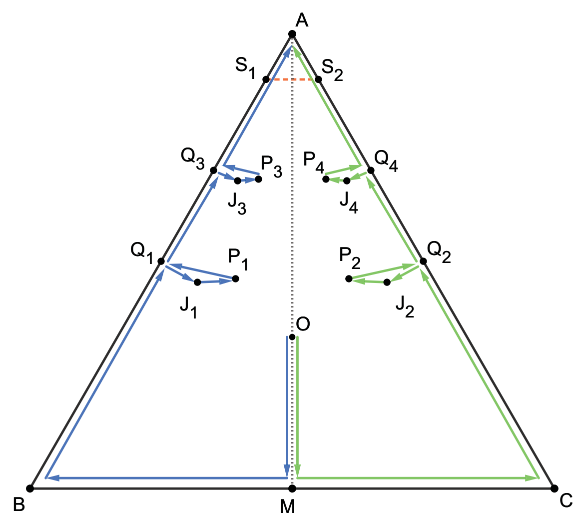


Figure 15: Evacuating two agents using two detours

side of the triangle and due to this difference, the ratio of r to length of the side of the triangle increases with every instance of detour until the point that no more detour is needed. The trajectory of two agents in two detour model is shown in Trajectory 3. Note that J_3 is located on line Q_3Q_2 in a way that $|Q_1Q_3| + |Q_3J_3| = |Q_2J_3| - r$ and P_3 is located on line J_3Q_4 in a way that $|Q_3J_3| + |J_3P_3| = |Q_4P_3| - r$. Points J_4 and P_4 are located symmetrically with those of points J_3 and P_4 respectively, with respect to line AM and definitions of points J_1, J_2, P_1 and P_2 remains the same as their definition in the previous section.

r	$\|BQ_1\|$	$\|Q_1Q_3\|$	Evac Time
0.10	0.594418	0.20558	2.25424
0.20	0.530613	0.18272	2.18584
0.30	0.467966	0.15949	2.12325
0.40	0.405201	0.13563	2.06506
> 0.472505	N/A	N/A	N/A

Table 2: Evacuation of 2 Agents using two detours

Trajectory 3.

R_1 follows the trajectory $:< O, M, B, Q_1, J_1, P_1, Q_1, Q_3, J_3, P_3, Q_3, A >$

R_2 follows the trajectory $:< O, M, C, Q_2, J_2, P_2, Q_2, Q_4, J_4, P_4, Q_4, A >$

We denote Algorithm 1 with respect to Trajectory 3 by $\mathcal{A}_3$. In the case of two detours, see Figure 15, similar to the proof of Theorem 3.21, it can be shown that there exists three critical points as C, the point right after Q_2 and the point right after Q_4. The evacuation times if we place the exit at these points will be as follows:

- $t_1 = |MB| + |BQ_1| + |Q_1C|$
- $t_2 = |MB| + |BQ_1| + |Q_1J_1| + |J_1P_1| + |P_1Q_1| + |Q_1Q_3| + |Q_3Q_2|$
- $t_3 = |MB| + |BQ_1| + |Q_1J_1| + |J_1P_1| + |P_1Q_1| + |Q_1Q_3| + |Q_3J_3| + |J_3P_3| + |P_3Q_3| + |Q_3Z| + |ZQ_4|$

Obviously $E_{\mathcal{A}_3}(2, r) = max\{t_1, t_2, t_3\}$. By solving the equations of $t_1 = t_2$ and $t_1 = t_3$ we get the values for $|BQ_1|$ and $|Q_1Q_3|$. Again because of the complexity of the equations, we were not able to provide any function for those values based on r, instead we solved the equations numerically. The results are shown in Table 2.

In [13] it is shown that in the face-to-face model, adding additional detours always improve the evacuation time, though the improvement obtained by successive detours decreases rapidly. In contrast, for large values of r, adding even one detour does not improve the evacuation time. Similar to Lemma 3.22, it can be shown that the second detour is not needed for $r > 0.472505$

3.3 A New Detour Trajectory

In this section we propose and analyse a different detour, namely Trajectory 4 and the two detour version of it, Trajectory 5, which yields an improved evacuation time compared with $\mathcal{A}_2$ and $\mathcal{A}_3$. Trajectories 4 and 5 used in Algorithm 1 will be referred to as $\mathcal{A}_4$ and $\mathcal{A}_5$, respectively. In contrast to the Strategies $\mathcal{A}_2$ and $\mathcal{A}_3$ in which detours were not beneficial for r greater than 0.7374049 and 0.472505, respectively, the new detour is applicable for large values of r as well.

3.3.1 $\mathcal{A}_4$ Strategy

In our new strategy $\mathcal{A}_4$, the trajectories of both agents are symmetric with respect to line AM. Therefore wlog, we assume that R_2 finds the exit and analyse the trajectory of R_1.

As in *Equal Travel with One Detour* which was described in the previous section, agent R_1 at some point, abandons exploring the perimeter of T and moves toward vertex C in order to decrease the evacuation time, in case the exit is located at C. By reaching a certain point, it finds out that the exit was not at C and changes the direction of the detour in order to facilitate the evacuation if the exit is found in some other segments. After the detour is finished, if it is not notified by the other agent, it returns to the same point from which it started the detour.

Trajectory 4.

$$R_1 :< O, M, B, Q_1, J_1, P_1, Q_1, A >$$

$$R_2 :< O, M, C, Q_2, J_2, P_2, Q_2, A >$$

We fix the point Q_1 on side AB, the exact value will be specified later (see Figure 16). Point J_1 is on segment Q_1C such that it satisfies the following equation:

$$|BQ_1| + |Q_1J_1| = |J_1C| - r$$

Similar to $\mathcal{A}_2$, the placement of point J_1 makes sure that if the exit is located at C, agent R_1 can be r-intercepted at J_1. In contrast to the previous algorithm, we

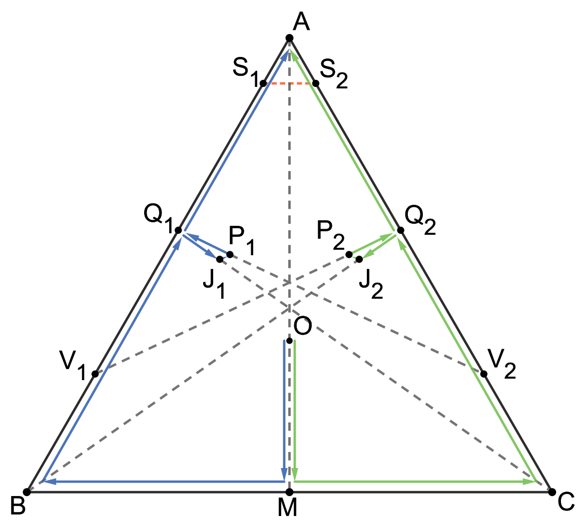


Figure 16: Trajectories of agents in $\mathcal{A}_4$

introduce two new points V_2 located on the side AC and P_1 on segment Q_1V_2 in a way that if the exit is located at V_2, then R_1 will be r-intercepted when it is at P_1, i.e.

$$|BQ_1| + |Q_1J_1| + |J_1P_1| = |CV_2| + |V_2P_1| - r \tag{1}$$

If the exit is located right after V_2, it is obvious that R_1 cannot be r-intercepted at P_1 and will only be notified when it finishes its detour. We have to make sure that if the exit is located right after V_2, the evacuation time does not exceed the time when the exit is at C. In order to optimize the worst-case evacuation time, we *equate* the evacuation time at C, and the time if the exit is immediately after V_2, that is

$$|Q_1J_1| + |J_1P_1| + |P_1Q_1| + |Q_1V_2| = |Q_1C| \tag{2}$$

Points V_1, Q_2, J_2 and P_2 are located symmetrically with points V_2, Q_1, J_1 and P_1 respectively, with respect to line AM.
The trajectories of the two agents are defined in Trajectories 4 and shown in Figure 16.

Our choices for the placement of all the mentioned points will result in the following lemmas.

Lemma 3.23. *If the exit is located at some point N on segment MC, then R_1 will be r-intercepted at or before reaching point J_1.*

Proof. The proof is similar to the proof of Lemma 3.12. □

Lemma 3.24. *If the exit is located at some point N on segment CV_2, then R_1 will be r-intercepted while moving on segment J_1P_1.*

Proof. Assume not, meaning that R_1 cannot be r-intercepted before reaching P_1. Then we have $|BQ_1|+|Q_1J_1|+|J_1P_1| < |CN|+|NP_1|-r$. Based on triangle inequality we have $|NP_1| < |NV_2|+|V_2P_1|$, substituting this in to the previous inequality results in

$$|BQ_1| + |Q_1J_1| + |J_1P_1| < |CN| + |NV_2| + |V_2P_1| - r = |CV_2| + |V_2P_1| - r$$

which contradicts Equation 1. □

Lemma 3.25. *Suppose the exit is located at some point N on segment V_2S_2 excluding the point V_2, then R_1 will be r-intercepted while exploring segment Q_1S_1.*

Proof. For the purpose of contradiction assume otherwise, meaning R_1 can be r intercepted at or before point Q_1 for some exit N which lies between V_2 and S_2. This implies

$$|BQ_1| + |Q_1J_1| + |J_1P_1| + |P_1Q_1| \geq |CN| + |NQ_1| - r = |CV_2| + |V_2N| + |NQ_1| - r$$

By triangle inequality we know $|V_2N| + |NQ_1| > |Q_1V_2|$, hence

$$|BQ_1| + |Q_1J_1| + |J_1P_1| + |P_1Q_1| > |CV_2| + |Q_1V_2| - r$$

By subtracting $|P_1Q_1|$ from both sides of the inequality we get

$$|BQ_1| + |Q_1J_1| + |J_1P_1| > |CV_2| + |Q_1V_2| - |P_1Q_1| - r = |CV_2| + |V_2P_1| - r$$

which contradicts Equation 1. □

We now split the trajectory of R_2 into different segments and for each segment, we will determine the critical point and propose the evacuation time for each critical points.

Lemma 3.26. *On segments MC and CV_2, vertex C is the critical point and the evacuation time will be at most $y + 0.5 + |BQ_1| + |Q_1C|$.*

Proof. Similar to the proof of Lemmas 3.15 to 3.18. □

Lemma 3.27. *On segment V_2Q_2, the evacuation time is maximum if the exit is located right after V_2. The evacuation is at most $y + 0.5 + |BQ_1| + |Q_1C|$*

Proof. Based on Lemma 3.3, the evacuation time is maximal in the immediate neighborhood of V_2 and is equal to $y + 0.5 + |BQ_1| + |Q_1J_1| + |J_1P_1| + |P_1Q_1| + |Q_1V_2|$. By substituting Equation 2 into this value, we get the desired results. □

Lemma 3.28. *On segment Q_2S_2, the evacuation time is at most $y + 0.5 + |BQ_1| + |Q_1J_1| + |J_1P_1| + |P_1Q_1| + |Q_1Z| + |ZQ_2|$ where Z is a point on segment AQ_1 such that $|Q_1Z| = |ZQ_2| - r$, i.e. point Z is the point that if the exit is located right after Q_2, agent R_1 will be r-intercepted at Z.*

Proof. Similar to the proof of Lemma 3.19. □

The evacuation time if the exit is located on segment S_2A is $t_{S_2} + r$ which is clearly less than the evacuation time if the exit is located at any point on segment Q_2S_2. Now combining the results of the previous lemmas, we obtain the evacuation time for $\mathcal{A}_4$ strategy.

Theorem 3.29. *Let $t_1 = y + 0.5 + |BQ_1| + |Q_1C|$ and $t_2 = y + 0.5 + |BQ_1| + |Q_1J_1| + |J_1P_1| + |P_1Q_1| + |Q_1Z| + |ZQ_2|$, where Z is a point on segment Q_1S_1 that R_1 will be r intercepted at, if the exit is located in the immediate neighborhood of Q_2. Then $E_{\mathcal{A}_4}(2, r) = max\{t_1, t_2\}$.*

Proof. Follows immediately from Lemmas 3.26 to 3.28. □

As in $\mathcal{A}_2$ increasing $|BQ_1|$ will increase t_1 and decreasing $|BQ_1|$ results in increase of t_2. Therefore, we conclude that the best value of $|BQ_1|$ is obtained when $t_1 = t_2$. The results for different values of r which are obtained by numerical calculations are shown in Table 3. Recall that for $r = 1$ an optimal evacuation algorithm is suggested in [19], so no further improvement is possible.

3.3.2 $\mathcal{A}_5$ Strategy

Consider the moment that both agents have finished making the detours and are back on the perimeter of T. This problem is identical to the problem in $\mathcal{A}_4$ when the side BC is fully explored and agents are at vertices B and C, the only difference is that the side of the triangle is smaller. Using the fact that the *new detour* can be

r	$E_{\mathcal{A}_4}(2,r)$	BQ_1	BV_1	Q_1J_1	J_1P_1
0.0	2.344339	0.672605	0.246081	0.105226	0.013629
0.1	2.234734	0.576648	0.260144	0.096381	0.021404
0.2	2.159036	0.504324	0.251749	0.080856	0.025759
0.3	2.096127	0.439301	0.234182	0.064424	0.028327
0.4	2.040527	0.377158	0.211072	0.048768	0.029459
0.5	1.990030	0.315998	0.183878	0.034679	0.029133
0.6	1.943552	0.254812	0.153253	0.022626	0.027195
0.7	1.900491	0.192977	0.119501	0.012930	0.023468
0.8	1.860477	0.130080	0.082708	0.005821	0.017777
0.9	1.823267	0.065825	0.042887	0.001470	0.009987

Table 3: Different values for $\mathcal{A}_4$ strategy

used for larger values of r, it is clear that another detour for the smaller triangle can further improve the evacuation time. However in order to achieve this goal, we have to re-balance all of the worst case evacuation times in T.

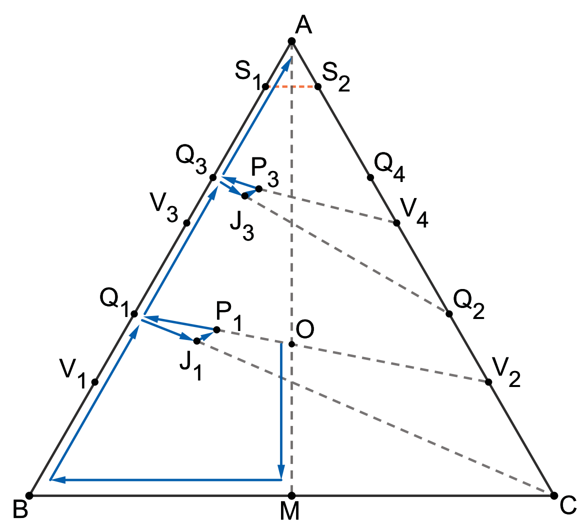


Figure 17: Trajectory of agent R_1 in $\mathcal{A}_5$

The trajectories of the two agents are defined in Trajectories 4 and shown in Figure 16. Note that the trajectory of R_2 is removed in order to improve the readability of the figure. For the same reason, the figure is not to scale, for example, in reality, the

r	$E_{\mathcal{A}_5}(2,r)$	BQ_1	BV_1	Q_1Q_3	Q_1V_3
0.0	2.286183	0.622816	0.323967	0.253695	0.092817
0.1	2.209896	0.553542	0.311850	0.218006	0.110798
0.2	2.147476	0.492745	0.285711	0.193098	0.107798
0.3	2.090640	0.433381	0.256494	0.168858	0.099266
0.4	2.037985	0.374193	0.225445	0.144368	0.087858
0.5	1.988921	0.314597	0.192760	0.119637	0.074675
0.6	1.943118	0.254214	0.158368	0.094869	0.060372
0.7	1.900349	0.192764	0.122105	0.070304	0.045448
0.8	1.860445	0.130026	0.083761	0.046174	0.030114
0.9	1.823264	0.065820	0.043129	0.022681	0.014830

Table 4: Different values for $\mathcal{A}_5$ strategy

value of Q_3J_3 is much smaller than Q_1J_1. Points V_1, Q_2, J_2, P_2, V_3, Q_4, J_4 and P_4 are located symmetrically with points V_2, Q_1, J_1, P_1, V_4, Q_3, J_3 and P_3 respectively, with respect to line AM.

Trajectory 5.

$$R_1 :< O, M, B, Q_1, J_1, P_1, Q_1, Q_3, J_3, P_3, Q_3, A >$$

$$R_2 :< O, M, C, Q_2, J_2, P_2, Q_2, Q_4, J_4, P_4, Q_4, A >$$

Recall that by $\mathcal{A}_5$ we refer to Algorithm 1 with respect to Trajectory 5. Let t_1 be the evacuation time if the exit is located at C, and let t_2 and t_3 be the evacuation times if the exit is located at the immediate neighborhood of points Q_2 and Q_4, respectively. The exact values of $|BQ_1|$ and $|BQ_3|$ are obtained by solving the equations $t_1 = t_2$ and $t_1 = t_3$. As before, the equations were solved numerically and the results are shown in Table 4.

3.4 Comparison of $\mathcal{A}_1$,$\mathcal{A}_2$,$\mathcal{A}_3$,$\mathcal{A}_4$ and $\mathcal{A}_5$

The evacuation time of all five variations of the proposed algorithms is shown in Table 5. As mentioned in the previous chapter and [13], the improvement obtained by the second detour is significantly smaller than the improvement gained from the first detour.

	Evacuation times				
r	$\mathcal{A}_5$	$\mathcal{A}_4$	$\mathcal{A}_3$	$\mathcal{A}_2$	$\mathcal{A}_1$
0.00	2.286183	2.34433	2.3367	2.3838	2.78867
0.10	2.209896	2.23473	2.25424	2.27422	2.53867
0.20	2.147476	2.15903	2.18584	2.19427	2.36010
0.30	2.090640	2.09612	2.12325	2.12651	2.22617
0.40	2.037985	2.04405	2.06506	2.06593	2.12200
0.50	1.988921	1.99003	N/A	2.01050	2.03867
0.60	1.943118	1.94355	N/A	1.95926	1.97049
0.70	1.900349	1.90049	N/A	1.91169	1.91367
0.80	1.860445	1.86047	N/A	N/A	1.86559
0.90	1.823264	1.82326	N/A	N/A	1.82438

Table 5: Evacuation times of 2 Agents algorithms.

3.5 Lower Bound for Evacuating 2 Agents

In this section we prove a lower bound for evacuation by two agents . The proof is essentially the same as the proof of the lower bound in [13] for the case $r = 0$, but needs to take into account the ability of agents to communicate at distance r. We give the entire proof here for completeness. First we need to generalize the Meeting Lemma used in [13]. We say two points have opposite positions if one point is a vertex of T and the other point is located on the opposite edge of that vertex.

Lemma 3.30 (Generalized Meeting Lemma)**.** *Assume that $p_1, p_2 \in T$ and they have opposite positions, e.g. points M_3 and C in Figure 18. For any algorithm in which one of the agents visits p_1 at time $t' \geq 0.5 + y$ and the other visits p_2 at time t such that $t' < t < 0.5 + 4y - r$, the two agents cannot communicate any information between time t' and t.*

Proof. Suppose the two agents exchange information between time t' and t. They have to get close to each other, in order to communicate.

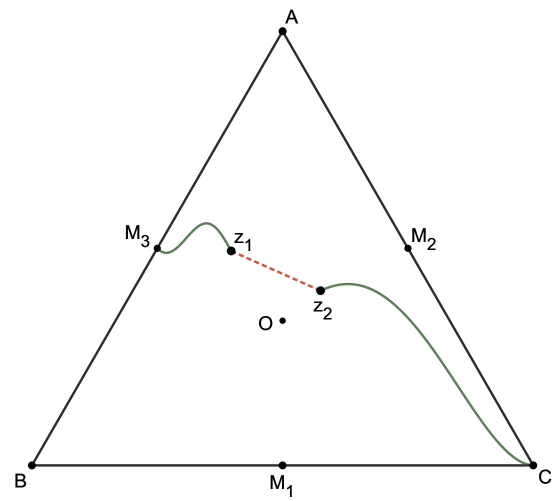


Figure 18: Modified Meeting Lemma

Then there exists a time t_z with $t' \leq t_z \leq t$ such that R_1 is at point z_1 and R_2 is at z_2 at time t_z, and $|z_1z_2| \leq r$. Since p_1 and p_2 have opposite positions, $|p_1p_2| \geq 3y$. Therefore $|p_1z_1|+|z_1z_2|+|z_2p_2| \geq 3y$. On the other hand, we know that $t_z - t' \geq |p_1z_1|$ and $t - t_z \geq |p_2z_2|$. Combining these facts together, we obtain:

$$h \leq |p_1z_1| + |z_1z_2| + |z_2p_2| \leq t_z - t' + |z_1z_2| + t - t_z = t - t' + r$$

$$t - t' + r < 0.5 + h + y - r - t' + r \leq 0.5 + h + y - 0.5 - y = h$$

which is a contradiction. □

Using a similar arguments used in [13] we now establish the lower bound for the case $r \geq 0$ for two agents.

Theorem 3.31. *Two agents are at a centroid of an equilateral triangle with sides 1. The evacuation time for two agents of transmission range $r \leq 3y - 0.5$ positioned at the centroid of a triangle with sides one is at least $1 + 4y - r$.*

Proof. For the purpose of contradiction assume there exists algorithm $\mathcal{A}$ such that $E_{\mathcal{A}}(2, r) < 1 + 4y - r$. Let us focus on the set of points $S = \{A, B, C, M_1, M_2, M_3\}$. We give an Adversary Argument. There exists some input I in which the exit is the last of the point in S visited by an agent. Suppose time t is the time that the fifth point from the set S is visited and v_1 through v_6 be the order that points are visited. Wlog we assume that v_5 is visited by R_1. Since at time t, the fifth vertex is visited, then 3 points must have been visited by one of the agents and $t \geq y + 1$. On the other hand since the algorithm should satisfy $E_{\mathcal{A}}(2, r) < 1 + 4y - r$, then $t < 0.5 + 4y - r \leq 0.5 + 4y$ since R_1 needs extra time 0.5 to get to the sixth vertex. We now examine the following exhaustive cases based on whether v_5 is a midpoint or a vertex.

Case 1. Point v_5 is a vertex of T: Wlog assume that v_5 is C. If v_6 is one of A, M_3 or B, then it takes at least h for R_1 to evacuate the triangle and $E^*(2, r) \geq t + h \geq 1 + 4y$ which is a contradiction. We conclude that the v_6 should be either M_1 or M_2. Note that R_1 could have visited at most one of A, M_3 and B by time t, hence R_2 should have visited at least two points of A, M_3 and B. Assume v is the second vertex of the set of A, M_3 and B visited by R_2 at time t'. Clearly $t' \geq 0.5 + y$. By the Generalized Meeting Lemma the two agents cannot communicate between t' and t on input I.

Now consider input I' in which the exit is located at v. On this input R_1 andR_2 behave identical to input I until time t'. After this time R_2 may try to r-intercept R_1 but by Modified Meeting Lemma we know that the r-interception does not occur before time t. Hence R_1 has to travel at least h to get to the exit which indicates that evacuation time will be at least $1 + y + h = 1 + 4y$ on input I', a contradiction.

Case 2. v_5 is a midpoint of a side of T, and v_6 is another midpoint: Wlog we assume that v_5 is M_2 and v_6 is M_3. If R_1 visits two vertices before arriving at M_2 then $E^*(2, r) \geq 2y + 1.5$ which is a contradiction. We conclude that R_2 must have visited 2 vertices before t. It is obvious that R_2 cannot visit the second of these two vertices sooner than time $2y + 1$. If the second vertex is B then the adversary places the exit at M_2 and if the second vertex is C, it will place the exit at M_3. In both cases $E^*(2, r) \geq 2y + 1 + 3y > 1 + 4y$ which is a contradiction. We conclude that the second vertex visited by R_2 must be A.

Observe that if M_1 is visited by R_2, then $E^*(2, r) \geq y + 2 > 1 + 4y$. Therefore R_1 should visit M_1 as well as either B or C before arriving at M_2. Let P be the second point from set S visited by R_1 at time t_1. Clearly $t_1 \geq y + 0.5$. On the other hand, R_2 must visit A before time $0.5+4y-r$. Let this time be t_2. Clearly $t_1 < t_2$, since if not, the time R_1 gets to M_2 will be at least $t_2 + 0.5 \geq 2y + 1.5$ which is a contradiction. By the modified meeting lemma, R_1 and R_2 cannot exchange information between t_1 and t_2. Now consider input I' in which the exit is located at P. Agent R_2 has the same behaviour until it reaches point A at time $t_2 \geq 2y + 1$ and has to travel at least $3y$ to get to the exit. Hence $E^*(2, r) \geq 2y + 1 + 3y > 1 + 4y$. A contradiction.

Case 3. v_5 is a midpoint of a side of T, and v_6 is a vertex: Wlog assume v_5 is M_2, if v_6 is B then $E^*(2, r) \geq 4y + 1$ which is a contradiction. We conclude that v_6 is either A or C. Wlog we assume v_6 is A. If a single agent visits both B and C, it takes time at least $2y + 2$ to get to A, hence B and C should be visited by different agents. Now we consider 2 cases: R_1 visits C and R_2 visits B; and R_1 visits B and R_2 visits C.

Suppose R_1 visits C. First observe that R_1 cannot also visit M_3 before reaching point M_2 as doing so will result in $E^*(2, r) \geq 4y + 1$ which is a contradiction. Therefore M_3 should be visited by R_2. If M_1 is visited by R_2, Lemma 3.33 assures that $E^*(2, r) \geq 4y + 1$ and if M_1 is visited by R_1, then Lemma 3.34 assures the same thing.

Now suppose R_1 visits B before reaching point M_2 and R_2 visits C. It is obvious that R_1 cannot visit both M_3 and M_1 as by doing so, it will take at least $y+1.5 > 4y+0.5$ to reach M_2. Lemma 3.35 now assures that $E^*(2, r) \geq 4y+1-r$

.

□

For Lemmas 3.33 through 3.35, we assume that $v_5 = M_2$ and $v_6 = A$, and R_1 visits M_2 at time $1 + y \leq t_{M_2} < 4y + 0.5 - r$. We use the following observation from [13].

Observation 3.32. *[13] Let p be a point on the boundary. If at time $1 + 4y - |Ap|$, both A and p are unvisited, then $E^*(2, r) \geq 1 + 4y$*

Lemma 3.33. *If R_2 visits B, M_3 and M_1, and R_1 visits C and M_2, then $E_{\mathcal{A}}(2, r) \geq 1 + 4y - r$.*

Proof. [13] First observe that if B is not visited before M_3 and M_1, then $t_B \geq 0.5 + y$ and $t_{M_2} \geq 1 + y$, hence by the Modified Meeting Lemma, R_1 and R_2 cannot communicate between t_B and t_{M_2}. Thus if the exit is located at B, it will take at

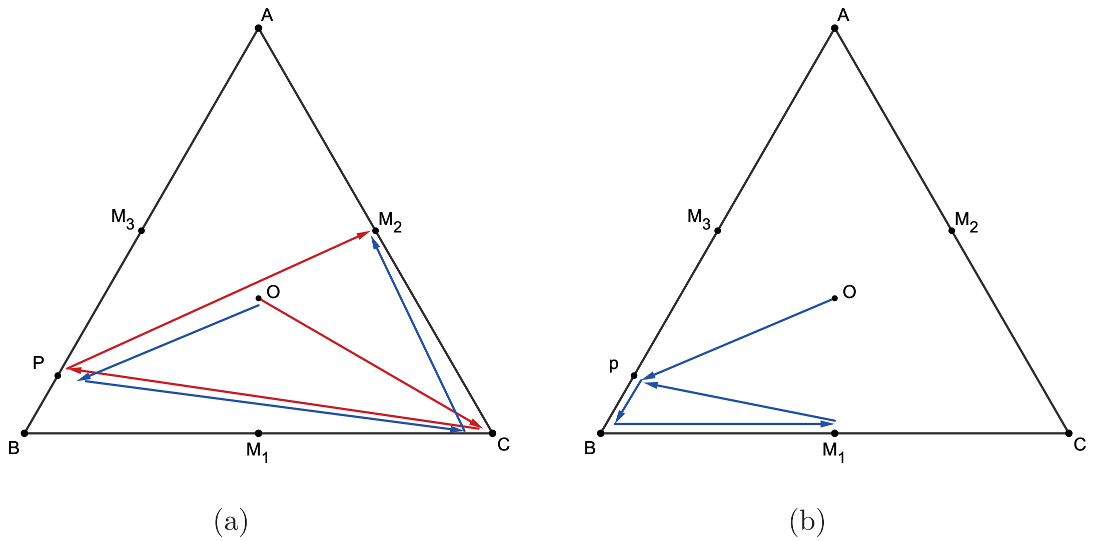


Figure 19: Illustration of (a) possible trajectories of R_1 and (b) trajectory of R_2 in support of Lemma 3.33

least $t_{M_2} + h \geq 1 + 4y$ for R_1 to get to the exit, a contradiction. If R_2 visits B, M_3 and M_1 in that order, then at time $t_{M_1} - \varepsilon$ both A and M_1 are unvisited, so by Observation 3.32, $E_{\mathcal{A}}(2, r) \geq t_{M_1} - \varepsilon + h > 1 + 4y$, a contradiction. We conclude that R_2 should visit B, M_1 and M_3 in that order.
Let point p be the closest point to B on segment BM_3 that is not visited by R_2 before visiting M_1. The earliest time for R_2 to visit p is $|Op| + |Bp| + |BM_1| + |M_1p|$, see Figure 19b. It can be shown that for any point m on segment BM_3, this time is more than $1 + 4y - |Am|$, therefore it also applies to p. On the other hand, R_1 cannot visit p on time: if it visits p before C (the blue trajectory in Figure 19a), we have $t_{M_2} \geq |Op| + |pC| + |CE| \geq |OM_3| + |M_3C| + |CM_2| = 4y + 0.5$, and if it visits p after visiting C (the red trajectory in Figure 19a), we have $t_{M_2} \geq |OC| + |Cp| + |pM_2| \geq |OC| + |CM_3| + |M_3M_2| = 5y + 0.5$. Hence no agent can visit p before time $1 + 4y - |Ap|$ and based on Observation 3.32, $E_{\mathcal{A}} \geq 1 + 4y$. □

Lemma 3.34. *If R_2 visits B, M_3, and R_1 visits C, M_1 and M_2, then $E_{\mathcal{A}}(2, r) \geq 1 + 4y - r$.*

Proof. Recall that $t_{M_2} \geq 1 + y$. If R_1 and R_2 don't meet between t_B and t_{M_2} then by placing the exit at B, it will take at least $1 + 4y$ for R_1 to get to the exit. So we conclude that they should communicate with each other between t_B and t_{M_2}. If

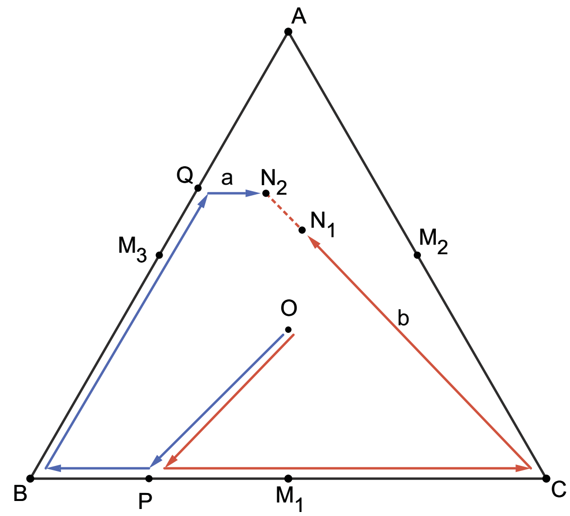


Figure 20: Trajectories when R_2 visits B before M_3 and R_1 visits M_1 before C

R_2 visits M_3 before B, then $t_B \geq 0.5 + y$ and since $t_{M_2} < 0.5 + 4y - r$, by the Generalized Meeting Lemma they could not exchange any information between t_B and t_{M_2}. Therefore R_2 must visit B before M_3.

Now if R_1 visits C before M_1, using a similar argument as in Lemma 3.33, it can be verified that an unvisited point P exists on segment CM_2 at time $1 + 4y - |AP|$. It follows from the Observation 3.32 that $E_{\mathcal{A}}(2, r) \geq 1 + 4y$. Hence M_1 should be visited before C.

Observe that $t_C \geq 0.5 + y$ and $t_{M_3} < 0.5 + 4y - r$, so by the Generalized Meeting Lemma R_1 and R_2 cannot exchange information between time t_C and t_{M_3}. We conclude that they should exchange information after t_C and after t_{M_3}. At this point, the whole segment BM_1 must be visited; because if not, before time $t_{M_3} + 3y/2$ there exist an unvisited point on BM_1 segment with distance at least $3y$ from A and by Observation 3.32, $E_{\mathcal{A}}(2, r) \geq t_{M_3} + 9y/2 \geq 0.5 + 13y/2$. Thus there must be a point P on segment BM_1 such that BP is explored by R_2 and CP is explored by R_1. Thus $t_B \geq |OP| + |BP|$ and $t_C \geq |OP| + (1 - |BP|)$. Suppose the exit is located at C, if the agents don't communicate before R_2 reaches A, then clearly $E_{\mathcal{A}}(2, r) \geq 1 + 4y$. So they have to exchange information before R_2 reaches A. Let Q be the last point visited by R_2 on segment AB, before either it gets r-intercepted by R_1 or moves inside T in order to get closer to the other agent. Let the interception points be N_1 and N_2, see Figure 20, and let $a = |QN_2|$ and $b = |N_1C|$. Clearly points Q and N_2 will merge and $a = 0$, if R_2 is r-intercepted on segment AB.

Note that R_1 reaches point N_1 at the same time that R_2 reaches point N_2 and they are r-apart. Observe that:

For R_2 to get to the exit at C in time, it must be that $t_{N_2} + r + b < 1 + 4y - r$ and since $t_{N_1} \geq t_C + b$, we obtain:

$$b < \frac{1 + 4y - 2r - t_C}{2} \leq \frac{1 + 4y - 2r - |OP| - 1 + |BP|}{2} \tag{3}$$

Now let Q' be a point infinitesimally close to Q on segment QA and visited at time $t_{Q'}$. Then by Observation 3.32, we have $t_{Q'} < 1 + 4y - r - |AQ'|$. Also we have $t_{Q'} \geq t_{N_2} + a \geq t_B + |BQ'| + 2a$, which implies that

$$a < \frac{1 + 4y - r - |AQ'| - |BQ'| - t_B}{2} \leq \frac{1 + 4y - r - 1 - (|OP| + |BP|)}{2} \tag{4}$$

Note that $a + b + r \geq 3y$, yet from inequalities 3 and 4 we get:

$$a + b + r < 4y - |OP| - r/2 \leq 3y$$

which is a contradiction. We conclude that these r-interception points do not exist and if the exit is located at C, agent R_2 cannot get to the exit point in time.

□

Lemma 3.35. *If R_2 visits C and at least one of points $M1$ or M_3, and R_1 visits B and at most one of points M_1 or M_3, then $E_{\mathcal{A}}(2, r) \geq 1 + 4y - r$.*

Proof. The proof is identical to the proofs of Lemmas 7 through 9 in [13]. □

Chapter 4

Evacuating Three Agents

In [13] an evacuation algorithm for 3 agents is proposed for $r = 0$ with evacuation time of 2.08872. In this algorithm T is divided into 4 segment (see Figure 21), three of these segments are assigned to individual agents and after they finish exploring these 3 segments, they move inside T in a manner that they will be at the centroid at the same time in order to exchange information. After the meeting, if the exit is not found, all three agents will search the fourth segment together. We refer to this segment as the *unexplored part* from now on.

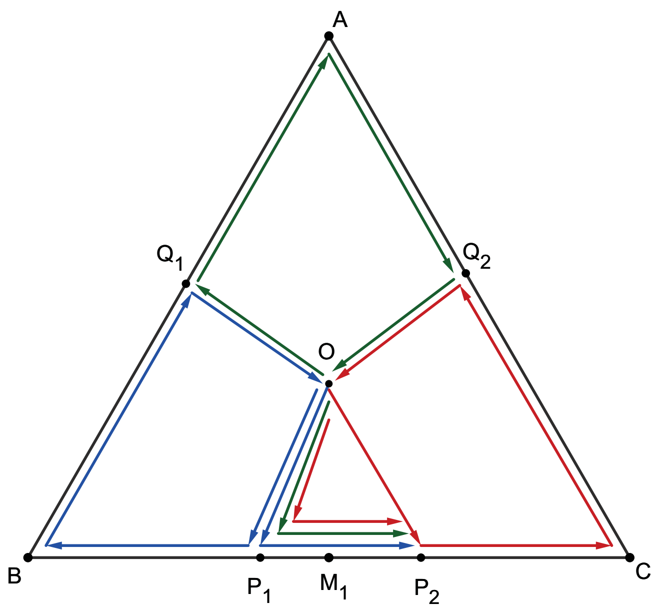


Figure 21: Trajectories of three agents in [13]

In [13] it is shown that a better result is achieved if $BQ_1 \neq CQ_2$ and $M_1P_1 \neq M_1P_2$ rather than these values to be equal. However in this chapter we will show that this will not always be the case and placing the unexplored part in the middle of side BC will yield a better result for some specific values of r.

Another approach used in wireless model in [19] is to partition the side BC in to three segments and assign each segment to one agent, see Figure 22. After each agent finishes searching its segment, two of the agents continue to explore sides AB and AC while the third agent moves to the unexplored vertex. The partitioning is done in a way that all three agents will arrive at vertex A at the same moment.

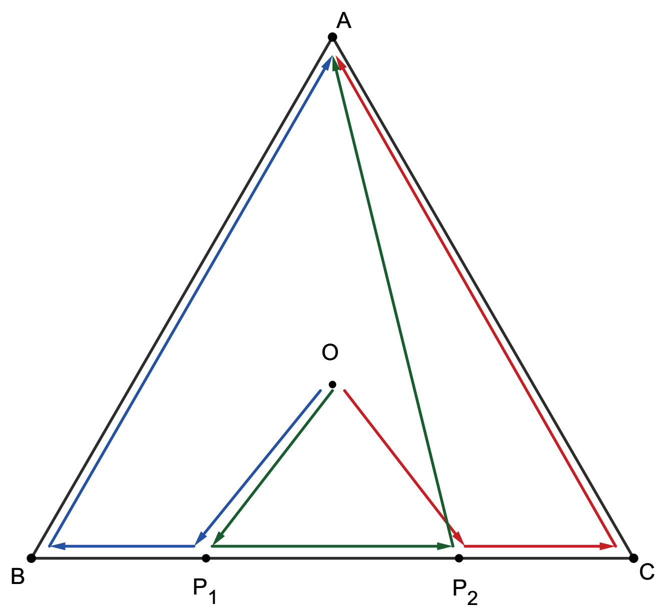


Figure 22: Trajectories of three agents in a wireless model

In this chapter, we will consider adaptation of both approaches, for the case $r > 0$. Since three agents are going to evacuate the triangle, the trajectory of at least one of them cannot be symmetric with respect to line AM.

Another thing to keep in mind is that when we have more than two agents, one of them can act as a *relay* and extend the range of an agent to send the message containing the location of the exit. Also since the trajectories of the agents are known to all of them before the execution of the algorithm, they know the moment and location in which an agent can act as a relay and expanding the effective communication range.

Three different trajectories are going to be suggested in this chapter, but they all follow a general schema described in Algorithm 2. The <found> and <msg_recd> are one bit flags, implemented in each of the agents. Both of them are initialized to *false* at the start of the algorithm.

Algorithm 2 Evacuation Algorithm for Three or more Agents

```
function Exploration
    found← false
    while not<found> and not<msg_recd> do
        move along the predetermined trajectory
    end while
    action
end function
function Action
    if found then
        P ← current location
        while all the other agents are not in effective communication range do
            continue moving on the trajectory
        end while
    end if
    broadcast < P >
    go to P and exit
end function
```

In Section 4.1 and 4.2 we will give two adaptations of *Equal-Travel Early-Meeting Algorithm* in [13] and in Section 4.3 we will discuss a modified version of wireless algorithm in [19]. Later on in this chapter, we will make a comparison of the three types of algorithms and determine which one is better for specific values of r.

4.1 Evacuating 3 Agents from a Triangle Using X3Cv1

The first adaptation of *Equal Travel Early Meeting* method proposed in [13] is described in Trajectory 6 and shown in Figure 23. Points M_1, M_2 and M_3 are midpoints of the side of T. By X3Cv1 we mean Algorithm 2 that uses Trajectory 6. The whole

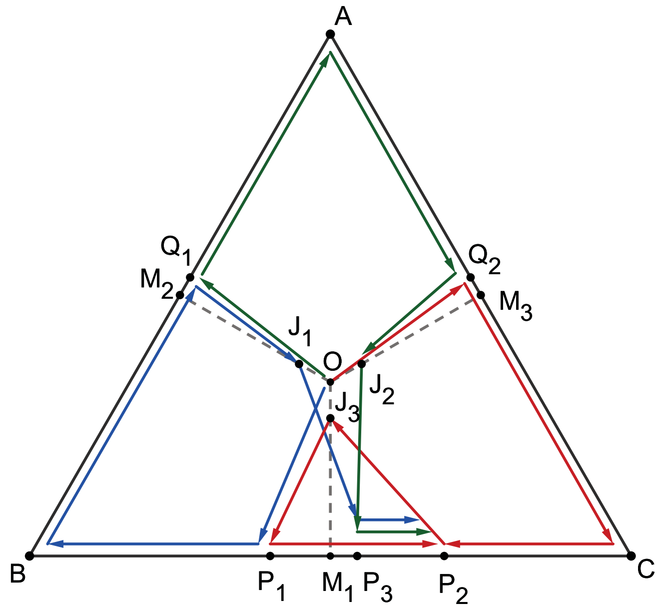


Figure 23: Evacuating three agents using X3Cv1

idea is to have an exchange of information after a portion of T is explored, and only if the exit was not found, move to the unexplored part of T.

Trajectory 6.

$$R_1 \text{ follows the trajectory: } < O, P_1, B, Q_1, J_1, P_3, \text{ wait for } R_3, P_2 >$$

$$R_2 \text{ follows the trajectory: } < O, Q_1, A, Q_2, J_2, P_3, \text{ wait for } R_3, P_2 >$$

$$R_3 \text{ follows the trajectory: } < O, Q_2, C, P_2, J_3, P_1, P_2 >$$

We partition the perimeter of T into 4 segments. Three of the segments are assigned to individual agents for exploration. After the exploration of these segments are finished, they move inside T to the *extended meeting points* J_1, J_2 and J_3. These points have the following properties:

- They are at distance r from each other.
- Their distance to point O are equal.
- Points J_1, J_2 and J_3 are located on line segments OM_2, OM_3 and OM_1 respectively.

After the information is exchanged, if the exit is not found, they move toward the fourth segment. At this point, due to the difference between the distance of each agent to point P_1, it is better if they don't move together. Only R_3 moves toward point P_1 and both R_1 and R_2 move toward P_3, the midpoint of segment P_1P_2, and wait there for R_3. If R_3 has found the exit, they move back toward point P_1 and if not, they move toward point P_2 together. It is obvious that R_1 and R_2 moving to M_1 does not have any negative effect on the worst case evacuation time, since if the exit is close to point P_1, agent R_3 from P_1 has to travel $\frac{|P_1P_2|}{2}$ to inform the other two agents and it takes another $\frac{|P_1P_2|}{2}$ for them to get to the exit, and if the exit is located near P_2, it again takes $|P_1P_2|$ for R_3 to get to the exit from P_1.

Mark that if $r \to 0$, X3Cv1 converges to Equal Travel Early Meeting in [13] and we will show that our results for $r = 0$ is identical to results shown in [13]. Because of the Equal Travel method used in this algorithm, agents should be at the extended meeting points at the same time. Therefore we have:

- $t_1 = |OP_1| + |P_1B| + |BQ_1| + |Q_1J_2|$
- $t_2 = |OQ_1| + |Q_1A| + |AQ_2| + |Q_2J_3|$
- $t_3 = |OQ_2| + |Q_2C| + |CP_2| + |P_2J_1|$

On the other hand, at the end of the first phase when information is exchanged, there will be two critical points

- for R_2 to reach point B and
- for R_3 to finish the unexplored part of the triangle.

Putting everything together, we obtain the following equations:

1. $t_1 = t_2 = t_3$ and
2. $|J_3B| = |J_1P_1| + |P_1P_2|$

Solving these equations with Maple software, we achieve the results shown in Table 6. The results are invalid for $r \geq 0.60$ as for some segments, there is no positive solutions when solving the equations.

Note that for $r = 0.1761$ this algorithm has the lowest evacuation time of 2.076211 and if r increases, the total evacuation time increases as well.

r	$\|BQ_1\|$	$\|BP_1\|$	$\|P_1P_2\|$	$\|CQ2\|$	$E_{\mathsf{X3Cv1}(3,r)}$
0.00	0.52520	0.38602	0.26698	0.54543	2.08871
0.05	0.53158	0.36381	0.31288	0.54642	2.08252
0.10	0.53839	0.34162	0.35505	0.54463	2.07848
0.15	0.54550	0.31935	0.39290	0.54025	2.07647
0.1761	0.549272	0.307634	0.410740	0.537022	2.076211
0.20	0.55273	0.29679	0.42581	0.53354	2.07642
0.25	0.55985	0.27363	0.45322	0.52482	2.07827
0.30	0.56664	0.24914	0.47440	0.514450	2.08210
0.40	0.57799	0.18995	0.49292	0.49089	2.09689
0.50	0.58847	0.06699	0.43301	0.47455	2.13037
≥0.60	N/A	N/A	N/A	N/A	N/A

Table 6: Evacuation of 3 Agents using X3Cv1

4.2 Evacuating 3 Agents from a Triangle Using X3Cv2

We take a similar approach to X3Cv1 with some minor modifications. First, points Q_1 and Q_2 and points P_1 and P_2 are symmetric with regards to line AM. Observe that in contrast to X3Cv1, when the exploration of first three segments is finished, agents move toward opposite vertices instead of the centroid and they communicate when they are r apart at points J_1 and J_2. The trajectories of each agents are defined in Trajectory 7. The exploration of the fourth segment is identical to X3Cv1, R_1 is responsible for exploring the whole segment while R_2 and R_3 will wait for R_1 at the midpoint of this segment. Again, by X3Cv2 we refer to Algorithm 2 that uses Trajectory 7. Note that when $r \to 0$, this algorithm does not converge to *Equal Travel Early Meeting* algorithm in [13]. Hence we will have larger evacuation time for small values of r.

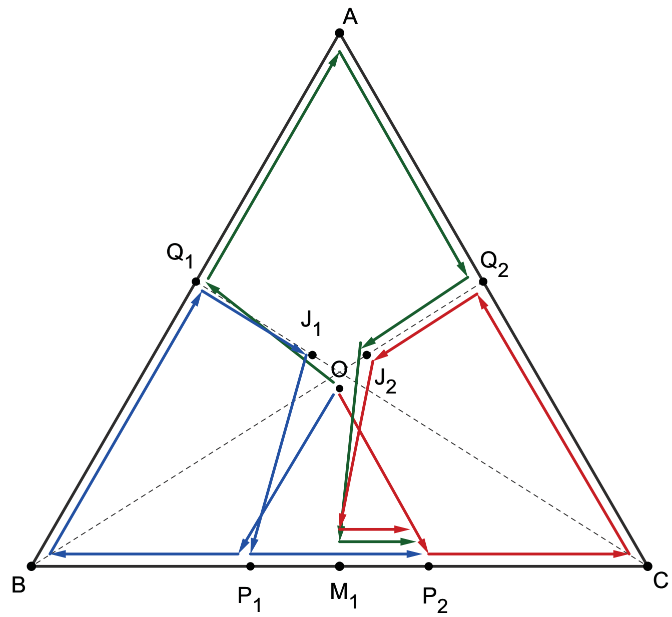


Figure 24: Evacuating three agents using X3Cv2

Trajectory 7.

R_1 follows the trajectory: $< O, P_1, B, Q_1, J_1, P_1, P_2 >$

R_2 follows the trajectory: $< O, Q_1, A, Q_2, J_2, M_1,$ wait for $R_1, P_2 >$

R_3 follows the trajectory: $< O, P_2, C, Q_2, J_2, M_1,$ wait for $R_1, P_2 >$

Again since we implement an *equal travel* method, the agents should be at the extended meeting point at the same time. Hence we will have:

- $t_1 = |OP_1| + |P_1B| + |BQ_1| + |Q_1J_1|$
- $t_2 = |OQ_1| + |Q_1A| + |AQ_2| + |Q_2J_2|$
- $t_3 = |OP_2| + |P_2C| + |CQ_2| + |Q_2J_2|$

Due to the symmetry of the algorithm, we know that $t_1 = t_3$. We only have to equate $t_1 = t_2$. Similar to previous algorithm, when agents are exchanging information, there exists two critical points as

1. point C for R_1 (same distance as B from R_2 and R_3)
2. and point P_2 for R_1

By equating these two point, we get the equation $|J_1C| = |J_1P_1| + |P_1P_2|$. Using Maplesoft to solve the equations, we acquire the results shown in Table 7. Note that same as before, the results are invalid for $r \geq 0.50$ as for some segments, there is no positive solutions when solving the equations.

r	BQ_1	BP_1	P_1P_2	Evac Time
0.00	0.53385	0.37418	0.25162	2.08962
0.05	0.53582	0.36452	0.27094	2.08599
0.10	0.53761	0.35536	0.28926	2.08270
0.15	0.53925	0.34670	0.30659	2.07972
0.20	0.54074	0.33851	0.32297	2.07700
0.25	0.54210	0.33077	0.33845	2.07454
0.30	0.54336	0.32347	0.35307	2.07231
0.40	0.54553	0.31006	0.37988	2.06837
≥0.50	N/A	N/A	N/A	N/A

Table 7: Evacuation of 3 Agents using X3Cv2

4.3 Evacuating 3 Agents from a Triangle Using X1C

As r increases, the improvements on evacuation time in the previous two algorithms are not sufficiently large to approach the evacuation time of the algorithm in [19] in which agents can communicate with each other with $r = 1$. We therefore consider an adaptation of the algorithm from [19] which results in a lower evacuation time for large value of r compared to X3Cv1 and X3Cv2. The trajectories of each agent are illustrated in Figure 25 and described in Trajectory 8. By X1C we mean the algorithm 2 with regards to Trajectory 8. In this strategy the agents first explore the side BC and later on, they will get connected to exchange information about the exit. If the exit is not found yet, they all move to vertex A. The Trajectory 8 is different from previous trajectories in a way that when R_2 reaches point Q_1, it reduces its speed so that it remains on a horizontal line that passes through the other two agents.

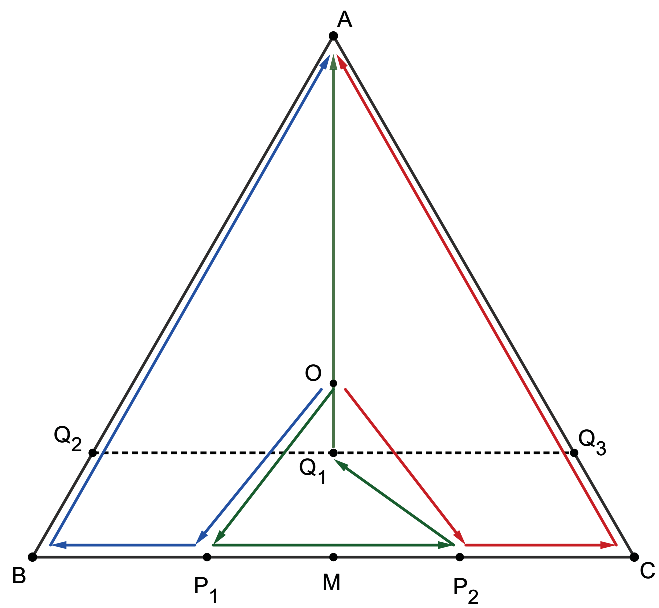


Figure 25: Evacuating three agents using X1C

Trajectory 8.

R_1 follows the trajectory: $< O, P_1, B, A >$

R_2 follows the trajectory: $< O, P_1, P_2, Q_1, A >$

R_3 follows the trajectory: $< O, P_2, C, A >$

Agents R_1, R_2 and R_3 will reach points Q_2, Q_1 and Q_3 respectively at the same time, let this time be t. Points Q_2 and Q_3 are on sides AB and AC respectively, and at the same distance from A and are chosen in a way so that $|Q_2Q_3| = \min\{2r, 1\}$. We will choose point Q_1 in a way that the agents be always in effective communication range at t and later on. As previously mentioned, when agent R_2 reaches Q_1, it will decrease its speed so it could be at the same level with R_1 and R_3 until they reach point A. Note that since the trajectories of R_1 and R_2 are symmetric, wlog we will focus on analyzing the evacuation times of R_1 and R_2.

Point Q_1

Point Q_1 is chosen based on value of r. If $0 < r < 0.5$, it will be placed on segment AM and if $r \geq 0.5$, then it will be placed on segment MC in a way that all three agents be at points B, Q_1 and C at the same time, as well as being in effective communication range with each other. Note that if $r \geq 2/3$, then points Q_1 and P_2 will be at the same location, because at time t, when R_1 is at B and R_2 is at P_2, they are already in communication range and there is no need for R_2 to move toward R_1.

In conclusion we could present 3 cases for point Q_1 based on value of r as follow:

Case 1, $0 \leq r < 0.5$**:** Point Q_1 is the midpoint of segment Q_2Q_3. Since at time t agents R_1 and R_2 should be at points Q_2 and Q_1 respectively, we have

$$|OP_1| + |P_1B| + |BQ_2| = |OP_1| + |P_1P_2| + |P_2Q_1|$$

By simplifying the equation and replacing $|P_2Q_1| = \sqrt{|Q_1M|^2 + (\frac{|P_1P_2|}{2})^2}$ and $|Q_1M| = 3y - |AQ_1|$, we get

$$\frac{1 - |P_1P_2|}{2} + (1 - 2r) = |P_1P_2| + \sqrt{(3y - \sqrt{3r^2 - 4r + 1})^2 + (\frac{|P_1P_2|}{2})^2}$$

from which $|P_1P_2|$ can be written as a function based on r.

Evacuation time for this case is $|OP_1| + |P_1B| + |BQ_2| + |Q_2C|$.

Case 2, $0.5 \leq r < 2/3$**:** Point Q_1 is positioned on segment MP_2 such that

$$|BP_1| = |P_1P_2| + |P_2Q_1| \text{ and } |BQ_1| = r$$

By solving these two equations we get $P_1P_2 = r/2$, and the evacuation time is

$$|OP_1| + |P_1B| + |BC| = \sqrt{y^2 + (r/4)^2} + (1/2 - r/4) + 1$$

Case 3, $r \geq 2/3$**:** Point Q_1 is at distance $2/3$ from B. Then we have $|P_1P_2| = 1/3$ and the evacuation time will be

$$|OP_1| + |P_1B| + |BC| = \sqrt{y^2 + (1/6)^2} + 1/3 + 1$$

The improvements compared to X3Cv1 and X3Cv2 for large values of r are achieved for the following reasons:

1. One agent acting like a relay, transfers the information between the other two, expanding the effective communication range, and

2. agents do not have to deviate from the boundary in order to exchange information, hence saving time.

The evacuation time of X1Cis illustrated in the following figure.

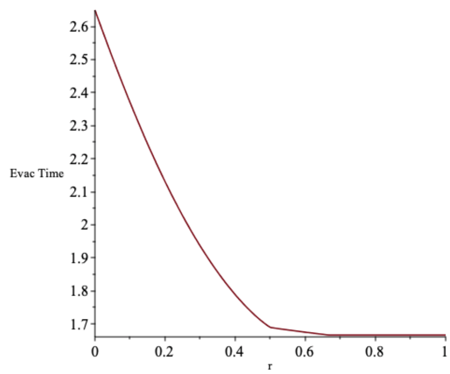


Figure 26: Evacuation time of $E_{\mathsf{X1C}}(3, r)$

4.4 Comparison

Now we are going to compare X3Cv1, X3Cv2and X1C based on their evacuation time. As can be seen in Table 8, the X3Cv1 and X3Cv2 strategies, which are adaptations of *Equal Travel Early Meeting* algorithm in [13], work better for small values of r, while the X1C strategy yields better results for large values of r. Note that the best known algorithm for face-to-face and wireless evacuates 3 agents in 2.08872 and 1.649708313 respectively.

Those specific values of r where small fluctuation in r results in one algorithm to do better than the other one is obtain empirically rather than analytically.

r	X3Cv1	X3Cv2	X1C
0	2.08872	2.08963	2.64971
0.1	2.07849	2.08271	2.37052
0.2	2.07642	2.07702	2.13056
0.20825	2.07660	2.07660	2.11274
0.22589	2.07714	2.07572	2.07572
0.25	2.07828	2.07455	2.02747
0.3	2.08210	2.07231	1.93620
0.4	2.09689	2.06838	1.78880
0.5	2.13037	N/A	1.68958
0.6	N/A	N/A	1.67532
0.7	N/A	N/A	1.666667
0.8	N/A	N/A	1.66667
0.9	N/A	N/A	1.66667
1	N/A	N/A	1.66667

Table 8: Comparison of evacuation time of 3 agents

Chapter 5

Evacuating Four or More Agents

In this chapter we take a look at four agents. The strategies described in the previous chapter can be generalized to be used in four agent variation of the problem. First, we modify the X3Cv1 to be used with four agents. Recall that in X3Cv1 strategy, at the extended meeting points the distance between any two agents is r, but with four agents, we could increase this amount to $2r$. Then we are going to describe another strategy based on X1C.

5.1 Evacuating 4 Agents from a Triangle Using X3Cv1

A similar approach to X3Cv1 is used here. The triangle is divided into 5 segments, each agent is responsible for exploring one of these segments and after the exploration of these 4 segments is finished, all the agents will move to the extended meeting points inside of the triangle, in order to exchange information. In the case that the exit is not found, the agent closest to the fifth segment, in our case R_1, starts exploring that segment and the other 3 agents will move to the midpoint of fifth segment and will wait for the other agent. By X3C, we refer to Algorithm 2 with respect to Trajectory 9. The trajectories of all 4 agents are shown in Figure 27.

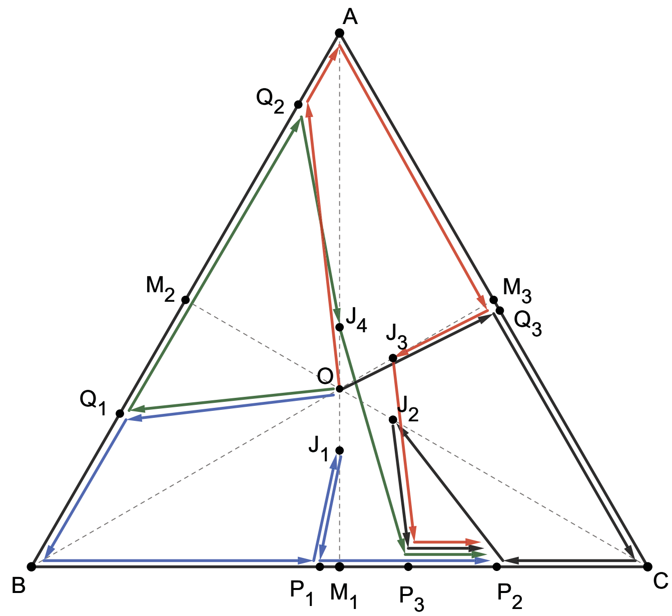


Figure 27: Evacuating four agents using X3Cv1

Trajectory 9.

R_1 follows the trajectory: $< O, Q_1, B, P_1, J_1, P_1, P_2 >$

R_2 follows the trajectory: $< O, Q_1, Q_2, J_4, P_3,$ wait for $R_1, P_2 >$

R_3 follows the trajectory: $< O, Q_2, A, Q_3, J_3, P_3,$ wait for $R_1, P_2 >$

R_3 follows the trajectory: $< O, Q_3, C, P_2, J_2, P_3,$ wait for $R_1, P_2 >$

Same as Algorithm X3Cv1, the length of the trajectories traversed by each agent up to the the point where they reach the extended meeting points, i.e. points J_{1-4}, are equal. Also points P_1 and P_2 are chosen in a way to satisfy the equation $|J_1P_1| + |P_1P_2| = |J_4C|$.

In this algorithm it is very critical for R_2 to reach point P_3 before R_1 as R_2 is the farthest agent to P_3. And this is true for small values of r, but as r gets larger than 0.11619, $|J_4P_3|$ gets larger than $|J_1P_1| + |P_1P_3|$ and hence, the evacuation time of the algorithm starts to increase at this point.

5.2 Evacuating 4 Agents from a Triangle Using X1C

A modified strategy to X1C is used here with the difference that the trajectories of agents are symmetric with regards to line AO. At first, agents travel to different points on edge BC and start exploring that edge. After the exploration of edge BC is finished, two of the agents continue exploring the other two edges while the two remaining agents will move inside the triangle in order to maintain a communication link between two agents moving alongside the edges. The trajectories of agents are illustrated in Figure 28 and defined Trajectory 10.

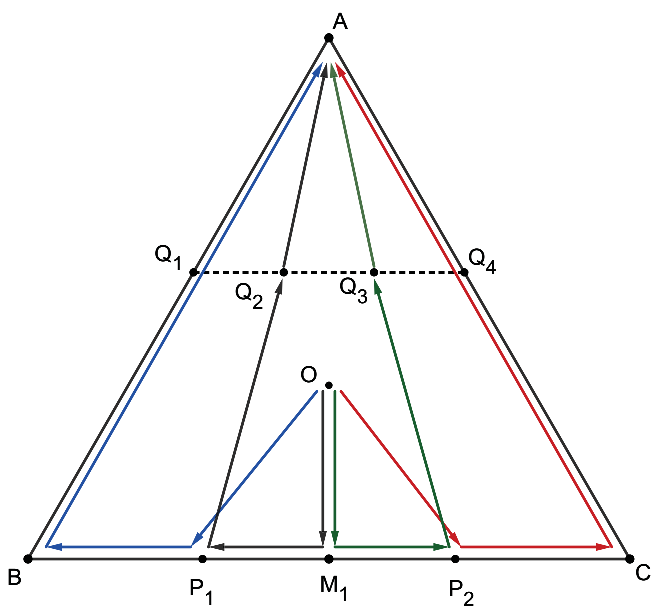


Figure 28: Evacuating four agents using X1C

Trajectory 10.

R_1 follows the trajectory: $< O, P_1, B, A >$

R_2 follows the trajectory: $< O, M_1, P_1, Q_2, A >$

R_3 follows the trajectory: $< O, M_1, P_2, Q_3, A >$

R_4 follows the trajectory: $< O, P_2, C, A >$

Note that if r is small and the exit is located at vertex C, the agents have to travel a long way to the top of T so that they could communicate and then they have to travel all the way down to the exit point. Hence this algorithm is not efficient for small values of r.

Since X1C for four agents is a symmetric algorithm, we only analyze the trajectories of R_1 and R_2 and assume the exit is located at vertex C. Based on the value of r we have the following cases for the position of points Q_1 and Q_2. Points Q_4 and Q_3 are symmetric to point Q_1 and Q_2 with respect to line AM_1.

Case1, $r < 1/3$: Point Q_1 is at distance $3r$ from A on edge AB. Point Q_2 is r-apart from point Q_1 on segment Q_1Q_4. Point P_1 is chosen in a way to satisfy the following equation:

$$y + |M_1P_1| + |P_1Q_2| = |OP_1| + |P_1B| + |BQ_1|$$

Case2, $1/3 \leq r < 0.6436493$: In this case, since $|P_1P_2| > r$, when R_2 and R_3 reach points P_1 and P_2, they have to move toward each other to be in communication range. Points Q_2 and Q_3 are placed on the edge BC, and points P_1 and Q_2 are obtained from solving the following two equations:

- $y + |M_1P_1| + |P_1Q_2| = |OP_1| + |P_1B|$ and
- $|M_1P_1| - |P_1Q_1| = r/2$

The evacuation time is $|OP_1| + |P_1B| + 1$.

Case3, $r \geq 0.6436493$: With large enough r, this problem converges to the wireless problem proposed in [18]. Points P_1 and P_2 are chosen in a way so that all the 4 agents finish exploring side BC at the same time, i.e. $y+|M_1P_1| = |OP_1|+|P_1B|$. When the edge BC is explored all the agents move toward vertex A. At all time of the execution of the algorithm, all the agents are in effective communication range. The evacuation time will be $y + |M_1P_1| + 1 \approx 1.610499805$.

Note that when R_2 and R_3 move toward vertex A, they move with speed less that 1 in a way that they always be on the line going through R_1 and R_4. The comparison of evacuation time of X1C and X3C for four agents is shown in Table 9

r	X3C	X1C
0.0	1.98157	2.59944
0.1	1.96199	2.19408
0.11619	1.95993	2.13688
0.1721	1.95993	1.95993
0.2	1.95993	1.88392
0.3	N/A	1.67649
0.4	N/A	1.62573
0.5	N/A	1.61912
0.6	N/A	1.61302
0.7	N/A	1.61050
1.0	N/A	1.61050

Table 9: Comparison for evacuating 4 agents with different strategies

5.3 Optimal Algorithm Employing Many Agents

It was previously shown that no algorithm can evacuate any number of agents from T sooner than $2y + 1$ [19]. For completeness we will give a short proof here.

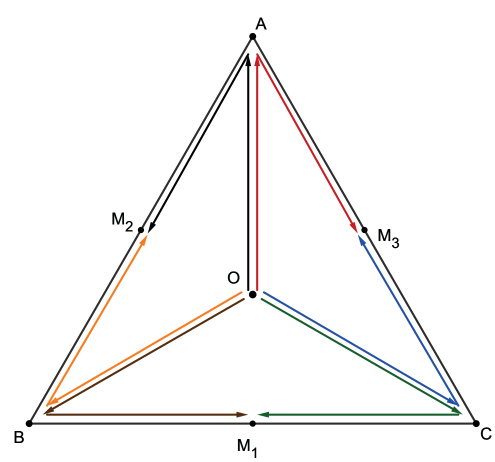


Figure 29: Trajectory of 6 agents with $r = 1$

Observation 5.1. *[19]* $\forall n\ E^*(r, n) \geq 2y + 1$.

Proof. Let t be the time that first agent visits any of the vertices A, B or C. We know that $t \geq 2y$. Then the adversary will place the exit at either of the other two vertices, forcing the agent to travel additional distance 1 to reach the exit, completing the proof. □

5.3.1 Evacuating 6 Agents with $r = 1$

It was shown in [18] that six agents can evacuate in time $1 + 2y$. The trajectories of all agents are shown in Figure 29. As soon as one agent finds the exit, all the other agents will be notified and they abandon their exploration and move toward the exit.

We now analyze the evacuation time of this algorithm.

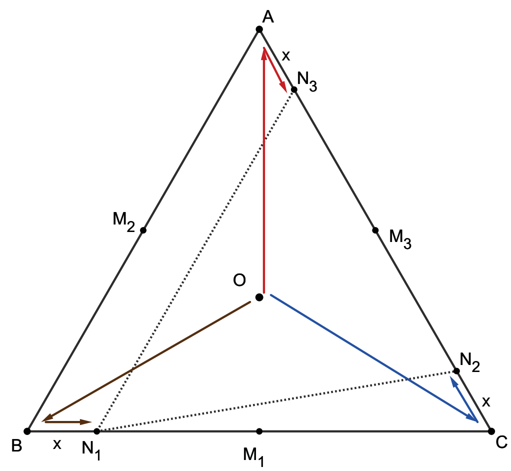


Figure 30: Lemma 5.2

Lemma 5.2. *[18] The algorithm which uses the trajectory illustrated in Figure 29 has evacuation time* $1 + 2y$ *regardless of the position of the exit.*

Proof. Wlog lets assume that the exit is located on segment BM_1 at distance x from B (see Figure 30). Let t be the time that the exit is found. It is obvious that the farthest agents to the exit are the agents that are at N_2 and N_3 at time t (see Figure 30). Now considering the triangle N_1N_3C which is an equilateral triangle with sides $1 - x$, the maximum distance between any two points on its boundary cannot be greater that $1 - x$. Hence total evacuation time will be $t + 1 - x = 2y + 1$. □

It follows from Observation 5.1 that this algorithm is optimal.

5.3.2 Evacuating 8 Agents with $r \geq 0.5$

We now show an algorithm utilizing 8 agents with $r \geq 0.5$ that is optimal. The trajectories of agents are shown is Figure 31. In this algorithm 6 agents are responsible for searching the boundary for the exit, from now on we refer to these agents as *explorers*. The other 2 agents are in charge to maintain a communication relay grid, hence we call them *relay agents*. These two agents move to M_2 and M_3 at the start of the algorithm and will stay there until the exit is found.

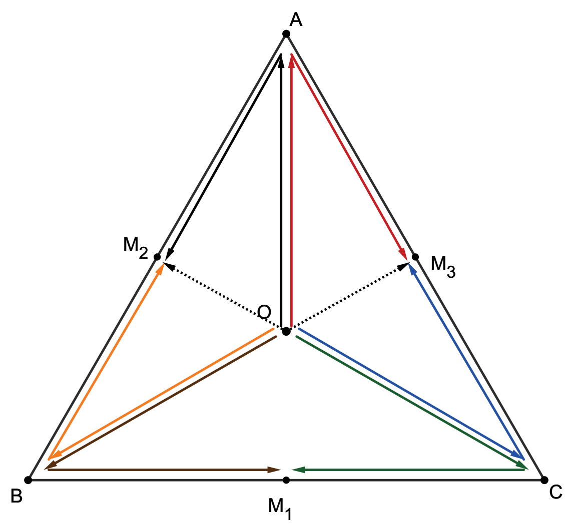


Figure 31: Evacuating 8 agents

Lemma 5.3. *Eight agents with* $r \geq 0.5$ *can evacuate* T *in* $2y + 1$.

Proof. Consider ΔBM_2M_1, since this is an equilateral triangle with side 0.5, then the distance between any two points is maximum 0.5. This is also applicable to ΔM_1M_3C. Hence all the 8 agents are in communication range at any stage of the algorithm. The evacuation time of $2y + 1$ follows directly from the proof of Lemma 5.2. □

5.3.3 Upper and Lower Bound on k

We will now show an optimal evacuation algorithm with $k = 6 + 2\lceil \frac{1}{r} - 1 \rceil$ number of agents. By T' we mean the equilateral triangle of side 0.5 with vertices at the midpoints of the 3 edges of T, namely M_1, M_2, and M_3.

We break the execution of this algorithm into multiple phases for simplicity.

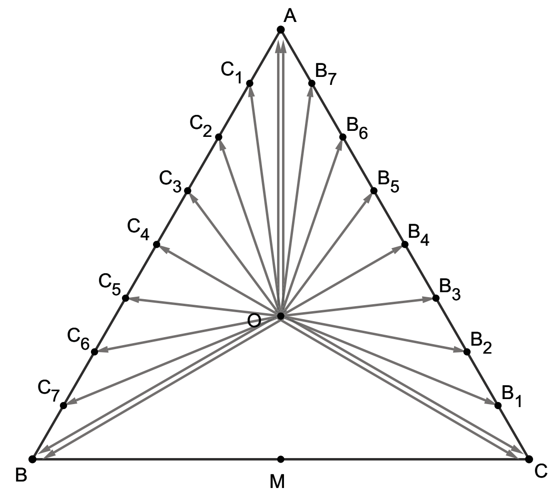


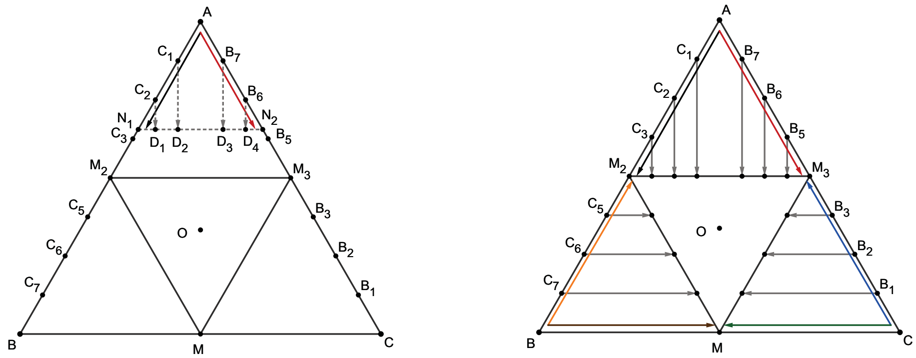
Figure 32: K agents moving to the boundary

Moving to the boundary: 6 agents will move to the 3 vertices and start exploring in the opposite directions. The trajectories of these 6 agents are identical to the trajectories of the agents shown in Figure 29. All the other agents will move to edges AB and AC in such a way that they are r-apart from their immediate left and right neighbor agents (see Figure 32), and they will stay there until all the 6 explorers reach the vertices.

Exploration: The exploration phase starts at time $2y$. Once any of the explorers reach any of the relay agents, that relay agent will start moving toward T' as shown in Figure 33b. The relay agents positioned on ΔM_2AM_3 always move on a line passing through the two explorer agents which were sent to vertex A at the previous step, see Figure 33a. Note that this implies that these relay agents move at a lower speed than the explorer agents. The other relay agents (that were positioned on segments M_2B and M_3C move on lines parallel to edge BC at unit speed.

Exit: Once one of the explorers finds the exit, all the other agents will be in effective communication range and therefore will move toward the exit immediately.

Lemma 5.4. *After the explorers reach the vertices of T, all agents will remain in effective communication range with each other until the algorithm is finished.*



(a) Positions of the agents located initially on ΔM_3AM_2 at time $2y + |AN_1|$

(b) Position of the agents at the end of the exploration phase

Figure 33: Exploration phase

Proof. It is easy to see that the distance between the agents located on the perimeter of ΔM_3AM_2 never increases. The movement of the agents located on the perimeter of ΔM_2BM_1 is symmetric to those located on ΔM_1M_3C, hence wlog we focus on the latter. Consider time t in which the blue explorer has passed the i^{th} relay agent, see Figure 34. It is easy to see that the agents from B_1 to B_i are located on a line passing through the two explorers on that part and the distance between each successive relay agent is exactly r. Also note that the distance between the green agent and B_1 is also r and the distance between the blue agent and B_1 is maximum r. Therefore, the agents located on ΔM_2BM_1 and ΔM_1M_3C are in communication range with agents located on ΔM_3AM_2 at any time. □

Theorem 5.5. *$k = 6+2\lceil \frac{1}{r} - 1\rceil$ agents located at the centroid of T with any $0 < r < 1$, can evacuate T optimally, i.e. in time $1 + 2y$.*

Proof. The explorers will reach the vertices of T at time $2y$. For any exit with distance $0 \leq x \leq 0.5$ from the nearest vertex, at least one explorer will reach it at time $2y + x$. Lemma 5.4 assures that all the agents will get notified immediately and start moving to the exit. As in Lemma 5.2, the farthest agent is at distance $1-x$ from the exit when the exit is found, and hence the evacuation time will be $2y + 1$ which is optimal. □

Now we give a lower bound on the number of agents to optimally evacuate T.

Theorem 5.6. *For any r, at least $\frac{1}{r} + 1$ number of agents are needed to evacuate T optimally, i.e. in time $1 + 2y$.*

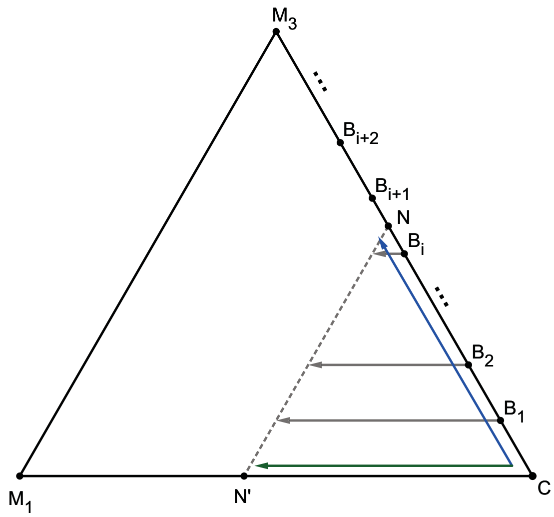


Figure 34: Agents which were initially located on segment CM_2, at time t

Proof. Let t be the time the first agent, say R_1, reaches a vertex, say A. Clearly, $t \geq 2y$. Since the adversary can place the exit at either B or C, for the evacuation time to be exactly $1+2y$, it must be that $t = 2y$, and furthermore, another agent must have reached either B or C or both, and must be able to instantly communicate the presence of the exit to R_1. For this communication to happen, an additional $1/r - 1$ agents are needed, for a total of $1/r + 1$ agents. □

Chapter 6

Conclusions and Future Work

The previously studied evacuation problems look at the problem of evacuating multiple agents from an enclosed domain which has one or multiple exits. The model used, *face-to-face* and *wireless* are two extreme cases of the communication spectrum, whereas our new model, revisits the problem with a more realistic approach; the agents should be close enough in order to communicate with each other.

In this thesis we focused on an *equilateral triangle* as a search domain. We first modified the previously known *Equal-Travel with Detour* algorithm, but the detours were only beneficial for some values of r. Then we proposed a new trajectory for the detours which is applicable to all values of r. These new trajectories give an evacuation time of 2.2862 for the evacuation of two agents in the face-to-face model, which is an improvement over the previously known result of 2.3367. We also give a lower bound $1.6547 - r$ on the evacuation time for any value of $r \leq 0.336$

For three agents, for any r, we provided an algorithm with three different trajectories; X3Cv1 and X3Cv2 where agents explore all three sides of T before connecting and X1C where the agents first fully explore one side and then connect. We showed that X1C is a better approach for large values of r and for smaller values, the other two approaches yields better results. For four agents, we propose two different trajectories, one to X3Cv1 and the other one similar to X1C in the case of three agents. Finally, we showed that for any r, evacuation of $k = 6 + 2\lceil(\frac{1}{r} - 1)\rceil$ agents can be done using the CXP strategy in time $1 + \sqrt{3}/3$, which is optimal in terms of time, and asymptotically optimal in terms of the number of agents.

Throughout this thesis, we were unable to give a closed form solution for the evacuation time as a function of r, due to the complexity of the equations. It is desired to have such a function or at least an estimation. We are aware that our bounds are not tight. For the two agents, we were only able to give a lower bound for $r \leq 0.336$, it would be interesting to have a better bound for $r > 0.336$ rather than the general lower bound for k agents introduced in [19].

All the proposed algorithms in this thesis are categorized under *offline algorithms*, since before the execution, the agents are aware of their surroundings. It would be an interesting idea to implement an *online algorithm* for agents which have no or limited information regarding the search domain. This could be a consequence of a fire blocking a part of the structure, destruction of some of parts of the building due to an earthquake or etc. We would also like to know what is the best behavior of the agents in other geometric shapes.

Finally, we would like to know what trajectories should be used if we need to save only a specific agent from danger as in [22].

Bibliography

[1] M. Aigner and M. Fromme. A game of cops and robbers. *Discrete Applied Mathematics*, 8(1):1 – 12, 1984.

[2] Steve Alpern. Search for point in interval, with high–low feedback. *Mathematical Proceedings of the Cambridge Philosophical Society*, 98(3):569–578, 1985.

[3] Steve Alpern and Shmuel Gal. *The theory of search games and rendezvous*. Kluwer Academic Publishers, 2003.

[4] Iman Bagheri, Lata Narayanan, and Jaroslav Opatrny. Evacuation of equilateral triangles by mobile agents of limited communication range. In Falko Dressler and Christian Scheideler, editors, *Algorithms for Sensor Systems*, pages 3–22. Springer International Publishing, 2019.

[5] V. J. Baston and F. A. Bostock. A high–low search game on the unit interval. *Mathematical Proceedings of the Cambridge Philosophical Society*, 97(2):345–348, 1985.

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

[7] Anatole. Beck. An optimal search (richard bellman). *SIAM Review*, 27(3):447–448, 1985.

[8] Anatole Beck and D. J. Newman. Yet more on the linear search problem. *Israel Journal of Mathematics*, 8(4):419–429, Dec 1970.

[9] Béla Bollobás, Gábor Kun, and Imre Leader. Cops and robbers in a random graph. *Journal of Combinatorial Theory, Series B*, 103(2):226 – 236, 2013.

[10] Sebastian Brandt, Felix Laufenberg, Yuezhou Lv, David Stolz, and Roger Wattenhofer. Collaboration without communication: Evacuating two robots from a disk. In Dimitris Fotakis, Aris Pagourtzis, and Vangelis Th. Paschos, editors, *Algorithms and Complexity*, pages 104–115. Springer International Publishing, 2017.

[11] Edmund Cheong Kong Chuan. A helicopter submarine search game. Master's thesis, Naval Postgraduate School, 1988.

[12] Huda Chuangpishit, Jurek Czyzowicz, Evangelos Kranakis, and Danny Krizanc. Rendezvous on a line by location-aware robots despite the presence of byzantine faults. In Antonio Fernández Anta, Tomasz Jurdzinski, Miguel A. Mosteiro, and Yanyong Zhang, editors, *Algorithms for Sensor Systems*, pages 70–83. Springer International Publishing, 2017.

[13] Huda Chuangpishit, Saeed Mehrabi, Lata Narayanan, and Jaroslav Opatrny. Evacuating an Equilateral Triangle in the Face-to-Face Model. In James Aspnes, Alysson Bessani, Pascal Felber, and João Leitão, editors, *21st International Conference on Principles of Distributed Systems (OPODIS 2017)*, volume 95 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:16, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[14] Huda Chuangpishit, Saeed Mehrabi, Lata Narayanan, and Jaroslav Opatrny. Evacuating equilateral triangles and squares in the face-to-face model. *CoRR*, abs/1812.10162, 2018.

[15] J. Czyzowicz, S. Dobrev, K. Georgiou, E. Kranakis, and F. MacQuarrie. Evacuating two robots from multiple unknown exits in a circle. *Theor. Comput. Sci.*, 709:20–30, 2018.

[16] J. Czyzowicz, L. Gasieniec, T. Gorry, E. Kranakis, R. Martin, and D. Pajak. Evacuating robots from an unknown exit located on the perimeter of a disk. In *Proceedings DISC, Austin, Texas*, pages 122–136. Springer, 2014.

[17] J. Czyzowicz, K. Georgiou, E. Kranakis, L. Narayanan, J. Opatrny, and B. Vogtenhuber. Evacuating robots from a disk using face-to-face communication (extended abstract). In Vangelis Th. Paschos and Peter Widmayer, editors,

Algorithms and Complexity, pages 140–152. Springer International Publishing, 2015.

[18] J. Czyzowicz, E. Kranakis, D. Krizanc, L. Narayanan, J. Opatrny, and S. Shende. Wireless autonomous robot evacuation from equilateral triangles and squares, extended version, in preparation. In Symeon Papavassiliou and Stefan Ruehrup, editors, *Ad-hoc, Mobile, and Wireless Networks*, pages 181–194. Springer International Publishing, 2015.

[19] J. Czyzowicz, E. Kranakis, D. Krizanc, L. Narayanan, J. Opatrny, and S. M. Shende. Wireless autonomous robot evacuation from equilateral triangles and squares. In *Proceedings of the 14th International Conference on Ad-hoc, Mobile, and Wireless Networks (ADHOC-NOW 2015), Athens, Greece*, pages 181–194, 2015.

[20] Jurek Czyzowicz, Stefan Dobrev, Konstantinos Georgiou, Evangelos Kranakis, and Fraser MacQuarrie. Evacuating two robots from multiple unknown exits in a circle. *Theoretical Computer Science*, 709:20 – 30, 2018. Special Issue of ICDCN 2016 (Distributed Computing Track).

[21] 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 Shantanu Das and Sebastien Tixeuil, editors, *Structural Information and Communication Complexity*, pages 158–173. Springer International Publishing, 2017.

[22] Jurek Czyzowicz, Konstantinos Georgiou, Ryan Killick, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, and Sunil Shende. God Save the Queen. In Hiro Ito, Stefano Leonardi, Linda Pagli, and Giuseppe Prencipe, editors, *9th International Conference on Fun with Algorithms (FUN 2018)*, volume 100 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 16:1–16:20, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[23] Jurek Czyzowicz, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, and Jaroslav Opatrny. Search on a line with faulty robots. In *Proceedings of the 2016*

ACM Symposium on Principles of Distributed Computing, PODC '16, pages 405–414, New York, NY, USA, 2016. ACM.

[24] Jurek Czyzowicz, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, and Jaroslav Opatrny. Search on a line with faulty robots. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, PODC '16, pages 405–414, New York, NY, USA, 2016. ACM.

[25] Jurek Czyzowicz, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, and Sunil Shende. Linear search with terrain-dependent speeds. In Dimitris Fotakis, Aris Pagourtzis, and Vangelis Th. Paschos, editors, *Algorithms and Complexity*, pages 430–441. Springer International Publishing, 2017.

[26] Stefan Dobrev, Rastislav Královic, and Dana Pardubská. Treasure Hunt with Barely Communicating Agents. In James Aspnes, Alysson Bessani, Pascal Felber, and João Leitão, editors, *21st International Conference on Principles of Distributed Systems (OPODIS 2017)*, volume 95 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 14:1–14:16, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[27] Sándor Fekete, Chris Gray, and Alexander Kröller. Evacuation of rectilinear polygons. In Weili Wu and Ovidiu Daescu, editors, *Combinatorial Optimization and Applications*, pages 21–30, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[28] Peter Frankl. Cops and robbers in graphs with large girth and cayley graphs. *Discrete Applied Mathematics*, 17(3):301 – 305, 1987.

[29] Shmuel. Gal. Minimax solutions for linear search problems. *SIAM Journal on Applied Mathematics*, 27(1):17–30, 1974.

[30] Shmuel Gal. *Search games.* Academic Press, 1980.

[31] Rufus Isaacs. *Differential Games.* John Wiley and Sons, 1965.

[32] Artur Jeż and Jakub Łopuszański. On the two-dimensional cow search problem. *Information Processing Letters*, 109(11):543 – 547, 2009.

[33] Ming-Yang Kao, John H. Reif, and Stephen R. Tate. Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem. *Information and Computation*, 131(1):63 – 79, 1996.

[34] Andrey Kupavskii and Emo Welzl. Lower bounds for searching robots, some faulty. *Distributed Computing*, Aug 2019.

[35] S. P. Lalley. A one-dimensional infiltration game. *Naval Research Logistics (NRL)*, 35(5):441–446, 1988.

[36] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.

[37] S. Neufeld and R. Nowakowski. A game of cops and robbers played on products of graphs. *Discrete Mathematics*, 186(1):253 – 268, 1998.

[38] Richard Nowakowski and Peter Winkler. Vertex-to-vertex pursuit in a graph. *Discrete Mathematics*, 43(2):235 – 239, 1983.

[39] Debasish Pattanayak, H. Ramesh, Partha Sarathi Mandal, and Stefan Schmid. Evacuating two robots from two unknown exits on the perimeter of a disk with wireless communication. In *Proceedings of the 19th International Conference on Distributed Computing and Networking*, ICDCN '18, pages 20:1–20:4, New York, NY, USA, 2018. ACM.

[40] K. Spieser and E. Frazzoli. The cow-path game: A competitive vehicle routing problem. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 6513–6520, Dec 2012.