Randomized Gathering of Asynchronous Mobile Robots

Debasish Pattanayak[1], John Augustine[2], and Partha Sarathi Mandal[1]

[1]Department of Mathematics, Indian Institute of Technology Guwahati, India
[2]Department of Computer Science & Engineering, Indian Institute of Technology Madras, India

Abstract

This paper revisits the widely researched *gathering* problem for two robots in a scenario which allows randomization in the asynchronous scheduling model. The scheduler is considered to be the adversary which determines the activation schedule of the robots. The adversary comes in two flavors, namely, oblivious and adaptive, based on the knowledge of the outcome of random bits. The robots follow *wait-look-compute-move* cycle. In this paper, we classify the problems based on the capability of the adversary to control the parameters such as wait time, computation delay and the speed of robots and check the feasibility of gathering in terms of adversarial knowledge and capabilities. The main contributions include the possibility of gathering for an oblivious adversary with (i) zero computation delay; (ii) the sum of wait time and computation delay is more than a positive value. We complement the possibilities with an impossibility. We show that it is impossible for the robots to gather against an adaptive adversary with non-negative wait time and non-negative computation delay. Finally, we also extend our algorithm for multiple robots with merging.

1 Introduction

1.1 Backgorund and Motivation

Recently, it has piqued the interest of researchers to use small and simple robots to achieve tasks which may seem complicated for a single robot to perform. These tiny robots, otherwise known as *swarm robots*, have various advantages, namely, robustness and scalability. Some common problems which has been addressed in the literature include *gathering* [1, 5, 6, 7, 12, 13, 20], *pattern formation* [15, 17, 18, 25], *scattering* [9, 21] and *flocking* [10, 11]. The problem of *gathering* had been investigated in the presence of byzantine faults [2, 16] and crash faults [4, 5, 24].

The widely accepted *weak robots* model for mobile robots was introduced by Flocchini et al. [17]. The robots are generally considered to be *homogeneous*, i.e., execute the same algorithm; *oblivious*, i.e., do not have any knowledge of past computations; *anonymous*, i.e., do not have any identifiers. No message exchange happens among the robots, hence, *silent*. The robots are dimensionless *point* robots. Each robot operates in *wait-look-compute-move* cycles. *Gathering* is a well-known problem that requires the robots to meet at a point which is not specified a priori. Gathering for two robots is also known as *Rendezvous* problem for two robots. Likewise, the gathering problem can be considered to be a point formation problem. The solution to the Rendezvous problem depends on the level of synchrony of the scheduler. There are three basic types of schedulers considered in the literature, namely, fully-synchronous ($FSYNC$), semi-synchronous ($SSYNC$) and asynchronous ($ASYNC$). In $SSYNC$, the global time is divided into discrete rounds and a subset of robots are activated in each round by the scheduler. $FSYNC$ can be considered as a special case of $SSYNC$, where all the robots are activated in each round. In $SSYNC$, the look states of all activated robots are aligned.

1.2 Related Works

The gathering problem for two robots is trivial in $FSYNC$ model. It has been proved that gathering two robots is impossible with a deterministic algorithm in the $SSYNC$ model [26]. Gathering is solved in the $SSYNC$ model with randomization by Izumi et al. [20], which takes a constant number of rounds in expectation with a local-weak multiplicity detection and scattered initial configuration – all robots occupy distinct initial positions. To achieve the constant number of rounds, they assume that the scheduler is bounded, i.e., all robots activate at least once for some constant number of activations of any robot.

Lights were introduced by Das et al. [14] as external persistent memories to expand capabilities of robots. In robots with lights model, Rendezvous has been solved with two color lights, which is optimal [19] in the $ASYNC$ model with non-rigid motion. Okumura et al. [23] further categorized the scheduler into multiple types including rigid and non-rigid motion of robots. The fault-tolerant gathering of robots has been solved in the presence of one fault [1] to multiple faults [5, 24]. Pattanayak et al. [24] also introduced the $ASYNC_{IC}$ model – also known as LC-atomic $ASYNC$ by Okumura et al. [23]. Randomization has also been used in [8, 10] to elect a leader and in [21] for choosing a direction of movement in scattering. Characterization of a continuous adversary in a discrete scenario has been addressed by Bampas et al. [3] for the Rendezvous problem of two mobile agents in graphs.

In this paper, we approach the problem of gathering robots in multiple variations of the $ASYNC$ model. This paper explores two different possibilities for the notion of an adversary for the randomized gathering of robots.

1.3 Our Contribution

We approach the problem to determine the tradeoff between randomization and asynchrony. We define the particulars of adversarial control and characterize them according to the access and knowledge of parameters such as wait time, computation delay, the speed of robots, and the outcome of random bits. We use the terms possible, improbable, and impossible to denote the results in various models. If the probability is non-zero, we say it is possible. Here improbable means that the probability of success is zero, but the set of successful outcomes is non-empty. For example, the probability of choosing $1/2$ in the interval $[0, 1]$ is zero, but it is possible to choose $1/2$. If the set of successful outcomes is empty, then we say that it is impossible. For example, choosing 2 in the interval $[0, 1]$ is impossible.

- It is possible to gather two robots with non-zero wait time and zero computation delay for an oblivious adversary, where the algorithm knows the speed ratio of two robots.

- It is improbable for two robots to gather if the algorithm does not know the speed ratio given zero wait time and zero computation delay.

- It is possible for two robots to gather if one robot has zero wait time and zero computation delay.

- It is also possible to gather two robots with non-negative wait time and non-negative computation delay if the sum of wait time and computation delay in any cycle is always greater than a positive value, τ. The estimated number of total looks by both of the robots for gathering is $18(\log_2(\delta/\tau) + 1)$ where δ is the initial distance between them.

- For the adaptive adversary, we show that it is impossible for two robots to gather for non-zero wait time and computation delay.

- Finally, we also extend the algorithm for gathering multiple robots with merging.

We summarize the findings in Table 1.

Table 1: Summary of results for gathering two robots

Result	Wait Time ($\mathcal{W}$)	Computation Delay ($\mathcal{C}$)	Speed Ratio (α)	Gathering
Oblivious Adversary				
Theorem 3	$\mathcal{W} \geq 0$	$\mathcal{C} = 0$	1	Possible
Theorem 4	$\mathcal{W} \geq 0$	$\mathcal{C} = 0$	Known	Possible
Theorem 5	$\mathcal{W} = 0$	$\mathcal{C} = 0$	Unknown	Improbable
Theorem 6	$\mathcal{W}_1 = 0$ $\mathcal{W}_2 \geq 0$	$\mathcal{C}_1 = 0$ $\mathcal{C}_2 \geq 0$	Known	Possible
Theorem 12	$\mathcal{W} + \mathcal{C} > \tau$		1	Possible
	$\mathcal{W} \geq 0$	$\mathcal{C} \geq 0$	1	?
Adaptive Adversary				
Theorem 14	$\mathcal{W} \geq 0$	$\mathcal{C} \geq 0$	1	Impossible

1.4 Paper Organization

The rest of the paper is organized as the following. In Section 2, we describe the model and notations. Then we describe the results of gathering for an oblivious adversary in Section 3, for various models, like zero computation delay, heterogeneous speed of the robots and analyze random bit complexity. In Section 4, we show the impossibility of gathering for an adaptive adversary. In Section 5, we explore the gathering of multiple robots with merging before concluding in Section 6.

2 Preliminaries

2.1 Model

We assume the robots to be anonymous, oblivious, silent represented as points on the Euclidean plane. The robots follow the *wait-look-compute-move* cycle. When the robots are in *wait* state, the robots are idle. In the *look* state, the robots obtain a snapshot of the surrounding and in the *compute* state they compute a destination based on the snapshot obtained. Finally, in the *move* state, the robots move towards the destination. The robots do not have any agreement on the coordinate system. There are two types of movement considered in the literature, namely, *rigid* movement and *non-rigid* movement. In rigid movement, the robot reaches the destination in the same activation cycle. In non-rigid movement, the robot moves at least a predefined distance towards the destination in an activation cycle. In this paper, we consider the rigid movement of robots. It is easy to observe that the non-rigid movement behaves as a rigid movement if the distance to travel becomes less than the predefined distance.

The robots have random bits with which they decide the destination as per the algorithm. To characterize the problems, we consider the scheduler to be the *adversary*. The adversary knows the algorithm in advance, but may or may not have access to the random bits used by the algorithm in each cycle [22]. Based on the above, the adversary is categorized into two types, namely, *oblivious*: does not know the outcome of random bits and *adaptive*: knows the outcome of *past* random bits. Note that, the word *oblivious* is used for denoting robots without memory of past computations and adversary without knowledge of the outcome of random bits.

We consider the general $ASYNC$ model for the scheduler, where each robot independently executes the *wait-look-compute-move* cycle. We consider various models which make the adversary control some aspect of a cycle. When a robot R_1 looks, it obtains the position of robot R_2, which may or may not be moving. If R_2 is moving, then the position of R_2 obtained at a particular instant of time that corresponds to the distance observed by R_1. So the look instant $\mathcal{L}$ divides the time a robot remains stationary in a cycle into two parts where the former is wait time, $\mathcal{W}$, and the latter is computation delay, $\mathcal{C}$. Since until the look happens and the robot knows the position of the other robot, it can be considered *idle*, and after that, it is doing *computation* or executing

the algorithm. Fig. 1 shows the division of the look state of the $ASYNC$ into wait and compute.
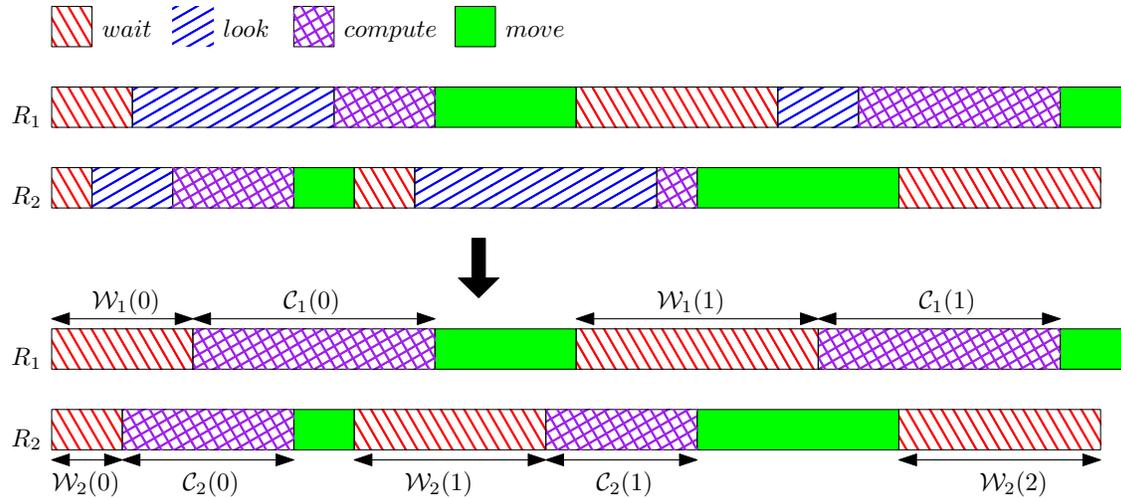

Figure 1: The top figure shows the *wait-look-compute-move* steps and the bottom figure shows the division of *look* state into wait time and computation delay

The destination is decided by the algorithm based on the outcome of random bits and the position of the other robot. The time required by a robot to move to its destination depends on the speed of the robot. The speeds of the robots may be the same or different. We assume that the speed of a robot remains the same throughout the execution of the algorithm. We classify the problems based on the capabilities of the adversary to control wait time, computation delay, and the speed of robots. When a robot looks and finds the other robot at its position, it decides that it has gathered. We assume that the robot which has decided that it has gathered stops moving thenceforth.

2.2 Notations

We use the following notations throughout the paper.

- Time periods are denoted as $\mathcal{W}$ for wait time and $\mathcal{C}$ for computation delay.

- To distinguish between the two robots' wait time and computation delay, we use $\mathcal{W}_j$ and $\mathcal{C}_j$ for robot R_j, where $j \in \{1, 2\}$.

- The sequence of wait time periods are denoted as $\{\mathcal{W}_j(0), \mathcal{W}_j(1), \mathcal{W}_j(2), \dots\}$ and computation delay as $\{\mathcal{C}_j(0), \mathcal{C}_j(1), \mathcal{C}_j(2), \dots\}$ corresponding to cycles $\{0, 1, 2, \dots\}$ for R_j.

- We also denote the look instants as $\{\mathcal{L}_j(0), \mathcal{L}_j(1), \mathcal{L}_j(2), \dots\}$ corresponding to cycles $\{0, 1, 2, \dots\}$ for R_j.

- δ is the initial distance between the robots.

- τ is the lower bound on the sum of wait time and computation delay for a particular robot in a cycle.

- α is the ratio of the speed of two robots.

2.3 λ-class Algorithms

Gathering two robots in multiple dimensions is equivalent to gathering in one dimension. We formalize the statement using the following results.

4

Lemma 1. *There exists an algorithm ϕ which gathers two robots in $\mathbb{R}^d$, if and only if there exists an algorithm ϕ' which gathers two robots in one dimension, i.e., the line joining two robots.*

Proof. We define a coordinate system for $\mathbb{R}^d$ with mid point of line joining p_1 and p_2 as the origin and p_1 as the positive x-axis. Suppose the destination decided by the algorithm ϕ in $\mathbb{R}^d$ is some point p. Take the projection of p on the line joining p_1 and p_2 as p'. Moving to the position p' in one dimension is the same as moving to p in $\mathbb{R}^d$. Now, we can always find a corresponding position on the line for each destination position provided by the algorithm ϕ. If the sequence of destinations gathers the two robots in $\mathbb{R}^d$, then the corresponding projections will also gather the two robots in one dimension. Conversely, if an algorithm ϕ' gathers two robots in one dimension, i.e., the line joining them, then ϕ' is sufficient to gather two robots in $\mathbb{R}^d$. $\square$

From Lemma 1, it is sufficient that the robots only move on the line joining two robots. All possible algorithms for gathering two robots in one dimension can be classified based on the parameter $\lambda \in \mathbb{R}$. In the *move* state a robot can move $\lambda\delta$ distance towards the other robot, where δ is the distance between two robots. For example, if $\lambda = 0$, a robot stays in its position, and if $\lambda = 1$, it moves to the position of the other robot.

Let us first describe the impossibility of gathering of two robots in *SSYNC* model [26]. The capability that the adversary holds in case of *SSYNC* model is whether to activate robots in a particular round or not.

Example 2. *If the deterministic algorithm has $\lambda = 1$, then the adversary would activate both robots in the same round.*

(a) $\lambda = 1$ (b) $\lambda = 1/2$

Figure 2: Execution of deterministic algorithms with $\lambda = 1/2$ and 1 in *SSYNC* model

If the deterministic algorithm has $\lambda = 1/2$, then the adversary would activate only one of them in a particular round. This would certainly result in reduction of the distance between the robots, but the distance would never be zero. Fig. 2 shows a sample execution of the robots' behavior for $\lambda = 1$ and $\lambda = 1/2$. All the figures included in this paper follow a convention of representing time in the x-axis and distance in the y-axis. We assume R_1 starts at $(0, \delta)$ and R_2 start at $(0, 0)$.

A randomized algorithm with $\lambda = 2$ and -1, has been presented by Izumi et al. [20], which gathers two robots in *SSYNC* model.

3 Oblivious Adversary

An oblivious adversary is not aware of the random bits produced by the algorithm. So an oblivious adversary decides the wait time and computation delay regardless of the outcome of the sequence of random bits generated. In other words, the adversary decides the sequence of wait times and computation delays for each robot before the start of the algorithm. The adversary may know the ratio of the speed between the robots. In this section, we show the possibility of gathering with zero computation delay, and the speed ratio is known to the algorithm. If the speed ratio is controlled by the adversary – unknown to the algorithm – the adversary can ensure that the probability of gathering is zero even with zero wait time and zero computation delay (without any adversarial control). If the adversary controls one of the robots and the other has zero wait time and zero computation delay, then the probability of gathering is also positive, albeit small.

3.1 Gathering in $ASYNC_{IC}$ model

First, we prove that two robots with the same speed can gather in the $ASYNC_{IC}$ model [24], i.e., the computation delay is zero ($\mathcal{C} = 0$). In this model, the robots move immediately after they look. For the proof, we gradually establish inequalities, which limit the choices of an adversary to prevent gathering. Finally, we show that for all choices of the adversary, there exists an algorithmic step which leads to the gathering of two robots.

Theorem 3. *For an oblivious adversary, it is possible to gather two robots with the same speed in the $ASYNC_{IC}$ model (asynchronous with zero computation delay), i.e., $\mathcal{W} \geq 0$ and $\mathcal{C} = 0$.*

Proof. Each robot when activated, looks at the other robot to determine the position of other robot and chooses a λ value based on the following three options (with equal probability and independent of other choices): (i) $\lambda = 1$, (ii) $\lambda = 1/2$, or (iii) a real value chosen uniformly at random from $(0, 1)$. We now show a sequence of interactions between the algorithm and the adversary that – with positive probability – will guarantee that the robots will gather.

We first introduce some notations. Since the computation delay is zero, let the wait time periods provided by the adversary be $\{\mathcal{W}_1(0), \mathcal{W}_1(1), \ldots\}$ and $\{\mathcal{W}_2(0), \mathcal{W}_2(1), \ldots\}$. We define the gap, γ as the time difference between the look of R_1 and R_2. Since both robots have a speed of one unit distance per unit time, in γ amount of time, a robot can cover γ distance. If the gap γ between look of two robots is more than the distance δ, between them, and R_2 is idle when R_1 looks, then R_1 can reach R_2 before R_2 looks as shown in Fig. 3. In that case, to prevent gathering adversary has to choose the gap such that $|\gamma| < \delta$. If $\gamma = 0$, then both robots look at the same time and gather at mid point provided they choose $\lambda = 1/2$. Without loss of generality, assume that $\mathcal{W}_1(0) > \mathcal{W}_2(0)$. Denote, $\gamma_0 = \mathcal{W}_1(0) - \mathcal{W}_2(0)$ as shown in Fig. 4. To avoid gathering, the adversary should choose $\mathcal{W}_1(0)$ and $\mathcal{W}_2(0)$ such that

$$-\delta < \gamma_0 < \delta \text{ and } \gamma_0 \neq 0 \tag{1}$$

If the adversary manages to enforce Inequality (1), we consider (as a first algorithmic step) the event that the algorithm chooses the following specific λ values (which are anyway chosen with probability 1/9).

Step 1: ($\lambda_1 = \lambda_2 = 1$)   Though the robots start with a gap γ_0, both robots finish moving at the same time because they have the same speed. After both robots finish their movement, the distance between them is $\delta - \gamma_0$. The next gap between looks is defined as $\gamma_1 = \mathcal{W}_1(1) - \mathcal{W}_2(1)$ because they had finished at the same time. Further, we have the inequalities for the second look of robots if they do not gather as the following.

$$-\delta + \gamma_0 < \gamma_1 < \delta - \gamma_0 \text{ and } \gamma_1 \neq 0. \tag{2}$$

Recall that if $\gamma_1 = 0$, then they will gather with constant probability when both choose $\lambda = 1/2$. So for the rest of the argument, we will condition on $\gamma_1 \neq 0$. Under this condition, next algorithmic step we consider (which again occurs with constant probability) is the following choice of λ values.

Step 2: ($\lambda_1 = 1/2, \lambda_2 \in U(0, 1)$)   Observe Fig. 4. If $\lambda_2 = (\delta + \gamma_0)/2\delta$, then R_1 and R_2 meet at M. Now we estimate the probability that R_2 would choose λ_2 such that they will gather in the next cycle. For the robots to gather in the next cycle, whichever looks first has to find the other robot in idle state.

Case 1: Now, the algorithm can choose λ_2 value such that it creates the situation in Fig. 3. Suppose, R_1 looks first in the next cycle, then it has to find R_2 in the idle state – second look of R_1 (at $\mathcal{W}_1(0) + (\delta - \gamma_0)/2 + \mathcal{W}_1(1)$) occurs after R_2 finishes moving (at $\mathcal{W}_2(0) + \lambda_2\delta$) and before it looks for second time (at $\mathcal{W}_2(0) + \lambda_2\delta + \mathcal{W}_2(1)$). So,

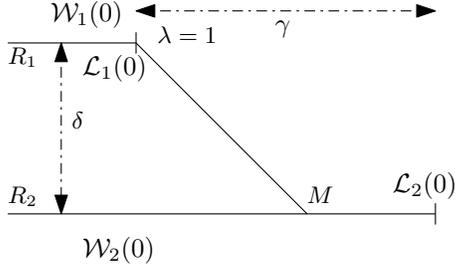$$\lambda_2\delta < (\delta + \gamma_0)/2 + \mathcal{W}_1(1) < \lambda_2\delta + \mathcal{W}_2(1) \tag{3}$$

6

Figure 3: Gathering of two robots when the difference between looks, γ, is greater than the current distance between the robots, δ.
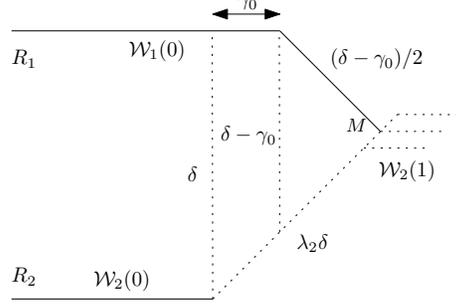

Figure 4: Gathering of two robots for $\lambda_1 = 1/2$ and $\lambda_2 \in U(0,1)$.

.

Also, the distance between them at that point of time $(\delta - \lambda_2\delta - (\delta - \gamma_0)/2)$ should be less than the gap between looks $(\lambda_2\delta + \mathcal{W}_2(1) - (\delta + \gamma_0)/2 - \mathcal{W}_1(1))$.

$$|\lambda_2\delta - (\delta + \gamma_0)/2| < \lambda_2\delta - (\delta + \gamma_0)/2 - \gamma_1 \tag{4}$$

- If $\lambda_2\delta > (\delta + \gamma_0)/2$, then $\gamma_1 < 0$ from Inequality (4). The range of λ_2 for which Inequality (3) and (4) are satisfied can be given by

$$\lambda_2 \in \Big(\max\big((\delta + \gamma_0)/2\delta, (\delta + \gamma_0 + 2\gamma_1)/2\delta\big), (\delta + \gamma_0 + 2\mathcal{W}_1(1))/2\delta \Big)$$

  Since $\gamma_1 < 0$, the probability of choosing such a λ_2 is $\mathcal{W}_1(1)/\delta$ – the length of the smaller interval since it is chosen uniformly at random from $(0,1)$.

- If $\lambda_2\delta < (\delta + \gamma_0)/2$, then $\lambda_2 > (\delta + \gamma_0 + \gamma_1)/2\delta$ from Inequality (4) and $\gamma_1 < 0$ from Inequality (3). So, we have

$$\lambda_2 \in \Big(\max\big((\delta + \gamma_0 + 2\gamma_1)/2\delta, (\delta + \gamma_0 + \gamma_1)/2\delta\big), (\delta + \gamma_0)/2\delta \Big)$$

  The probability of choosing such a λ_2 is $-\gamma_1/2\delta$, since $\gamma_1 < 0$.

Case 2: Conversely, if R_2 looks first in the next step, then it has to find R_1 in the idle state. So,

$$(\delta + \gamma_0)/2 < \lambda_2\delta + \mathcal{W}_2(1) < (\delta + \gamma_0)/2 + \mathcal{W}_1(1) \tag{5}$$

Similarly, the distance between them at that point of time should be less than the gap between looks.

$$|\lambda_2\delta - (\delta + \gamma_0)/2| < (\delta + \gamma_0)/2 + \gamma_1 - \lambda_2\delta \tag{6}$$

- If $\lambda_2\delta < (\delta + \gamma_0)/2$, then $\gamma_1 > 0$. The range of λ_2 for which Inequality (5) and (6) are satisfied can be given by

$$\lambda_2 \in \Big((\delta + \gamma_0 - 2\mathcal{W}_2(1))/2\delta, (\delta + \gamma_0)/2\delta \Big)$$

  The probability of choosing such λ_2 is $\mathcal{W}_2(1)/\delta$.

- If $\lambda_2\delta > (\delta + \gamma_0)/2$, then $\lambda_2 < (\delta + \gamma_0 + \gamma_1)/2\delta$. So, we have

$$\lambda_2 \in \Big((\delta + \gamma_0 - 2\mathcal{W}_2(1))/2\delta, \min((\delta + \gamma_0 + 2\gamma_1)/2\delta, (\delta + \gamma_0 + \gamma_1)/2\delta) \Big)$$

  The probability of choosing such a λ_2 is $\mathcal{W}_1(1)/\delta$, if $\gamma_1 < 0$ and $(\mathcal{W}_1(1) + \mathcal{W}_2(1))/2\delta$, if $\gamma_1 > 0$.

As the robot has a probability of $1/3$ for choosing λ values from 1, $1/2$ and $U(0,1)$, we have a multiplier of $1/27$ for the three choices of λ values made in total by the two robots.

Recall that $\gamma_1 \neq 0$ from Inequality (2). So the probability of gathering is dependent on the values $\mathcal{W}_1(1)$ and $\mathcal{W}_2(1)$, the adversary can choose these to be zero. Now, we consider the following algorithmic step to show a constant probability of gathering if $\mathcal{W}_1(1)$ or $\mathcal{W}_2(1)$ is equal to zero.

7

Step 3: $(\lambda_1 = 1/2)$

Case 1: If $\mathcal{W}_1(1) = 0$, then R_1 looks again at M and finds R_2 at M and decides that it has gathered. Once R_1 decides it has gathered, it does not move. We can correspondingly assign λ values to be 1 and 1 for two consecutive looks of R_2 for the robots to gather. The probability of such an event is $1/27$.

Case 2: If $\mathcal{W}_2(1) = 0$, then R_2 can choose λ value $1/2$. It corresponds to two situations, given the value of γ_0.

- If $\gamma_0 \leq \delta/2$, then at the second look of R_2, R_1 is moving. So R_2 can choose $\lambda = 1/2$ again and gather at M.

- If $\gamma_0 > \delta/2$, then at the second look of R_2, R_1 is still in idle state. The gathering scenario repeats itself with $\gamma_0 \to \gamma_0 - \delta/2$ and $\delta \to \delta/2$. The adversary can choose $\mathcal{W}_2(l) = 0$ for $l \in \mathbb{N}$ to repeat this case. But, this situation can be repeated finitely many times before Inequality (7) holds since $\gamma_0 < \delta$ from Inequality (1).

$$\delta/2 + \delta/4 + \cdots + \delta/2^k > \gamma_0 \tag{7}$$

The associated probability is $1/3^{k+1}$.

Regardless of the value of wait times chosen by adversary, we show that the probability of gathering is always positive. This concludes the proof. $\square$

Next, we extend the possibility result of Theorem 3 to accommodate different speed robots where the algorithm knows the speed of robots. Though it is not possible for the algorithm to distinguish between the faster robot and the slower robot, the algorithm can still utilize the ratio of speed to achieve gathering.

Theorem 4. *It is possible to gather two robots if the algorithm knows the speed ratio, α, of the robots for an oblivious adversary in $ASYNC_{IC}$ model.*

Proof. In this proof, we recreate the same situations as in the proof of Theorem 3. Similar to the previous proof, we assume $\mathcal{W}_1(0) > \mathcal{W}_2(0)$. Let us normalize the speed of R_1 to 1 and denote α as the speed ratio of R_2 to R_1. Choosing $\lambda_1 = 1/(\alpha + 1)$ instead of $1/2$ leads to similar situation as Fig. 4. Likewise, we can determine the probability of choosing λ_2 in a small interval around the meeting point. Since the robots are anonymous, the algorithm must include the reciprocal of α also. So the other value of λ is $1/(1 + 1/\alpha) = \alpha/(\alpha + 1)$. $\square$

3.2 Gathering with Unknown α

Unlike Theorem 4, if the adversary controls the speed of robots, i.e., the algorithm does not have any knowledge about the speed of robots, then it is improbable for the robots to gather. The analogy follows from the fact that the adversary can change the speed of the robots in each round such that it is unlikely for both robots meet at the same point at the same time. Here improbable stands for a zero probability event where the set of successful outcomes is non-empty. The following theorem and proof shows that the algorithm has to choose a specific real number in an interval,

Theorem 5. *It is improbable for two robots to gather if the speed ratio α is controlled by the adversary with zero wait time and zero computation delay for an oblivious adversary.*

Proof. Consider the last look before two robots gather. For the two robots to gather, one robot would look at a time when the other robot is at its position. That can happen in the two scenarios, as shown in Fig. 5. Let the ratio of the speed of R_1 to R_2 is α with the speed of R_2 is normalized to 1. Here, $\alpha \in (0, \infty)$. For robot R_1 to reach the destination, it has to travel a distance of $\lambda\alpha\delta$ while the other robot travels a distance of $\lambda\delta$ in the same amount of time. If both the robots are
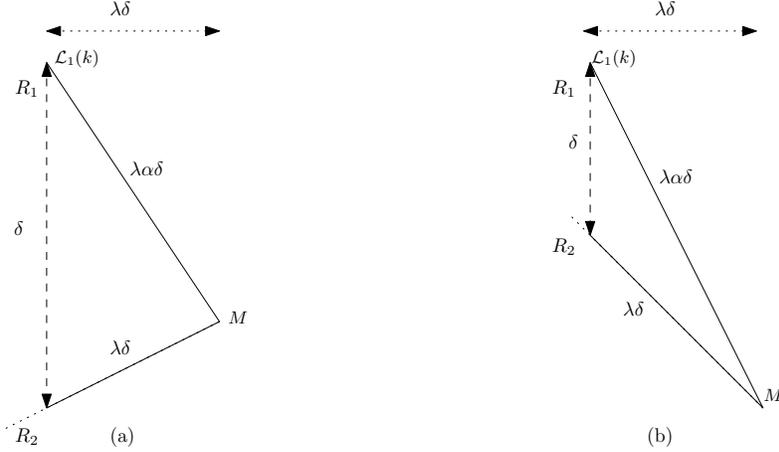
Figure 5: Improbability of gathering two robots with unknown α

traveling towards each other as shown in Fig. 5(a), then the sum of distance covered should be δ as observed by R_1 at $\mathcal{L}_1(k)$,

$$\lambda\alpha\delta + \lambda\delta = \delta \implies \lambda = 1/(1+\alpha)$$

In the other case, if both the robots are traveling in the same direction as shown in Fig. 5(b), then the difference of the distance should be δ,

$$\lambda\alpha\delta - \lambda\delta = \delta \implies \lambda = 1/(\alpha - 1)$$

If the algorithm chooses $\lambda \in \{1/(\alpha+1), 1/(\alpha-1)\}$, then the robots gather. The algorithm can choose any real number as λ value. The probability of choosing $\lambda \in \{1/(\alpha+1), 1/(\alpha-1)\}$ from $(-\infty, \infty)$ is equal to zero, if the algorithm does not know the value of α. Hence, it is improbable for the robots to gather. $\qquad\square$

3.3 Gathering of Robots when only one Robot controlled by the Adversary

Now, we consider a situation where one robot has fixed wait time and computation delay such that the adversary cannot manipulate the robot. Following result shows the possibility of gathering.

Theorem 6. *It is possible for two robots to gather if one robot has zero wait time and zero computation delay for an oblivious adversary if the algorithm knows the speed ratio α.*

Proof. Consider the situation shown in Fig. 6. Robot R_1 has both wait time and computation delay to be 0. We construct a scenario of gathering utilizing the situation in Theorem 5. If R_1 looks at R_2 when R_2 is in the move state and chooses $\lambda = 1/2$, then R_1 and R_2, both will cover half the distance before R_1 looks. At the next look of R_1, R_1 finds R_2 at its own position and decides that it has gathered. Now R_1 stays put. At R_2's next look, it can choose $\lambda = 1$ and they gather. For such a situation to happen, there are few constraints. R_1 chooses $\lambda = 1/2$ continuously until it observes R_2 is in the move state. We have the following equation

$$P(\delta/2 + \delta/4 + \delta/8 + \cdots + \delta/2^k > \mathcal{W} + \mathcal{C}) = P(\delta - \mathcal{W} - \mathcal{C} > \delta/2^k) = 1/2^k$$

This is true for some $k \in \mathbb{N}$. The probability of gathering is associated with the ratio $(\delta - t_i - t_c)/\delta$. Note that, for different speed robots, R_1 can choose $\lambda = 1/(\alpha+1)$ instead of $1/2$. Analysis would be likewise. $\qquad\square$
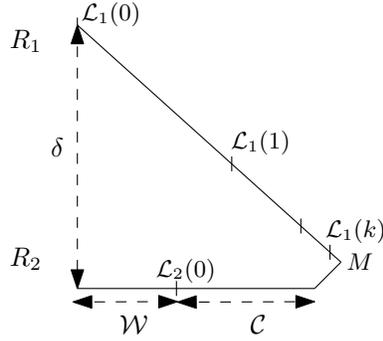
Figure 6: Gathering with R_1 having zero wait time and zero computation delay for $\alpha = 1$

3.4 Gathering of two same speed robots for $\mathcal{W} + \mathcal{C} \geq \tau$

As per the proof of Theorem 3, observe that the adversary can make the wait times infinitesimally smaller to have a very small probability albeit non-zero. In this section, we restrict the adversary to prevent it from making the wait time and computation delay arbitrarily small or zero. So the adversary must choose the wait time and computation delay such that the sum of the wait time and computation delay shall be greater than a predetermined constant τ. The look of a robot can occur at any time instance during that period. Consider an algorithm that chooses one of the λ values 1, 1/2 and 0 with probability 1/3 each (and independent of all other choices). First, we present some definitions and lemmas before going to the final result.

Definition 7 (Maximum Distance). *Distance δ between two robots is the maximum distance at time t, if the distance at any point of time $t' > t$ does not exceed δ.*

Note that, the instantaneous distance between the robots can be zero when the robots are crossing each other, but that does not represent the maximum distance between them.

We define a sequence of attempts starting from the initiation of the algorithm. Consider the first cycle of each robot. Let R_1 be the robot which has moved later in the first cycle at time t. Then we take the latest choice of λ value of R_1 and the choice of λ value of R_2 immediately prior to t. These two looks together make an attempt. Notice that, R_2 can have multiple looks before R_1 starts moving. We only consider the latest look of R_2 as part of the attempt. The subsequent attempt would consider the next cycles of both robots.

Definition 8 (Successful attempt). *An attempt is successful if the maximum distance between the robots is at most half of the maximum distance before the choices of λ values done by the robots.*

Lemma 9. *An attempt is successful with a probability at least 2/9.*

Proof. Let δ be the maximum distance between the two robots with R_1 at δ and R_2 at 0. Without loss of generality assume that R_2 is the robot which moves later. An attempt is successful if R_2 chooses $\lambda = 1/2$. Because, R_1 moves first, it can be at any position $x \in [0, \delta]$ when R_2 looks. R_1 can choose $\lambda = 1$ or $1/2$ to move. As R_2 moves to the midpoint, it will go to $x/2 \in [0, \delta/2]$.

- If R_1 has chosen $\lambda = 1$, then the robots would cross each other. The maximum distance between them is $x/2 \leq \delta/2$ if R_1 reaches 0 as shown in Fig. 7(a).

- If R_1 has chosen $\lambda = 1/2$, then the robots may or may not cross each other. If the robots cross each other, it is similar to the previous case with $\lambda = 1$ for R_1. If the robots do not cross each other, there are two situations depending on which robot looks first.

    - If R_1 looks first, then it may find R_2 before R_2 reaches 0, i.e., before R_2 is at the farthest distance from R_1. Let R_2 be at y when R_1 has looked. Now, $y \in [0, \delta)$. Since R_1 moves first, it finishes its movement first as it is moving only half the distance. Now,
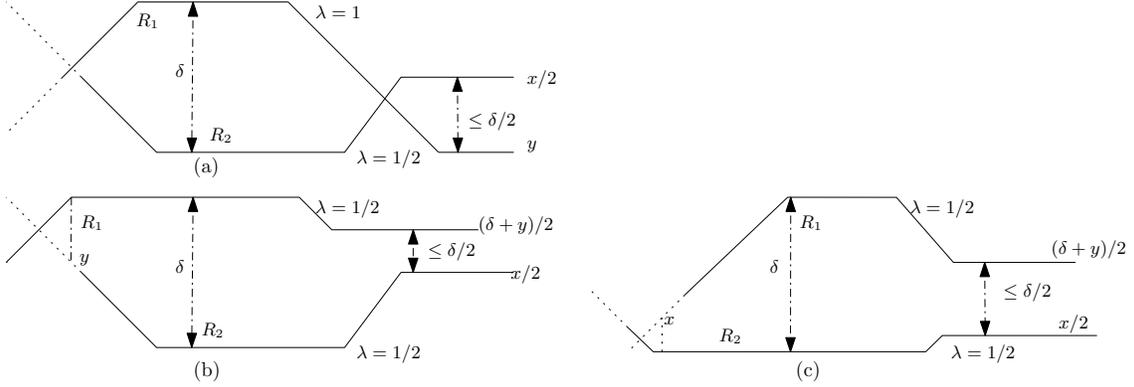
10

Figure 7: Occurrence of a phase when R_2 choses $\lambda = 1/2$ and moves later

R_2 can observe R_1 at any position between $x \in [(\delta + y)/2, \delta]$. So, the next distance between them is $(\delta + y)/2 - x/2 \leq (\delta + y)/2 - (\delta + y)/4 \leq \delta/2$ as shown in Fig. 7(b).

- If R_2 looks first, then $y \in [0, x/2]$. The next distance between them is $(\delta + y)/2 - x/2 \leq \delta/2 - x/4 \leq \delta/2$ as shown in Fig. 7(c).

Hence, the maximum distance of R_1 from R_2 is bound by $\delta/2$. The probability for R_1 to choose λ to be 1 or $1/2$ is $2/3$ and for R_2 to choose $\lambda = 1/2$ is $1/3$. So the probability of a successful attempt is at least $2/3 \times 1/3 = 2/9$. □

Definition 10 (Phase). *A phase is a set of consecutive attempts until a successful attempt.*

Lemma 11. *A phase contains at most 18 looks in expectation.*

Proof. Let X_i be a random variable which denotes the number of looks in a phase. Since a phase contains successive attempts until a successful attempt, let Y_i be the random variable denoting the number of attempts in a phase. From Lemma 9, each attempt has a success probability of $2/9$. Each attempt is independent of the previous attempts. So the number of attempts before a successful attempt follows a geometric distribution. Thus, $E[Y_j] = 1/(2/9) = 9/2$. Let Z be the random variable denoting the number of looks in an attempt. By definition, $Z \geq 2$. An attempt will finish if the robot which started first chooses λ value to be 1. The number of looks before the robot chooses $\lambda = 1$ also follows a geometric distribution with expectation 3. So, the expected number of looks in an attempt is 4. The number of looks in a phase is, $X_i = \sum_{j=1}^{Z} Y_j$. By Wald's Equation [27], $E[X_i] = E[Y_j]E[Z] = 4 \times 9/2 = 18$. □

Theorem 12. *It is possible to gather two robots if the sum of wait time and computation delay is always greater than some value τ for both the robots having the same speed.*

Proof. Let T be a random variable which denotes the number of phases required such that the maximum distance between two robots is less than τ. As each phase reduces the distance between robots by at least half the maximum distance in the previous phase, we can say that $T \leq \log_2(\delta/\tau)$. Let us estimate the total number of looks required for the maximum distance to become less than τ. Each look corresponds to a choice of λ value, which also signifies the random bit complexity of the process as a whole. Let X_i be a random variable which represents the number of looks in each phase. We have $E[X_i] = 18$ from Lemma 11. Let random variable X denote the total number of looks required for maximum distance to be less than τ. So,

$$X = \sum_{i=1}^{T} X_i$$

11

Using Wald's Equation [27], we have

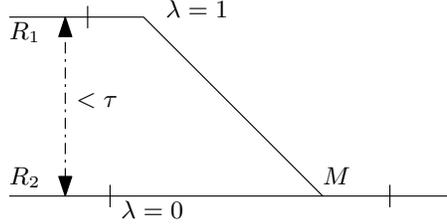$$E[X] = E[X_i]E[T] = 18E[T] \le 18\log_2(\delta/\tau)$$

Figure 8: Gathering of two robots when distance between them is less than τ.

Next, we describe the probability of gathering after the maximum distance between robots in a phase is less than τ. If the robot moving first chooses $\lambda = 1$ and the other chooses $\lambda = 0$ in the next attempt, that is sufficient for gathering as shown in Fig. 8. The gathering happens definitely because whichever robot looks after the robot moving first has finished moving, would find the other at the same location and decide that they have gathered. The probability of such an event is $1/9$. Hence, the expected number of attempts before the robots gather is 9. The number of looks for such an event is 2. All other combinations of λ values in two looks may not lead to gathering. So, the corresponding number of looks is 18. Expected number of looks required by both robots combined is at most $18(\log_2(\delta/\tau) + 1)$. This concludes the proof. $\qquad\square$

Remark 13. *Two robots can gather with two different values of λ (e.g., 0 and 1), so the number of random bits required per cycle is 1, which is optimal.*

4 Adaptive Adversary

An adaptive adversary is aware of all the outcome of random bits that appeared; consequently, it knows the destination of a robot in a cycle after the look instant. The adversary decides the computation delay of the current cycle and wait time of the next cycle for a robot after the look instant.
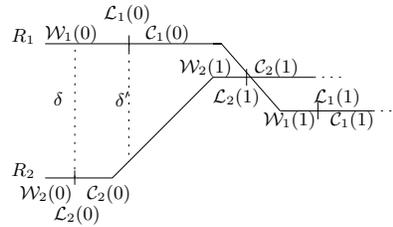

Figure 9: R_1 and R_2 look each other during movement

Theorem 14. *It is impossible for two robots with the same speed to gather in ASYNC model with an adaptive adversary.*

Proof. The adaptive adversary can create a situation such that at each look of a robot, it looks at the other robot in move state. Suppose, R_1 is looking and R_2 is in move state, based on R_1's position R_2 decides to move to some position. After R_2 starts moving, R_1 looks. And this cycle continues perpetually.

Suppose, the adversary has chosen $\mathcal{W}_1(0) > \mathcal{W}_2(0)$. As shown in Fig. 9, R_2 looks first at $\mathcal{L}_2(0)$, when R_1 is idle. So, R_2 decides the value of λ_2 at $\mathcal{L}_2(0)$. Since the adversary is adaptive, it has knowledge of λ_2. Now, the adversary decides the value of $\mathcal{C}_2(0)$ and $\mathcal{W}_2(1)$ such that

$$\mathcal{W}_2(0) + \mathcal{C}_2(0) < \mathcal{W}_1(0)$$

Note that, adversary can choose $\mathcal{W}_2(1)$ to be any non-negative real number. Next, R_1 looks at $\mathcal{L}_1(0)$ and R_2 is in move state. Since, the adversary already knows the time at which R_2 looks next time, i.e., $\mathcal{L}_2(1)$, it can choose $\mathcal{C}_1(0)$ such that

$$\mathcal{W}_2(0) + \mathcal{C}_2(0) + \lambda_2\delta + \mathcal{W}_2(1) > \mathcal{W}_1(0) + \mathcal{C}_1(0) \text{ and}$$
$$\mathcal{W}_2(0) + \mathcal{C}_2(0) + \lambda_2\delta + \mathcal{W}_2(1) < \mathcal{W}_1(0) + \mathcal{C}_1(0) + \lambda_1\delta'$$

where, δ' is the distance of R_2 from R_1 at $\mathcal{L}_2(0)$. Continuing in this manner, the adversary can guarantee that at each look apart from $\mathcal{L}_2(0)$ each robot finds the other in move state.

Also, as a special case, if λ chosen by a robot is 0, then the adversary can choose the next computation delay and wait time to be zero. Then the robot has to choose λ value again. Hence, it is impossible for two robots to gather with an adaptive adversary. $\qquad\square$

5 Gathering multiple robots with merging

We define merging as once multiple robots are collocated, they will behave as a single entity thenceforth. With merging, we can reduce the multi-robot gathering problem to a two robot gathering problem for which the solution has been explored in the previous sections. First, we describe Algorithm `ReduceToLine`, which reduces the gathering problem in $\mathbb{R}^2$ to gathering on a line. Then we describe the reduction to a scenario where the robots are located at two distinct positions.

5.1 Reduce to Line

The robots are on a Euclidean plane. Determine the farthest pair of robots. If there are multiple pair of robots with the maximum distance, each activated robot moves $\lambda/100$ times the distance between them linearly outwards along the line joining them. Here 100 is chosen as an arbitrary constant. This movement makes one pair of robots as the unique maximum distance pair. The adversary can activate all the robots in the farthest pairs at the same time so that there can be multiple pairs of robots with the maximum distance in the next cycle. Specifically, if there are k pairs of robots with the same distance and two different values of λ, then it would take $O(\log_2(k))$ rounds to arrive at a unique pair of robots which are the farthest distance from each other. The line joining the maximum distance pair of robots is unique. Once the unique line is determined, each robot determines its projection on the line and moves towards it. Note that, moving towards the projection cannot make another pair of robots as the maximum distance pair, as the projected distance of a pair of robots is smaller compared to their original distance. Once the process is complete, the robots will form a line.

5.2 With Merging

If two robots merge and behave as a single robot, then we can deterministically achieve a two robot gathering scenario. The inner robots on the line do not move. Observe that, an intermediate robot can see other robots in two opposite directions. The outermost robots move towards the nearest inner robot, and this continues recursively. A robot can determine that it is the outermost robot if it sees other robots only in one direction. It is possible to achieve gathering directly from a situation with three distinct robot positions, say A, B, and C. It can happen if and only if the activation time of the robot at the middle position B does not lie between the arrival time of A at B and C at B. Otherwise, it becomes a two robot gathering problem.

5.3 Discussion for gathering without merging

If merging is not allowed, then the collocated robots behave as individual entities. The process of forming a configuration with two distinct robot locations from a multi-robot configuration on the line is similar to the previous subsection. The difficulty here lies in the fact that the configuration with two distinct robot locations can diverge further into a configuration with multiple robot locations. Once a subset of robots at some position are activated by the adversary, they start moving according to the configuration with two robot locations. While the robots are in movement, if other robots are activated subsequently, then the configuration becomes a configuration with multiple robot location for the robots activated later, and it will cause the robots activated latter to take decision based on a configuration with multiple robot location.

6 Conclusion

This paper sheds light on the gathering of two robots with random bits. The adversary is characterized into two types, oblivious and adaptive. In case of an oblivious adversary, we have proved the possibility of gathering in the $ASYNC_{IC}$ model for a known speed ratio of the two robots. With a lower bound on adversarial control, we also provide an estimate for the expected number of looks. The approach of gathering has been generalized to include multiple robots with merging. It remains to be investigated whether it is possible to gather in the general $ASYNC$ model with no restrictions on an oblivious adversary. We also show that an adaptive adversary can always prevent the gathering of the robots in $ASYNC$.

As future work, further characterization of choice of λ values can be considered, and the analysis can be improved. One can also explore potential adversarial behaviors in the gathering of multiple robots without merging.

References

[1] Noa Agmon and David Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. *SIAM J. Comput.*, 36(1):56–82, 2006.

[2] Cédric Auger, Zohir Bouzid, Pierre Courtieu, Sébastien Tixeuil, and Xavier Urbain. Certified impossibility results for byzantine-tolerant mobile robots. In *Stabilization, Safety, and Security of Distributed Systems - 15th Intl. Symposium, SSS 2013, Osaka, Japan, November 13-16, 2013. Proceedings*, pages 178–190, 2013.

[3] Evangelos Bampas, Lélia Blin, Jurek Czyzowicz, David Ilcinkas, Arnaud Labourel, Maria Potop-Butucaru, and Sébastien Tixeuil. On asynchronous rendezvous in general graphs. *Theor. Comput. Sci.*, 753:80–90, 2019.

[4] S. Bhagat, S. Gan Chaudhuri, and K. Mukhopadhyaya. Fault-tolerant gathering of asynchronous oblivious mobile robots under one-axis agreement. *J. Discrete Algorithms*, 36:50 – 62, 2016.

[5] Subhash Bhagat and Krishnendu Mukhopadhyaya. Fault-tolerant gathering of semi-synchronous robots. In *Proceedings of the 18th International Conference on Distributed Computing and Networking, Hyderabad, India, January 5-7, 2017*, page 6, 2017.

[6] Zohir Bouzid, Shantanu Das, and Sébastien Tixeuil. Gathering of mobile robots tolerating multiple crash faults. In *IEEE 33rd Intl. Conference on Distributed Computing Systems, ICDCS 2013, 8-11 July, 2013, Philadelphia, Pennsylvania, USA*, pages 337–346, 2013.

[7] Quentin Bramas and Sébastien Tixeuil. Wait-free gathering without chirality. In *Structural Information and Communication Complexity - 22nd International Colloquium, SIROCCO 2015, Montserrat, Spain, July 14-16, 2015, Post-Proceedings*, pages 313–327, 2015.

[8] Quentin Bramas and Sébastien Tixeuil. Brief announcement: Probabilistic asynchronous arbitrary pattern formation. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing, PODC 2016, Chicago, IL, USA, July 25-28, 2016*, pages 443–445, 2016.

[9] Quentin Bramas and Sébastien Tixeuil. The random bit complexity of mobile robots scattering. *Int. J. Found. Comput. Sci.*, 28(2):111–134, 2017.

[10] Davide Canepa, Xavier Défago, Taisuke Izumi, and Maria Potop-Butucaru. Flocking with oblivious robots. In *Stabilization, Safety, and Security of Distributed Systems - 18th International Symposium, SSS 2016, Lyon, France, November 7-10, 2016, Proceedings*, pages 94–108, 2016.

[11] Sruti Gan Chaudhuri. Flocking along line by autonomous oblivious mobile robots. In *Proceedings of the 20th International Conference on Distributed Computing and Networking, ICDCN 2019, Bangalore, India, January 04-07, 2019*, pages 460–464, 2019.

[12] Sruti Gan Chaudhuri and Krishnendu Mukhopadhyaya. Gathering asynchronous transparent fat robots. In *Distributed Computing and Internet Technology, 6th International Conference, ICDCIT 2010, Bhubaneswar, India, February 15-17, 2010. Proceedings*, pages 170–175, 2010.

[13] Mark Cieliebak and Giuseppe Prencipe. Gathering autonomous mobile robots. In *SIROCCO 9, Proceedings of the 9th Intl. Colloquium on Structural Information and Communication Complexity, Andros, Greece, June 10-12, 2002*, pages 57–72, 2002.

[14] Shantanu Das, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Masafumi Yamashita. Autonomous mobile robots with lights. *Theor. Comput. Sci.*, 609:171–184, 2016.

[15] Shantanu Das, Paola Flocchini, Nicola Santoro, and Masafumi Yamashita. Forming sequences of geometric patterns with oblivious mobile robots. *Distributed Computing*, 28(2):131–145, 2015.

[16] Xavier Défago, Maria Gradinariu Potop-Butucaru, Julien Clément, Stéphane Messika, Philippe Raipin Parvédy, and Philippe Raipin Parvédy. Fault and byzantine tolerant self-stabilizing mobile robots gathering - feasibility study -. *CoRR*, abs/1602.05546, 2016.

[17] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Hard tasks for weak robots: The role of common knowledge in pattern formation by autonomous mobile robots. In *Proc. ISAAC*, pages 93–102, 1999.

[18] Nao Fujinaga, Hirotaka Ono, Shuji Kijima, and Masafumi Yamashita. Pattern formation through optimum matching by oblivious CORDA robots. In *Principles of Distributed Systems - 14th International Conference, OPODIS 2010, Tozeur, Tunisia, December 14-17, 2010. Proceedings*, pages 1–15, 2010.

[19] Adam Heriban, Xavier Défago, and Sébastien Tixeuil. Optimally gathering two robots. In *Proceedings of the 19th International Conference on Distributed Computing and Networking, ICDCN 2018, Varanasi, India, January 4-7, 2018*, pages 3:1–3:10, 2018.

[20] Taisuke Izumi, Tomoko Izumi, Sayaka Kamei, and Fukuhito Ooshita. Feasibility of polynomial-time randomized gathering for oblivious mobile robots. *IEEE Trans. Parallel Distrib. Syst.*, 24(4):716–723, 2013.

[21] Taisuke Izumi, Daichi Kaino, Maria Gradinariu Potop-Butucaru, and Sébastien Tixeuil. On time complexity for connectivity-preserving scattering of mobile robots. *Theor. Comput. Sci.*, 738:42–52, 2018.

[22] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[23] Takashi Okumura, Koichi Wada, and Xavier Défago. Optimal rendezvous L-algorithms for asynchronous mobile robots with external-lights. In *22nd International Conference on Principles of Distributed Systems, OPODIS 2018, December 17-19, 2018, Hong Kong, China*, pages 24:1–24:16, 2018.

[24] Debasish Pattanayak, Kaushik Mondal, H. Ramesh, and Partha Sarathi Mandal. Gathering of mobile robots with weak multiplicity detection in presence of crash-faults. *J. Parallel Distrib. Comput.*, 123:145–155, 2019.

[25] Kazuo Sugihara and Ichiro Suzuki. Distributed algorithms for formation of geometric patterns with many mobile robots. *J. Field Robotics*, 13(3):127–139, 1996.

[26] Ichiro Suzuki and Masafumi Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM J. Comput.*, 28(4):1347–1363, 1999.

[27] Abraham Wald. On cumulative sums of random variables. *Ann. Math. Statist.*, 15(3):283–296, 09 1944.