Collaboration Without Communication: Evacuating Two Robots from a Disk

Sebastian Brandt[1](✉), Felix Laufenberg[1], Yuezhou Lv[2], David Stolz[1], and Roger Wattenhofer[1]

[1] ETH Zürich, Zürich, Switzerland
{brandts,stolzda,wattenhofer}@ethz.ch, felix.laufenberg@gmx.de
[2] Tsinghua University, Beijing, China
lvyuezhou@gmail.com

Abstract. We consider the problem of evacuating two robots from a bounded area, through an unknown exit located on the boundary. Initially, the robots are in the center of the area and throughout the evacuation process they can only communicate with each other when they are at the same point at the same time. Having a visibility range of 0, the robots can only identify the location of the exit if they are already at the exit position. The task is to minimize the time it takes until both robots reach the exit, for a worst-case placement of the exit. For unit disks, an upper bound of 5.628 for the evacuation time is presented in [8]. Using the insight that, perhaps surprisingly, a forced meeting of the two robots as performed in the respective algorithm does not provide an exchange of any non-trivial information, we design a simpler algorithm that achieves an upper bound of 5.625. Our numerical simulations suggest that this bound is optimal for the considered natural class of algorithms. For dealing with the technical difficulties in analyzing the algorithm, we formulate a powerful new criterion that, for a given algorithm, reduces the number of possible worst-case exits radically. This criterion is of independent interest and can be applied to any area shape. Due to space restrictions, this version of the paper contains no proofs or illustrating figures; the full version can be found at http://disco.ethz.ch/publications/ciac2017-robotevac.pdf.

1 Introduction

Imagine that two robots are trapped in the middle of a room with a single door. Their goal is to evacuate both of them via the door in the shortest possible time. However, there is a problem: The position of the door is unknown to them in the beginning. Moreover, the robots have no sight and no wireless communication; they cannot see the door or the other robot except when they are right on top of it, and they can only communicate when they are at the same point at the same time. However, they do know the shape of the room and share all information before they start searching for the door. How should they divide the work of scouring the boundary for the door? Which routes should they take? Does it make sense to meet at some predefined point in order to exchange information?

D. Fotakis et al. (Eds.): CIAC 2017, LNCS 10236, pp. 104–115, 2017.
DOI: 10.1007/978-3-319-57586-5_10

We consider the above problem for the most fundamental room shape possible, namely, the unit disk. More formally, the goal is to minimize the time for evacuating both robots from the room where the exit is assumed to be worst-case placed and the robots move with unit speed. The state-of-the-art algorithm for this problem due to Czyzowicz et al. [8] proceeds roughly as follows: First, both robots move to the same point on the perimeter, then they search the perimeter in different directions until they meet at the opposite point. At some predefined point in time during their search along the perimeter they leave the perimeter on symmetric non-linear routes in order to meet inside the circle upon which they return to their search. If a robot finds the exit at any point in time, then it immediately calculates the shortest route for meeting the other robot and subsequently brings the other robot to the exit. The authors of [8] show that this algorithm achieves an evacuation time of 5.628 and remark that it is possible to improve upon this result by truncating the detour to the middle slightly.

We note that the forced meeting of the robots cannot be used to exchange any non-trivial information. Moreover, the requirement to meet introduces dependencies between the parameters of the detour such as position, length, shape and angle. We prove that removing this requirement indeed allows for an improved algorithm that utilizes the independence of the aforementioned parameters while preserving the simplicity of the algorithm. In fact, we present an algorithm that simplifies the algorithm described above by omitting the forced meeting and instead using one (symmetric) detour (per robot) that is a straight line with fixed depth. An important point in omitting the forced meeting is that there is actually an implicit exchange of information between the robots even before any meeting: When a robot finds the exit, the best it can do is to meet the other robot as quickly as possible in order to communicate the location of the exit. Conversely, from not being visited by the other robot up to some point a robot can deduce that the exit does not lie in a certain part of the perimeter.

Furthermore, we show that, surprisingly, the shape of the detour and its angle to the perimeter do not affect the evacuation time if they are chosen from some reasonable range. In particular, our linear detour is optimal for the parameters chosen for the depth and the position of the detour.

Our algorithm achieves an evacuation time of 5.625, thereby slightly improving upon the previously best known upper bound. For the class of algorithms as described above with exactly one symmetric detour per robot, our numerical simulations suggest that this bound is optimal (up to numerical precision, of course). A theoretical substantiation for this optimality claim is given by the fact that for our algorithm there are three different worst-case exit placements with the same evacuation time (again, up to numerical precision). These three exit positions are characteristic for the algorithms from the mentioned class—it stands to reason that, in an optimal algorithm, they have to exhibit the same evacuation time (since otherwise the parameters of the algorithm could be changed locally in a way that improves the evacuation time for the worst of the three points) and that there is only one algorithm that has the same worst-case evacuation time for these three exit positions.

A fundamental problem regarding evacuation from a disk is that the evacuation time for a fixed algorithm and a fixed exit placement is usually the solution to some equation of the kind $x = \cos(x)$ (only more complex). For equations of this kind no closed-form solutions are known in general, which makes it difficult to find the worst-case exit placement for a fixed algorithm, not to mention to find an algorithm with an improved worst-case evacuation time. We take a substantial step in remedying this problem by proving that a very specific condition must be satisfied for an exit in order to be worst-case placed. In "reasonable" algorithms this condition is satisfied at only a few exit positions which makes it a powerful tool for determining that an exit is not worst-case placed. In fact, in order for an exit to be worst-case placed, it must satisfy one of the two following conditions: (1) The movement of one of the two robots at the exit point, resp. pick-up point[1], is not differentiable, or (2) the angles β and γ between the line connecting exit and pick-up point and the directions of movement of the robots at the exit, resp. the pick-up point, satisfy $2\cos\beta + \cos\gamma = 1$. Moreover, the presented tool is not restricted to the disk—it can be applied to any room shape. For the analysis of our aforementioned algorithm, we rely heavily on this tool. In fact, one might consider the development of this tool as the foremost contribution of this work, while its application to the disk scenario may serve as an example of its practicability.

1.1 Related Work

A thriving area in the context of problems involving mobile agents are search problems in all its diversity. Such problems include ants searching for food (cf. [11–13]), rendezvous problems (cf. [1,10,17]), pursuit-evasion games (cf. [14,20,21]) and graph exploration problems (cf. [15,16,19]). Another example are evacuation problems, where one or multiple robot(s) search for one or multiple exit(s) through which usually all of the robots have to evacuate. Evacuation problems have been studied in a centralized setting in which the robots know the search terrain and where the other robots are, and in a distributed setting where the knowledge of the robots is restricted to the area they have already explored. Very recent results concerning optimal strategies on graphs in both settings can be found in [4]. In the following, we assume that the area is known to the robots, the exit is worst-case placed and the robots move with unit speed.

Evacuation problems can be grouped into two main categories, namely, evacuation problems on graphs and geometric evacuation problems. Since our paper deals with a problem from the latter class, we will focus on the related work in this domain. Another distinction is given by the model of communication between the robots: Here, we distinguish between instantaneous wireless communication and non-wireless communication where explicit communication can only take place when the communicating entities are at the same point.

[1] Recall that upon finding the exit, a robot immediately takes the shortest possible tour to meet the other robot and communicate the location of the exit. We call the point where this meeting happens the *pick-up point*.

In the geometric setting, research has considered different areas from which the robots have to escape. The famous cow-path problem asks how long it takes a single robot (or cow) to evacuate through a worst-case placed exit on a line, in terms of the distance d between initial position and exit. The correct answer of $9d$ (up to lower order terms) was given by Beck and Newman [3] already in 1970, and later rediscovered by Baeza-Yates et al. [2]. In [5], Chrobak et al. show that, somewhat surprisingly, the same is true for the evacuation time of multiple robots on the line in the non-wireless communication model.

For the case of two robots in equilateral triangles and squares, Czyzowicz et al. [9] present optimal evacuation trajectories, given wireless communication.

Study of the unit disk as the confining environment was initiated by Czyzowicz et al. in [7]. The authors present upper bounds of $3 + 2\pi/k$ and $3 + \pi/k + O(k^{-4/3})$ and lower bounds of $3 + 2\pi/k - O(k^{-2})$ and $3 + \pi/k$ for the non-wireless and the wireless communication model, respectively, where k is the number of robots. Moreover, they give better upper and lower bounds for the case of 2 and 3 robots, amongst them a lower bound of approximately 5.199 and an upper bound of approximately 5.74 for the case of two robots in the non-wireless model. In [8], Czyzowicz et al. improve the latter two bounds to a lower bound of approximately 5.255 and an upper bound of 5.628. Lamprou et al. [18] present (partly matching) upper and lower bounds for two robots in the wireless model where one robot, deviating from the above, has speed of larger than 1. Finally, in [6], Czyzowicz et al. consider variations of the problem of evacuating from a disk where the two robots do not know their own initial locations.

For our paper, the algorithms from [7] and [8] that achieve the upper bounds for two robots in the non-wireless communication model are of particular interest. The algorithm from [7] proceeds as follows: Starting in the center M of the disk, both robots move to the same point A on the perimeter and start searching for the exit in opposite directions. When one of the robots finds the exit, it picks the other robot up as fast as possible and returns with it to the exit. This results in an upper bound of approximately 5.74. The authors of [8] improve on this algorithm by incorporating a *forced meeting* of the two robots before all of the perimeter is searched. For this, the robots leave the perimeter in a straight line, symmetric to each other, until they meet, and then return to their search of the perimeter if the exit has not been found yet. The authors were able to improve on this algorithm even more by moving towards the meeting point in a triangular fashion. The reasons for this further improvement are explained in Sect. 3.1.

1.2 Model

The specifics of the model for our robot evacuation problem, developed in [7], are as follows: The area from which the robots have to escape is a disk of radius 1. Somewhere on its perimeter, there is a point, called *exit*, which two robots, initially placed in the center of the disk, have to find and evacuate through. The task of the robots is to minimize the time until both robots have reached the exit, which we call the *evacuation time*. We assume that the location of the exit on the perimeter is worst-case for the algorithm the two robots perform, i.e., the exit position maximizes the evacuation time. The robots itself are point-shaped and move at unit speed. Changing direction takes no time and communication is also instantaneous, but only possible if both robots are at the same point.[2] Since this is the case in the beginning, the robots can exchange all information about each others algorithm before they start searching for the exit. The robots have no vision, and therefore can only identify the exit position when they are at the exact location of the exit. Computation also takes no time and we assume that the robots are able to actually perform all necessary computations. The robots know the shape of the area and they have the same sense of direction, i.e., we may assume that they have the same underlying coordinate system.

1.3 Notation

In the following, we give an overview of the notation and the most important terms we use.

R_1 and R_2 References for the two robots. We will call the robot that finds the exit first R_1, and the other robot R_2.

$\widehat{AB}$ For two points A and B on the perimeter, $\widehat{AB}$ denotes the shorter arc from A to B along the perimeter.

AB Denotes the straight line between A and B.

$|\widehat{AB}|$ **or** $|AB|$ Denote the lengths of $\widehat{AB}$, resp. AB.

Cut. A movement of a robot from the perimeter onto the disk and back to the point where the perimeter was left. Note that a cut can take any shape in general. However, in our algorithm, the term *cut* describes a linear cut, i.e., a movement from the perimeter onto the disk and back on a straight line.

Cut length. The distance traveled when moving along a cut.

Cut depth. Only used if the cut is linear, in which case the cut depth is defined as half of the cut length.

Cut position. Point where a robot leaves the perimeter to perform a cut.

Meeting protocol. A term coined in [8]. When R_1 finds the exit, the best it can do to minimize the evacuation time is to compute (and take) the shortest route to meet R_2. Since R_1 knows the algorithm R_2 follows (and therefore also that R_2 has not found the exit so far), R_1 can actually determine this shortest route.[3] Note that this route is always a straight line since otherwise there would be a shorter route, by the triangle inequality. After meeting each other, both robots travel straight to the exit. This process of picking the other

[2] Note that a robot can also infer information from the fact that the other robot is not at the same point as it is. For instance, it may conclude that the other robot has not already found the exit in some specific segment of the perimeter, since otherwise the other robot would have picked him up at the latest at the current position. This indirect information transfer plays an important role in our arguments that the robots cannot infer any non-trivial information from a forced meeting.

[3] We emphasize that R_1 does not calculate a shortest route to the point where R_2 is when R_1 finds the exit, but rather the shortest route for picking R_2 up, knowing that and how R_2 will move until being picked up.

robot up after finding the exit and traveling to the exit together is called the *meeting protocol.* If R_1 finds the exit at time t and the aforementioned shortest route has length x, then the evacuation time is $t + 2x$.

Pick-up point. The point where R_1 picks R_2 up, following the meeting protocol.

2 Determining the Worst-Case Placement of the Exit

If an algorithm for the two robots is fixed, it is still a challenging task to determine the worst-case exit placement and thereby the evacuation time. One reason is that already determining the pick-up point for a fixed exit placement often involves solving equations where polynomial and trigonometric functions in some variable x occur side by side. For equations of this kind no closed-form solutions are known in general. In this section, we develop a new technique to determine possible candidates for the worst-case placement of the exit. More precisely, we give a criterion that determines for a pair (exit, pick-up point) whether the exit can be excluded from the list of candidates of worst-case placed exits, by only looking at the behaviour of the algorithm in ε-neighborhoods of the exit B and the pick-up point C. The criterion is quite strong in the following sense: Let β denote the angle between the straight line from exit to pick-up point and the direction of the movement of R_1 at the exit. Let γ denote the angle between the straight line from exit to pick-up point and the direction of the movement of R_2 at the pick-up point. Then, a very specific relation between β and γ has to be satisfied in order that the exit is not excluded from the list of possible worst-case placed exits. The key result is the following theorem:

Theorem 1. *If the trajectories of the two robots are differentiable around B and C and $2\cos\beta + \cos\gamma \neq 1$, then there is an exit position that yields a larger evacuation time than placing the exit at B.*

The proof of the theorem is long and involved and can be found in the full version of the paper. According to the theorem, in order to be able to exclude an exit, the movement of the two robots at the exit and the corresponding pick-up point have to be differentiable. However, in reasonable algorithms, this property holds for almost all possible exit points. Out of these exit points, the only ones that are not excluded are those that satisfy $2\cos\beta + \cos\gamma = 1$, yielding a large reduction in the number of possible exit points. We note that our considerations are independent of the shape of the area, i.e., they hold for arbitrarily shaped areas, allowing us, for instance, to finally tackle fundamental shapes like circles.

Another interesting information can be obtained from the proof of Theorem 1: On which side of 1 the term $2\cos\beta + \cos\gamma$ lies, determines to which direction one has to move the exit in order to obtain an exit with a larger evacuation time. If $2\cos\beta + \cos\gamma < 1$, then shifting the exit at B in the direction of the movement of R_1 (if it did not find the exit at B) will provide an exit position with larger evacuation time (than the exit position at B). If $2\cos\beta + \cos\gamma > 1$, then shifting the exit at B in the reverse direction will provide an exit position with larger evacuation time. We note that it does not matter if the two robots move to the same side of the infinite line through B and C or to the same side.

3 Evacuating from a Disk

In this section we use the criterion $2\cos\beta + \cos\gamma \neq 1$, which we developed in Sect. 2, in order to improve the upper bound for evacuating two robots from a unit disk to 5.625. Like the algorithm presented in [8], which achieves the previously best known upper bound of 5.628, our algorithm consists of each robot exploring its assigned half of the perimeter, only interrupted by exactly one detour each, called cut, to the inside of the circle (and symmetric to the other's cut). In contrast, the algorithm from [8] additionally contains a forced meeting of the two robots at the far end of the cuts.

We will show that, perhaps counterintuitively, the robots cannot infer any non-trivial information from this meeting that they could not have inferred from the previous course of events. Thus, such a meeting can be omitted. We will see in more detail that all the advantages of the meeting come from the actual movement of cutting to the middle and not from an explicit exchange of information.

Without the condition that the two robots actually have to meet at the endpoint of their (symmetric) cuts, many cuts are possible candidates for improving the runtime of the evacuation algorithm. A cut is determined by four properties: the position on the perimeter where the cut starts and ends, the shape of the cut, the angle at which the cut protrudes from the perimeter and the size of the cut, which corresponds to the cut depth in the case of a linear cut. As we will show, somewhat surprisingly, the shape of the cut is optimal if it is linear, for the choices of the other three parameters made in our algorithm. Similarly, we will show that the angle does not influence the performance of the algorithm if it is chosen in a reasonable range. We will provide a choice of the remaining two parameters for linear cuts that achieves the stated bound of 5.625. Moreover, we give a rigorous proof for the evacuation time.

3.1 The Algorithm $A(y, \alpha, d)$

In this section, we describe a parameterized evacuation algorithm and provide a partitioning of R_1's half of the perimeter into segments that will be useful in the evacuation time analysis. We show that the forced meeting in the previously best algorithm from [8] does not help in exchanging non-trivial information between the robots. In Sect. 3.2, we prove that the parameters can actually be chosen in a way that improves the previously best known upper bound.

Our parameterized algorithm $A(y, \alpha, d)$ is similar to the algorithm proposed in [8]: From the center of the disk, both robots move to the same point A of the perimeter and continue on the perimeter in opposite directions. At some point C, resp. B, where $\widehat{AC} = \widehat{AB} = y$, the robots leave the perimeter at angle α in a straight line, until they reach depth d and then return straight to point C, resp. B. Then both robots continue to search along the perimeter until they meet at point D. If a robot finds the exit at any point in time, it immediately performs the meeting protocol to pick the other robot up and evacuate through the exit. Note that, for convenience, α denotes the angle between the cuts and BC.

Now we examine Algorithm $A(y, \alpha, d)$ in more detail, laying the foundation for the analysis of the evacuation time for a specific choice of the parameters y, α and d in Sect. 3.2. Since the described algorithm is symmetric, it is sufficient to analyze possible exit positions on one side of the symmetry axis, i.e., for one of the two robots. Without loss of generality, we assume that the exit lies on the arc from A to D that contains C, which implies that the robot that explores this arc is called R_1 and the other one R_2. We partition this arc into four segments by specifying the points on the arc where one segment ends and the next one begins. Note that, for simplicity, we include any of these dividing points in both its adjacent segments if not explicitly specified otherwise. The choice of the segments depends on the parameters of our algorithm.

1. **Segment $\widehat{AI_1}$:** Here, I_1 is the point with the following property: If the exit is at I_1, then R_1 will pick R_2 up at point B before R_2 performs its cut, i.e., I_1 satisfies $|\widehat{AI_1}| + |I_1B| = |AB|$. This segment contains exactly those exit positions for which the evacuation time is not influenced by the cut.
2. **Segment $\widehat{I_1I_2}$:** Here, I_2 is the point with the following property: If the exit is at I_2, then R_1 will pick R_2 up at point B after R_2 performs its cut, i.e., I_2 satisfies $|\widehat{AI_2}| + |I_2B| = |AB| + 2d$. This segment contains exactly those exit positions for which the pick-up point lies on the cut.
3. **Segment $\widehat{I_2I_3}$:** Here, $I_3 = C$. The exit positions in this segment are those for which R_1 finds the exit before performing its cut, but R_2 is picked up after performing its cut.
4. **Segment $\widehat{I_3D}$:** For this segment, we explicitly specify that I_3 itself does not belong to the segment. This segment contains exactly those exit positions that R_1 reaches after performing its cut.

One main difference of our general algorithm to the one suggested in [8] is that in the latter the robots always cut far enough to meet each other. In the following, we argue that this meeting is not necessary since no real information can be shared. Consider the following three cases for the algorithm from [8]:

Case 1**:** The exit is located in one of the first three segments, excluding I_3. If R_1 went now immediately to the meeting point on a straight line, then it would arrive there earlier than if it had not found the exit before performing its cut. Because of symmetry reasons, it would also arrive earlier than R_2 at the meeting point. Hence, by using the meeting protocol upon finding the exit, R_1 picks R_2 up before R_2 reaches the meeting point. Thus, R_2 never reaches the meeting point and therefore no information can be shared.

Case 2**:** The exit is located at I_3. In this case, there actually is an exchange of information at the meeting point, but the reason is that the predefined meeting point happens to be the pick-up point for the exit position at I_3. In other words, if R_1 (but not R_2) followed a completely different algorithm without a forced meeting but with the property that it finds the exit at I_3 at the same time as in Algorithm $A(y, \alpha, d)$, then it would still pick R_2 up at the same point at the same time and the resulting evacuation time would not change. Thus, even in this specific case, the benefit in the algorithm from [8] does not come from the forced meeting, but from the fact that R_2 cuts far enough in the direction of the exit to be picked up at the tip of its cut.

Case 3: The exit lies behind R_1's cut, in the fourth segment. At the meeting point, the only relevant information that can be shared is that neither robot has found the exit yet. However, both robots can deduce this information from the fact that they have not been picked up yet (see Case 1).

Note that in [8], two algorithms were presented, as described in Sect. 1.1: One with a linear cut and an improved one where the robots cut to the meeting point in a triangular fashion. In the latter, the exit position at I_3 is also dealt with by the explanations in the above Case 1, while Case 2 is not needed at all.

We can conclude that the meeting itself does not contribute to a better runtime of the algorithm. But it *does* limit the algorithm by introducing a dependency between cut position and cut length. At first sight it might seem as if the improvement between the two algorithms presented in [8] simply comes from the shape of the cut and therefore a shortening of the pick-up distance. However, the possibility to find parameters for the algorithm with the triangular cut that give an improved evacuation time essentially comes from a decoupling of cut position and cut length. Yet there is still some correlation between cut position and cut length which is completely nullified in our algorithm $A(y, \alpha, d)$.

3.2 The Evacuation Time for $y = 2.62843$, $\alpha = \pi/4$ and $d = 0.48793$

In this section, we show an evacuation time of 5.625 for Algorithm $A(y, \alpha, d)$ for the parameters $y = 2.62843$, $\alpha = \pi/4$ and $d = 0.48793$.[4] To do so we determine, for each of the four segments defined in Sect. 3.1, the potential candidates for the worst-case exit position and then take the maximum over the evacuation times for those exits positions. For determining these candidates we use our findings from Sect. 2. To this end, for any pair (exit position, pick-up point), let β and γ denote the same angles as in Sect. 2, i.e., β is the angle between the direction of movement of R_1 at the (potential) exit position and the line connecting exit position and pick-up point and γ the angle between this line and the direction of movement of R_2 at the (potential) pick-up point. Note that when both robots move along the perimeter, we have $\beta = \gamma$ for reasons of symmetry.

For calculating distances, observe that for any two points on the perimeter with a distance of w *along the perimeter*, their euclidian distance is $2\sin(w/2)$. For instance, the value of $|\widehat{AI_1}|$ is equal to the solution of the equation $x +$

[4] These parameters are chosen in a way that for the (only) three possible global worst-case exit positions (determined in the following), the evacuation times are the same up to numerical precision. While the parameter values were determined numerically, we give a rigorous proof for the correctness of the claimed evacuation time.

$2\sin((x+y)/2) = y$, where in our case $y = 2.62843$. For the given parameters, we obtain $|\widehat{AI_1}| \approx 0.63196$, $|\widehat{AI_2}| \approx 2.5837$ and $|\widehat{AI_3}| = 2.62843$. Examining the first three segments one by one, we obtain the following lemma:

Lemma 2. *If there is a (global) worst-case exit position in the first segment, then this exit position is at* I_1*. If there is a (global) worst-case exit position in the second segment, then it is at* I_1 *or* I_2*. If there is a (global) worst-case exit position in the third segment, then it is at* I_2 *or* I_3*.*

For our examination of the fourth segment, we add a virtual point I_3' to the fourth segment that coincides with I_3, but has the additional property that if the exit is at I_3', then R_1 will only find the exit after performing its cut. The reason for this is that without the addition of I_3' the fourth segment is half-open which makes it possible that there is a sequence of exit positions with increasing evacuation times that converges towards I_3 and for which there is no exit position that has a larger evacuation time than all exit positions in the sequence.

Lemma 3. *If there is a (global) worst-case exit position in the fourth segment, then this exit position is at* I_3'*.*

Observe that the evacuation time for the exit at I_3' cannot be smaller than the evacuation time for the exit at I_3 since in the latter case R_1 could just simulate the former case which is worse than activating the meeting protocol right away. Combining this fact with Lemmas 2 and 3, we obtain the following theorem:

Theorem 4. *For* $y = 2.62843$*,* $\alpha = \pi/4$ *and* $d = 0.48793$*, the worst-case exit placement for Algorithm* $A(y, \alpha, d)$ *is at* I_1*,* I_2 *or* I_3'*.*

In order to determine the evacuation time for the worst-case exit, we simply take the maximum of the evacuation times for the exit placements at these three locations. We obtain evacuation times of approximately 5.6249, 5.62488 and 5.62491 for I_1, I_2 and I_3', respectively. Hence, the evacuation time for the worst-case exit is approximately 5.62491. Thus, we obtain the following corollary:

Corollary 5. *For* $y = 2.62843$*,* $\alpha = \pi/4$ *and* $d = 0.48793$*, the evacuation time of Algorithm* $A(y, \alpha, d)$ *is at most* 5.625*.*

Observe that, if the length of the cut is not changed, then altering the shape or angle of the cut does not affect the evacuation times for the exit positions at I_1, I_2 and I_3'. Thus, by Theorem 4, in such a case the overall evacuation time cannot decrease. We cast this insight into the following corollary:

Corollary 6. *For* $y = 2.62843$*,* $\alpha = \pi/4$ *and* $d = 0.48793$*, the evacuation time of Algorithm* $A(y, \alpha, d)$ *cannot be improved by altering shape or angle of the cut.*

Since all the inequalities in Lemmas 2 and 3 are not sharp, we can choose α in some reasonable range without compromising our evacuation time.[5] As long as the chosen α ensures that there is no worst-case exit placement such that R_2 is picked up on the cut (without the start and end points of the cut), Corollary 5 holds. The value of exactly $\pi/4$ for α is chosen for the reason of convenience.

[5] The same holds for the shape of the cut, by the same reason.

4 Conclusion

In this paper, we studied the evacuation of two robots from a disk using non-wireless communication. We presented a new tool for the analysis of evacuation algorithms for any area shape by showing that a strong local condition has to be satisfied in order for an exit to be worst-case placed. Using this tool and further insights, e.g., about the nature of forced meetings and the irrelevance of the chosen shape and angle in some range, we improved the state-of-the-art algorithm and gave indicators for where to look for further improvement.

However, we believe that our improved upper bound on the evacuation time is already very close to the tight bound that is the correct answer. We do not believe that our upper bound is optimal (up to numerical precision) because of the following reason: Imagine an additional second cut of very small depth ("ε-cut") close to the point opposite of the point on the perimeter where the robots start their search. If we choose the position (and the angle and depth) of this ε-cut appropriately, then the evacuation time for the exit at I'_3 will be improved since R_1 will pick R_2 up at around the tip of the ε-cut which is somewhat closer to I'_3 than if there was no such ε-cut. Now we can make small changes to position and depth of the first cut that result in improving the evacuation times for the exit positions at I_1 and I_2 while increasing the previously decreased evacuation time for the exit position at I'_3. By finding the parameters that again lead to equal evacuation times for these three exits, the overall evacuation time is improved. If one is careful not to let other points become worse exit positions, this approach can even be applied iteratively. However, the improvement in the evacuation time achieved by the collection of these very small cuts is negligibly small, even compared to the improvement given by our algorithm.

While the lower bound is still a long way from our upper bound, it is hard to imagine how an improvement to our algorithm apart from the ε-cuts might look like. In fact, we conjecture that, apart from these ε-cuts and numerical precision, the algorithm we presented is indeed optimal.

References

1. Alpern, S.: The rendezvous search problem. SIAM J. Control Optim. **33**(3), 673–683 (1995)
2. Baeza-Yates, R.A., Culberson, J.C., Rawlins, G.J.E.: Searching with uncertainty extended abstract. In: Karlsson, R., Lingas, A. (eds.) SWAT 1988. LNCS, vol. 318, pp. 176–189. Springer, Heidelberg (1988). doi:10.1007/3-540-19487-8_20
3. Beck, A., Newman, D.J.: Yet more on the linear search problem. Isr. J. Math. **8**(4), 419–429 (1970)
4. Borowiecki, P., Das, S., Dereniowski, D., Kuszner, Ł.: Distributed evacuation in graphs with multiple exits. In: Suomela, J. (ed.) SIROCCO 2016. LNCS, vol. 9988, pp. 228–241. Springer, Cham (2016). doi:10.1007/978-3-319-48314-6_15
5. Chrobak, M., Gasieniec, L., Gorry, T., Martin, R.: Group search on the line. In: Italiano, G.F., Margaria-Steffen, T., Pokorný, J., Quisquater, J.-J., Wattenhofer, R. (eds.) SOFSEM 2015. LNCS, vol. 8939, pp. 164–176. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46078-8_14

6. Czyzowicz, J., Dobrev, S., Georgiou, K., Kranakis, E., MacQuarrie, F.: Evacuating two robots from multiple unknown exits in a circle. In: ICDCN (2016). doi:10.1145/2833312.2833318
7. Czyzowicz, J., Gasieniec, L., Gorry, T., Kranakis, E., Martin, R., Pajak, D.: Evacuating robots via unknown exit in a disk. In: Kuhn, F. (ed.) DISC 2014. LNCS, vol. 8784, pp. 122–136. Springer, Heidelberg (2014). doi:10.1007/978-3-662-45174-8_9
8. Czyzowicz, J., Georgiou, K., Kranakis, E., Narayanan, L., Opatrny, J., Vogtenhuber, B.: Evacuating robots from a disk using face-to-face communication (extended abstract). In: Paschos, V.T., Widmayer, P. (eds.) CIAC 2015. LNCS, vol. 9079, pp. 140–152. Springer, Cham (2015). doi:10.1007/978-3-319-18173-8_10
9. Czyzowicz, J., Kranakis, E., Krizanc, D., Narayanan, L., Opatrny, J., Shende, S.: Wireless autonomous robot evacuation from equilateral triangles and squares. In: Papavassiliou, S., Ruehrup, S. (eds.) ADHOC-NOW 2015. LNCS, vol. 9143, pp. 181–194. Springer, Cham (2015). doi:10.1007/978-3-319-19662-6_13
10. Dessmark, A., Fraigniaud, P., Pelc, A.: Deterministic rendezvous in graphs. In: Battista, G., Zwick, U. (eds.) ESA 2003. LNCS, vol. 2832, pp. 184–195. Springer, Heidelberg (2003). doi:10.1007/978-3-540-39658-1_19
11. Emek, Y., Langner, T., Stolz, D., Uitto, J., Wattenhofer, R.: How many ants does it take to find the food? In: Halldórsson, M.M. (ed.) SIROCCO 2014. LNCS, vol. 8576, pp. 263–278. Springer, Cham (2014). doi:10.1007/978-3-319-09620-9_21
12. Feinerman, O., Korman, A.: Memory lower bounds for randomized collaborative search and implications for biology. In: Aguilera, M.K. (ed.) DISC 2012. LNCS, vol. 7611, pp. 61–75. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33651-5_5
13. Feinerman, O., Korman, A., Lotker, Z., Sereni, J.-S.: Collaborative search on the plane without communication. In: PODC (2012). doi:10.1145/2332432.2332444
14. Förster, K.-T., Nuridini, R., Uitto, J., Wattenhofer, R.: Lower bounds for the capture time: linear, quadratic, and beyond. In: Scheideler, C. (ed.) Structural Information and Communication Complexity. LNCS, vol. 9439, pp. 342–356. Springer, Cham (2015). doi:10.1007/978-3-319-25258-2_24
15. Förster, K.-T., Wattenhofer, R.: Directed graph exploration. In: Baldoni, R., Flocchini, P., Binoy, R. (eds.) OPODIS 2012. LNCS, vol. 7702, pp. 151–165. Springer, Heidelberg (2012). doi:10.1007/978-3-642-35476-2_11
16. Fraigniaud, P., Ilcinkas, D., Peer, G., Pelc, A., Peleg, D.: Graph exploration by a finite automaton. In: Fiala, J., Koubek, V., Kratochvíl, J. (eds.) MFCS 2004. LNCS, vol. 3153, pp. 451–462. Springer, Heidelberg (2004). doi:10.1007/978-3-540-28629-5_34
17. Kranakis, E., Krizanc, D., Rajsbaum, S.: Mobile agent rendezvous: a survey. In: Flocchini, P., Gasieniec, L. (eds.) SIROCCO 2006. LNCS, vol. 4056, pp. 1–9. Springer, Heidelberg (2006). doi:10.1007/11780823_1
18. Lamprou, I., Martin, R., Schewe, S.: Fast two-robot disk evacuation with wireless communication. In: Gavoille, C., Ilcinkas, D. (eds.) DISC 2016. LNCS, vol. 9888, pp. 1–15. Springer, Heidelberg (2016). doi:10.1007/978-3-662-53426-7_1
19. Megow, N., Mehlhorn, K., Schweitzer, P.: Online graph exploration: new results on old and new algorithms. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011. LNCS, vol. 6756, pp. 478–489. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22012-8_38
20. Nowakowski, R.J., Winkler, P.: Vertex-to-vertex pursuit in a graph. Discret. Math. **43**(2–3), 235–239 (1983). doi:10.1016/0012-365X(83)90160-7
21. Parsons, T.D.: Pursuit-evasion in a graph. In: Alavi, Y., Lick, D.R. (eds.) Theory and Applications of Graphs. Lecture Notes in Mathematics, vol. 642, pp. 426–441. Springer, Heidelberg (1978)