

The agreement power of disagreement

Quentin Bramas, Anissa Lamani, Sébastien Tixeuil

▶ **To cite this version:**

Quentin Bramas, Anissa Lamani, Sébastien Tixeuil. The agreement power of disagreement. 2021. hal-03381747

HAL Id: hal-03381747
https://hal.science/hal-03381747

Preprint submitted on 17 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

The agreement power of disagreement*

Quentin Bramas[1], Anissa Lamani[1], and Sébastien Tixeuil[2]

[1] ICUBE, Strasbourg University, CNRS, France
[2] Sorbonne University, CNRS, LIP6, France

Abstract. We consider the rendezvous problem of two autonomous robots with very weak capacities. This problem is notoriously impossible to solve in the semi-synchronous execution model when robots are deterministic, oblivious, and their ego-centered coordinate system is fully symmetric.
We show that if the robots disagree on the unit distance of their coordinate system, it becomes possible to solve rendezvous and agree on a final common location, without additional assumptions.

1 Introduction

We consider swarms of mobile robots that must coordinate to solve a given task. More precisely, we consider robots modeled by dimensionless points that evolve in a Euclidean bidimensional space according to the Look-Compute-Move (LCM) model introduced by Suzuki and Yamashita [11]. In the LCM model, robots repeatedly execute cycles of Look-Compute-Move phases. In the Look phase, the robot obtains an ego-centered view of the position of the other robots (in its own coordinate system). In the Compute phase, the robot decides where it should move next (still in its own coordinate system). Finally, in the Move phase, the robot simply moves toward its destination.

The vast majority of the research effort in the LCM model [5] focuses on understanding the exact hypotheses that make a task solvable. In most cases, those hypotheses are tightly coupled with the amount of synchronization between robots. Three main synchronization models have been considered: the fully synchronous (FSYNC) model mandates *all* robots to execute their LCM cycles simultaneously, the semi-synchronous (SSYNC) model allows that only a non-empty subset of robots executes its LCM cycle simultaneously, while the asynchronous (ASYNC) model makes no hypothesis about the relative speed of each robot or each phase.

A benchmarking problem in this context is that of *rendezvous*, where two robots have to meet in finite time at the exact same location, not known beforehand. Despite its apparent simplicity, this problem triggered interest from the research community as in FSYNC, it is solvable [11], while in SSYNC, it is unsolvable [11, 3] deterministically, without additional assumptions. One of the key reasons for impossibility is that the two robots may have initial symmetric views (and hence make symmetric moves when operated synchronously), but when only one robot is scheduled for execution (as is possible in SSYNC), only half of the symmetric algorithm is performed at any time, preventing the robots from actually meeting (they only converge toward one another).

Related Works To circumvent the aforementioned impossibility result, several options for breaking the initial symmetry were considered.

One line of work considers adding extra capacities to the robots. The seminal paper by Suzuki and Yamashita [11] provides a probabilistic solution to the problem (and each robot makes an constant expected number of coin tosses). The rest of the literature focused on deterministic solutions. Another series of papers considers robot that are endowed with some variant of persistent memory. In more details, it was proposed to endow each robot with a *light* [4], that is, a robot is capable of emitting one color among a fixed number of available colors, visible to all other robots. This additional capacity allows to solve rendezvous in the most general ASYNC model, provided

* This work was partially funded by the ANR project SAPPORO, ref. 2019-CE25-0005-1.

that lights of robots are capable to emit at least *four* colors [4]. In the SSYNC model, Viglietta [12] proved that being able to emit two colors is sufficient to solve rendezvous. In the same paper [12], Viglietta proves [12] that three colors are sufficient in ASYNC. Both solutions in ASYNC [4, 12] and SSYNC [12] output a correct behavior independently of the initial value of the lights' colors. Then, Okumura *et al.* [9] presented a rendezvous algorithm with two colors in ASYNC assuming *rigid* moves (that is, the move of every robot is never stopped by the scheduler before completion), or assuming non-rigid moves but robots being aware of δ (the minimum distance before which the scheduler cannot interrupt their move). Also, the solution of Okumura *et al.* [9] requires lights to have a specific color in the initial configuration. Finally, Heriban *et al.* [7] prove that two colors are necessary and sufficient in ASYNC without extra assumptions.

Another line of work, most related to the current paper, considers restricting the amount of symmetry that can occur in (an persist from) the initial configuration. One set of papers relates to the directions and orientations of the coordinate systems that are given to each robot by the adversary, to prevent a symmetric situation to occur. This is abstracted by the notion of *compass*, that supposedly points to the "North" of the local coordinate system of each robot. When compasses are perfect (i.e., the two robots have the same "North"), SSYNC rendezvous can be achieved [6]. Also, even if compasses are only eventually perfect (there is a time t, unknown to the robots, after which compasses are perfect), SSYNC rendezvous is also feasible [10]. Finally, a complete SSYNC characterization of rendezvous solvability with respect to compasses is due to Izumi et al. [8]: *(i)* if compasses are fixed (they do not change throughout the execution), rendezvous is solvable if and only if the two compasses angle difference ϕ is smaller than $\frac{\pi}{2}$, and *(ii)* if compasses are dynamic (their direction may vary between two LCM cycles), rendezvous is solvable if and only if ϕ is smaller than $\frac{\pi}{4}$. The case of ASYNC is only partially solved: while the results for static compasses extend to ASYNC, rendezvous with dynamic compasses is feasible if $\phi < \frac{\pi}{6}$ and impossible if $\phi \geq \frac{\pi}{4}$, but the interval $\left[\frac{\pi}{6}, \frac{\pi}{4}\right)$ remains unknown.

Finally, Bramas et al. [1] showed recently that, when robots agree only on the North direction but not on the East (*i.e.*, they might not have the same chirality), then the rendezvous is solvable, which might seems surprising because robots can start with symmetric views (and keep their views symmetric until the rendezvous is achieved).

Our contribution We observe that the coordinate system that is given to each robot also includes a *unit distance* (that may be different for each robot as it is given by an adversary). Yet, to our knowledge, this unit distance was never considered as a tool to break initial symmetry, but rather as an assumption (all unit distances are equal) when designing impossibility results.

In this paper, we investigate the possibility to include the two robots' unit distance in the analysis of rendezvous solvability in SSYNC. In more details, we consider two robots that are arbitrarily disoriented (so, the angle difference ϕ may be equal to $\frac{\pi}{2}$), but have a different unit distance. Then, ρ is the ratio between the largest and the smallest unit distance of the robots. In this setting, we show that for any two real numbers ρ_{min} and ρ_{max}, known to the robots, such that $1 < \rho_{min} < \rho_{max}$, if $\rho \in [\rho_{min}, \rho_{max}]$, then rendezvous is solvable in SSYNC, without any additional assumption (robots are deterministic, oblivious, and their compasses are arbitrary). The extreme case $\rho = 1$ (both unit distances are equal) is known to render the problem unsolvable.

The rest of the paper is organized as follows. Section 2 describes the execution model. To warm up, Section 3 considers the simple case when $\rho = 2$. Then, the general solution is described in section 4. Concluding remarks are provided in Section 5

2 Preliminaries

We consider two robots, evolving in a Euclidean two dimensional space. Robots are modeled as points and are assumed to be uniform (they execute the same algorithm), and oblivious (they cannot remember past actions).

Let Z be a global coordinate system. A *configuration* at time t, denoted C_t, is a set $\{r_1, r_2\}$ containing the positions of both robots in Z at time t. Notice that r_i, $i = 1, 2$, denotes at the same

time a robot and its position in $\mathbb{R}^2$ in the coordinate system Z. Robots do not know Z, instead, each robot r_i has its own coordinate system Z_{r_i} centered at the current position of r_i. We assume *disoriented* robots (they do not agree on any axis) that have different unit distance. Let ρ be the ratio between the largest and the smallest unit distance of the robots *i.e.*, $unit_2 = \rho \cdot unit_1$, with $unit_2 > unit_1$. For a robot r, d_r denotes the distance between the two robots in its own coordinate system. Thus, if r and r' are the two robots, we have $d_r = \rho d_{r'}$ or $d_r = \frac{d_{r'}}{\rho}$. For simplicity, in the remaining of the paper, r_1 denotes the robot with the largest unit distance and r_2 the other robot *i.e.*, $d_1 < d_2$ (where we abusively write d_i instead of d_{r_i}, for $i = 1, 2$). Of course, a robot is *not* aware of it being the robot with the largest unit distance.

Robots operate in cycles that comprise three phases: Look, Compute and Move. More precisely, at each time instant, an activated robot first takes a snapshot to see the position of the other robot in its ego-centered coordinate system. Based on this snapshot, the robot either computes a destination or decides to remain idle. Finally, the robot moves towards the computed destination (if any) following a straight path. We assume *non-rigid* movements *i.e.* a robot can be stopped anywhere along the path to its destination after traveling at least a fixed positive distance δ. The value of δ is common to the two robots but it is unknown. Its value can be arbitrarily small but it is fixed and never changes.

In configuration C, the *local view* of a robot r_i, denoted $\mathcal{V}_{r_i}$ is the output of the look phase. More precisely, when a robot r_i takes a snapshot, it observes the position of the other robot in its own coordinate system Z_{r_i} (translated by $-r_i$ so that r_i is always at the center). An algorithm A is a function mapping local views to destinations. When r_i is activated at time t, algorithm A outputs r_i's destination p in its local coordinate system Z_{r_i} based on $\mathcal{V}_{r_i}$.

We consider the SSYNC model where at each time instant, a non-empty subset of robots is activated by an external entity called scheduler. The activated robots execute their Look-Compute-Move cycle synchronously. We assume that the scheduler is fair *i.e.* each robot is activated infinitely often. An execution $\mathcal{E} = (C_0, C_1, \dots)$ of an algorithm A is a sequence of configurations, where C_0 is an initial configuration, and every configuration C_{t+1} is obtained from C_t by applying A to the robots scheduled for execution by the scheduler.

3 An algorithm when $\rho = 2$

As an introduction, we show a simple algorithm solving the problem when $\rho = 2$. In this case, the *level* $l_i \in \mathbb{Z}$ of a robot r_i is the unique integer such that $d_i \in [2^{-l_i}, 2^{-l_i+1})$. By construction, we know that $l_2 = l_1 - 1$, because $d_2 = 2d_1$. Then, Algorithm 1 solves the rendezvous with $\rho = 2$. Indeed, by construction, only one robot remains idle and one robot moves to the other.

Algorithm 1: Rendezvous with $\rho = 2$, executed by robot r.

if $l_r \equiv 0 \mod 2$ **then** Remain idle
else Move to the other robot.

Visually:

case $l_r \equiv 0 \mod 2$



case $l_r \equiv 1 \mod 2$



Theorem 1. *Algorithm 1 solves the rendezvous problem in SSYNC when $\rho = 2$.*

Proof. As $\rho = 2$, at each time instant, $|l_1 - l_2| = 1$ where for any $i \in \{1, 2\}$, $l_i \in \mathbb{Z}$ is the level of robot r_i. That is, by Algorithm 1, at each time instant, a single robot is allowed to move. Its

destination is the other robot's position. Let d_t be the distance between the two robots at time t. Two cases are possible:

1. $d_t \leq \delta$. As the scheduler is assumed to be fair, the robot allowed to move is eventually activated. When it moves, as the other robot remains idle by Algorithm 1, the rendezvous is achieved.
2. $d_t > \delta$. First observe that the distance between the two robots never increases. Indeed, at each time instant, a robot either moves towards the other robot along the straight line connecting them or remains idle. As the scheduler is assumed to be fair, the robot allowed to move is eventually activated. When it moves, the distance between the two robots decreases by at least δ. That is, at each time a robot moves, the distance between the two robots decreases by at least δ. Hence, we can deduce that there exists a time $t' > t$ such that $d_{t'} \leq \delta$ and we retrieve Case 1.

From Cases 1 and 2 we can deduce that the rendezvous is eventually achieved. Hence, the theorem holds. □

4 An algorithm when $\rho \in [\rho_{\min}, \rho_{\max}]$

In this section, we assume that the robots know an upper and a lower bound on the value ρ *i.e.*, $\rho \in [\rho_{\min}, \rho_{\max}]$. In this case, the intervals defining the level of a robot are more complex.

We define two infinite sequences of intervals as follows:

$$\forall i \in \mathbb{Z} \qquad S_i = [\rho_{\min}^{-i}\rho_{\max}^{-i}, \rho_{\min}^{-(i-1)}\rho_{\max}^{-i}) \tag{1}$$

$$M_i = [\rho_{\min}^{-i}\rho_{\max}^{-(i+1)}, \rho_{\min}^{-i}\rho_{\max}^{-i}) \tag{2}$$

The sets S_i and M_i are called *levels*. We consider that the levels are ordered by the inverse of their length *i.e.*, for all $i \in \mathbb{Z}$, we say level M_i, resp. S_i, is greater than level $M_{i'}$, resp. $S_{i'}$, when $i > i'$. Moreover, level M_i is greater than level S_i.

First, notice that

$$\bigcup_{i\in\mathbb{Z}} S_i \cup M_i = \mathbb{R}_+^*$$

and the intervals are pairwise disjoints, so the sequences form a partition of $\mathbb{R}_+^*$. Figure 1 illustrates this partition. We can see that when the distance d_r seen by a robot r decreases, its level increases. For simplicity, we say a robot r is in a set X, or *has level* X, if its distance d_r is in X.

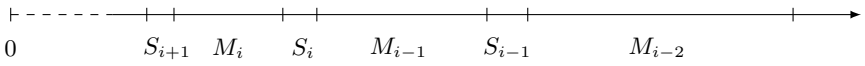


Fig. 1. Partition of the line $\mathbb{R}_+^*$ into levels

We now prove a lemma that states that both robots cannot have a level of type S, and the levels of the robots are not too far away.

Lemma 1. *For every $i \in \mathbb{Z}$, if a robot r has level S_i, then the other robot r' has level M_{i-1} or M_i.*

Proof. We have $d_r \in [\rho_{\min}^{-i}\rho_{\max}^{-i}, \rho_{\min}^{-(i-1)}\rho_{\max}^{-i})$. If $d_r < d_{r'}$, then

$$d_{r'} = \rho d_r \in \left[\rho_{\min} \times \rho_{\min}^{-i}\rho_{\max}^{-i}, \rho_{\max} \times \rho_{\min}^{-(i-1)}\rho_{\max}^{-i}\right) = \mathcal{M}_{i-1}$$

If $d_r > d_{r'}$, then

$$d_{r'} = \frac{d_r}{\rho} \in \left[\frac{1}{\rho_{\max}} \times \rho_{\min}^{-i}\rho_{\max}^{-i}, \frac{1}{\rho_{\min}} \times \rho_{\min}^{-(i-1)}\rho_{\max}^{-i}\right) = \mathcal{M}_i$$

□

For simplicity, let

$$\mathcal{M}_0 = \bigcup_{i \equiv 0 \mod 2} M_i$$
$$\mathcal{M}_1 = \bigcup_{i \equiv 1 \mod 2} M_i$$
$$\mathcal{S}_0 = \bigcup_{i \equiv 0 \mod 2} S_i$$
$$\mathcal{S}_1 = \bigcup_{i \equiv 1 \mod 2} S_i$$

Also, consider the indexes of those sets modulo 2 *eg.*, $\mathcal{M}_{-1} = \mathcal{M}_3 = \mathcal{M}_1$.

Let $\mathfrak{s}(d)$ be the smallest value defined as follows:

$$d\frac{1-\rho_{\min}^{-1}}{2\rho_{\min}^2\rho_{\max}^2} \leq \mathfrak{s}(d) \leq d\frac{1-\rho_{\min}^{-1}}{2} \text{ such that } \mathfrak{s}(d) \in \mathcal{S}_0 \tag{3}$$

Lemma 2. *$\mathfrak{s}(d)$ is well defined*

Proof. We have to prove that, for any $d > 0$, we have

$$\left[d\frac{1-\rho_{\min}^{-1}}{2\rho_{\min}^2\rho_{\max}^2}, d\frac{1-\rho_{\min}^{-1}}{2}\right] \cap \mathcal{S}_0 \neq \emptyset$$

Assume, for the sake of contradiction, that the intersection is empty for a given $d > 0$. Let a be the smallest number in $\mathcal{S}_0$ such that

$$d\frac{1-\rho_{\min}^{-1}}{2} < a \tag{4}$$

a is well defined because each interval S_i is closed to the left. Then $a \in S_i$ for some $i \in \mathbb{Z}$, $i \equiv 0 \mod 2$, and it is clear that $a = \rho_{\min}^{-i}\rho_{\max}^{-i}$ (*i.e.*, a is the lower bound of the interval S_i). Since by assumption

$$S_{i+2} \cap \left[d\frac{1-\rho_{\min}^{-1}}{2\rho_{\min}^2\rho_{\max}^2}, d\frac{1-\rho_{\min}^{-1}}{2}\right) = \emptyset$$

and by the minimality of a, we have

$$\rho_{\min}^{-(i+2)}\rho_{\max}^{-(i+2)} \leq d\frac{1-\rho_{\min}^{-1}}{2\rho_{\min}^2\rho_{\max}^2}$$
$$\Rightarrow \qquad a = \rho_{\min}^{-i}\rho_{\max}^{-i} \leq d\frac{1-\rho_{\min}^{-1}}{2}$$

The last inequality contradicts Equation (4). □

We say a robot r executes `Move`($\mathfrak{s}$) if it moves a distance $\mathfrak{s}(d_r)$ towards the other robot. The first inequality ensures that if both robots execute `Move`($\mathfrak{s}$), then one of them eventually reaches the next level. The second part of the definition ensures that if one robot executes `Move`($\mathfrak{s}$) and the other executes `Move`($Other$), the one that executes `Move`($\mathfrak{s}$) is now in $\mathcal{S}_0$.

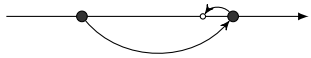
Algorithm 2: The movement of a robot r depends on the distance between the two robots d (seen by robot r), and on where the robot r sees itself in the line (on the right or on the left)

$d \in \mathcal{S}_0 \cup \mathcal{S}_1$



$d \in \mathcal{M}_0$

The right robot moves a distance $\mathfrak{s}(d)$



$d \in M_1$

The left robot moves a distance $\mathfrak{s}(d)$



Our algorithm is defined in Figure 2. First we notice that, by Lemma 1, robots cannot both stay stationary. Indeed, if r_1 is in S_i, then $d_2 \in M_{i-1}$ and if r_2 is in S_i, then $r_1 \in M_i$ (recall r_1 and r_2 denotes the two robots such that $d_1 < d_2$).

Let $\texttt{Conf}(X, Y)$ be the set of configurations where $r_1 \in X$ and $r_2 \in Y$. Recall that r_1 and r_2 are such that the distance d_1 is smaller than d_2, so r_1 has a greater level than (or the same as) r_2.

We directly have the following lemma to reduce the number of cases we handle in the sequel.

Lemma 3. *For every $i, j \in \{0, 1\}$,*

$$\mathit{Conf}(\mathcal{S}_i, \mathcal{M}_i) = \mathit{Conf}(\mathcal{M}_{i+1}, \mathcal{S}_i) = \mathit{Conf}(\mathcal{S}_i, \mathcal{S}_j) = \emptyset$$

Proof. Since r_1 has a greater level compared to r_2 so, for every $i \in \mathbb{Z}$, if $r_1 \in S_i$, then $r_2 \in M_{i-1}$, by Lemma 1, hence $\texttt{Conf}(\mathcal{S}_i, \mathcal{M}_i) = \emptyset$. Similarly, if $r_2 \in S_i$, then $r_1 \in M_i$ (because M_i is the level right above S_i). Finally, we saw that, by construction, both robots cannot be in an $\mathcal{S}$ level. □

This means $\texttt{Conf}(\mathcal{M}_0, \mathcal{S}_0)$, $\texttt{Conf}(\mathcal{M}_0, \mathcal{M}_0)$, $\texttt{Conf}(\mathcal{S}_1, \mathcal{M}_0)$, $\texttt{Conf}(\mathcal{M}_1, \mathcal{M}_0)$, $\texttt{Conf}(\mathcal{M}_1, \mathcal{S}_1)$, and $\texttt{Conf}(\mathcal{M}_1, \mathcal{M}_1)$ are the only non-empty set of configurations.

From flexible to rigid movements. The following lemma shows that, the robots eventually are, and remain, at distance at most δ from each other. Using this lemma, we can now assume in the remaining of the paper that movements are rigid.

Lemma 4. *If the distance d between the two robots is greater than δ (in the global coordinate system), then, after two rounds, the distance between robots decreases by at least a constant C (that depends only on δ, $\rho_{\min}$ and $\rho_{\max}$).*

Proof. First, it is clear that robots cannot increase the distance between them.

If one or two robots execute $\texttt{Move}(\mathfrak{s})$, then the distance between the robots decreases by at least

$$\min\left(\delta, d\frac{1-\rho_{\min}^{-1}}{2\rho_{\min}^2\rho_{\max}^2}\right) \geq C \qquad \text{with } C = \delta\frac{1-\rho_{\min}^{-1}}{2\rho_{\min}^2\rho_{\max}^2}$$

If one robot remains idle and the other robot execute $\texttt{Move}(\mathfrak{s})$, then the distance decreases by at least δ.

The last remaining case is when both robots are in $\mathcal{M}$ and execute $\texttt{Move}(\mathit{Other})$ at time t. It is possible that the distance does not decrease at all (if both robots reach their destination) or the distance decreases by an arbitrarily small amount. In the next round, at time $t+1$, either (a) the robots are in the same level as before, (b) the level of only one robot increases, or (c) the level of both robots increases.

In case (a), the positions of the robots at time $t+1$ are exchanged, so they now both execute $\texttt{Move}(\mathfrak{s})$ and the property of the lemma is obtained after one more round.

In case (b), one robot reaches $\mathcal{S}$ while the other remains in $\mathcal{M}$, at time $t+1$, so the property is obtained after one more round as well (since one robot remains idle).

In case (c), since both robots cannot reach $\mathcal{S}$ (by Lemma 1) so one robot must increases from M_i to $M_{i'}$ with $i < i'$. we observe that, to do so, the distance must decrease by at least $\frac{d}{\rho_{\min}} \geq \frac{\delta}{\rho_{\min}}$, which is greater than C defined above, and the Lemma is proved. □

When a single robot is activated. First, we compute the configurations that are eventually reached when only one robot is activated.

Lemma 5. *If a single robot is activated and executes* $\texttt{Move}(Other)$*, then the robots gather in one round.*

Lemma 6. *From* $\texttt{Conf}(\mathcal{M}_i, \mathcal{M}_i)$*, if a single robot is activated and executes* $\texttt{Move}(\mathfrak{s})$*, then eventually we reach either a configuration in* $\texttt{Conf}(\mathcal{S}_{i+1}, \mathcal{M}_i)$*.*

Proof. Consider $C \in \texttt{Conf}(M_i, M_i)$, with $i \in \mathbb{Z}$, and d_1 and d_2 the distances seen by r_1 and r_2 respectively in C. We have $d_1 \geq \rho_{\min}^{-i}\rho_{\max}^{-(i+1)}$. After a single robot, say r_1, executes $\texttt{Move}(\mathfrak{s})$, let d_1' and d_2' the distances seen by r_1 and r_2 respectively. So:

$$d_1' = d_1 - \mathfrak{s}(d_1) \geq d_1 - d_1\frac{1-\rho_{\min}^{-1}}{2} \geq d_1\rho_{\min}^{-1} \geq \rho_{\min}^{-(i+1)}\rho_{\max}^{-(i+1)}$$

Where the first inequality comes from the definition of $\mathfrak{s}(d)$, in Equation 3. Hence, $d_1' \notin M_{i+1}$. Similarly, $d_2' \notin M_{i+1}$. The same is true if only r_2 executes $\texttt{Move}(\mathfrak{s})$.

This implies that, if a single robot comes closer by executing $\texttt{Move}(\mathfrak{s})$, the level of the robots can increase by at most one, so we reach a configuration in $\texttt{Conf}(M_i \cup S_{i+1}, \mathcal{M}_i \cup S_{i+1})$.

Then, observe that both robots cannot increase simultaneously their level because $\texttt{Conf}(S_{i+1}, S_i) = \emptyset$, by Lemma 3. Also, observe that r_2 cannot increase its level alone because $\texttt{Conf}(M_{i+1}, S_{i+1}) = \emptyset$. Hence, eventually r_1 enters level $\mathcal{S}_{i+1}$ and we reach configuration $\texttt{Conf}(\mathcal{S}_{i+1}, \mathcal{M}_i)$. □

Lemma 7. *From* $\texttt{Conf}(\mathcal{M}_i, \mathcal{M}_j)$*, with* $i \neq j$*, if a single robot is activated and executes* $\texttt{Move}(\mathfrak{s})$*, then eventually we reach either a configuration in* $\texttt{Conf}(\mathcal{M}_i, \mathcal{S}_i)$*.*

Proof. We know that $C \in \texttt{Conf}(M_i, M_{i-1})$, for some $i \in \mathbb{Z}$ (because the level of r_2 is smaller than the one of r_1). Similarly to the previous lemma, a single robot increases its level and it cannot be r_1 because $\texttt{Conf}(S_{i+1}, M_{i-1}) = \emptyset$. Hence, eventually r_2 enters level S_i and we reach configuration $\texttt{Conf}(M_i, S_i) \subset \texttt{Conf}(\mathcal{M}_i, \mathcal{S}_i)$. □

When both robots are activated. The nine following Lemmas consider all the possible cases, when both robots are activated, depending on the level of each robot. Lemma 8-12 consider the cases where both robots are in $\mathcal{M}_*$, depending on which move the robots are executing (both $\texttt{Move}(\mathfrak{s})$ – Lemma 8-9 –, both $\texttt{Move}(Other)$ – Lemma 10 – or only one $\texttt{Move}(s)$ – Lemma 11-12), Lemma 13-15 consider the case where one robot is in $\mathcal{S}_*$ (depending on whether the moving robot executes $\texttt{Move}(Other)$ – Lemma 13 – or $\texttt{Move}(\mathfrak{s})$ – Lemma 14-15) and Lemma 3 proves that the remaining cases cannot occur.

Lemma 8. $\forall i \in \{0, 1\}$*, if* $C \in \texttt{Conf}(\mathcal{M}_i, \mathcal{M}_i)$ *and both robots execute* $\texttt{Move}(\mathfrak{s})$*, then eventually we reach a configuration in* $\texttt{Conf}(\mathcal{S}_{i+1}, \mathcal{M}_i)$*. The same is true if a single robot is activated.*

Proof. Consider $C \in \texttt{Conf}(M_i, M_i)$, with $i \in \mathbb{Z}$, and d_1 and d_2 the distances seen by r_1 and r_2 respectively in C. We have $d_1 \geq \rho_{\min}^{-i}\rho_{\max}^{-(i+1)}$. After both robots execute $\texttt{Move}(\mathfrak{s})$, let d_1' and d_2' the distances seen by r_1 and r_2 respectively. So:

$$d_1' = d_1 - \mathfrak{s}(d_1) - \mathfrak{s}(d_2) \geq d_1 - 2d_1\frac{1-\rho_{\min}^{-1}}{2} = d_1\rho_{\min}^{-1} \geq \rho_{\min}^{-(i+1)}\rho_{\max}^{-(i+1)}$$

Where the first inequality comes from the definition of $\mathfrak{s}(d)$, in Equation 3, and from the fact that $d_1 < d_2$ (the same inequality is true if a single robot is activated). Hence, $d_1' \notin M_{i+1}$. Similarly, $d_2' \notin M_{i+1}$. But since robots come closer, robots cannot remain in M_i infinitely and eventually one robot reaches S_{i+1} (both cannot reach S_{i+1} simultaneously because $\texttt{Conf}(S_{i+1}, S_{i+1}) = \emptyset$). Since r_1 have a level greater than r_2, eventually we must reach $\texttt{Conf}(S_{i+1}, M_i) \subset \texttt{Conf}(\mathcal{S}_{i+1}, \mathcal{M}_i)$. □

Lemma 9. *$\forall i \in \{0,1\}$, if $C \in \mathtt{Conf}(\mathcal{M}_{i+1}, \mathcal{M}_i)$, and both robots execute $\mathtt{Move}(\mathfrak{s})$, then eventually we reach a configuration in $\mathtt{Conf}(\mathcal{M}_{i+1}, \mathcal{S}_{i+1})$. The same is true if a single robot is activated.*

Proof. Consider $C \in \mathtt{Conf}(M_{i+1}, M_i)$, for some $i \in \mathbb{Z}$. Using the same proof as in the previous Lemma, we obtain that $d'_1 \notin M_{i+2}$ and $d'_2 \notin M_{i+1}$. But since robots come closer, robots cannot remain at the same level infinitely and eventually reaches a level S. Since r_1 have a level greater than r_2, and r_1 cannot reach S_{i+2} while r_2 is still in M_i (Lemma 1), eventually we must reach $\mathtt{Conf}(M_{i+1}, S_{i+1}) \subset \mathtt{Conf}(\mathcal{M}_{i+1}, \mathcal{S}_{i+1})$. □

Lemma 10. *$\forall i, j \in \{0,1\}$, if $C \in \mathtt{Conf}(\mathcal{M}_i, \mathcal{M}_j)$, and both robots execute $\mathtt{Move}(Other)$, then after one round, the configuration is still in $\mathtt{Conf}(\mathcal{M}_i, \mathcal{M}_j)$ and the robots have reversed their position, so they execute $\mathtt{Move}(\mathfrak{s})$ in the next round.*

Proof. If both robots execute $\mathtt{Move}(Other)$, then both robots exchange their position and the distance between them remain the same so that the lemma follows. □

Lemma 11. *$\forall i, j \in \{0,1\}$, if $C \in \mathtt{Conf}(\mathcal{M}_i, \mathcal{M}_j)$ and only r_1 executes $\mathtt{Move}(\mathfrak{s})$ (r_2 execute $\mathtt{Move}(Other)$), then we reach a configuration in $\mathtt{Conf}(\mathcal{S}_0, \mathcal{M}_1)$ (and the robots have reversed their position).*

Proof. If r_2 execute $\mathtt{Move}(Other)$ and r_1 executes $\mathtt{Move}(\mathfrak{s})$, then, by definition of $\mathfrak{s}(d_1)$, robot $r_1 \in \mathcal{S}_0$. Then, using Lemma 3, we know that $r_2 \in M_1$. □

Lemma 12. *$\forall i, j \in \{0,1\}$, if $C \in \mathtt{Conf}(\mathcal{M}_i, \mathcal{M}_j)$ and only r_2 executes $\mathtt{Move}(\mathfrak{s})$ (r_1 execute $\mathtt{Move}(Other)$), then we reach a configuration in $\mathtt{Conf}(\mathcal{M}_0, \mathcal{S}_0)$ (and the robots have reversed their position).*

Proof. If r_1 execute $\mathtt{Move}(Other)$ and r_2 executes $\mathtt{Move}(\mathfrak{s})$, then, by definition of $\mathfrak{s}(d_2)$, robot $r_2 \in \mathcal{S}_0$. Then, using Lemma 3, we know that $r_1 \in M_0$. □

Lemma 13. *$\forall i \in \{0,1\}$, if $C \in \mathtt{Conf}(\mathcal{S}_i, \mathcal{M}_{i-1}) \cup \mathtt{Conf}(\mathcal{M}_i, \mathcal{S}_i)$ and the moving robot executes $\mathtt{Move}(Other)$, then the robots gather in one round.*

Proof. Clearly if one robot remains idle while the other executes $\mathtt{Move}(Other)$, then the robots gather. □

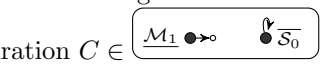
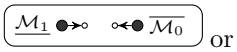
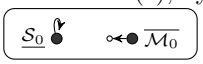
Lemma 14. *$\forall i \in \{0,1\}$, if $C \in \mathtt{Conf}(\mathcal{S}_i, \mathcal{M}_{i-1})$ and r_2 executes $\mathtt{Move}(\mathfrak{s})$, then eventually we reach either a configuration in $\mathtt{Conf}(\mathcal{M}_i, \mathcal{M}_{i-1})$ or a configuration in $\mathtt{Conf}(\mathcal{M}_i, \mathcal{S}_i)$.*

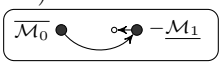
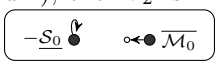
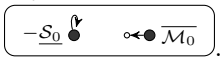
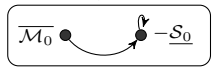
Proof. If r_1 remains idle and r_2 comes closer by executing $\mathtt{Move}(\mathfrak{s})$, then one or both robots eventually increase their levels. By definition of $\mathfrak{s}(d_2)$ (using the same proof as in Lemma 8), r_2 cannot go from level $\mathcal{M}_{i-1}$ to level $\mathcal{M}_i$ directly. Hence, either r_2 enters level $\mathcal{S}_i$ (in this case, r_1 simultaneously enters $\mathcal{M}_i$), and we reach configuration $\mathtt{Conf}(\mathcal{M}_i, \mathcal{S}_i)$, or only r_1 enters $\mathcal{M}_i$ and we reach configuration $\mathtt{Conf}(\mathcal{M}_i, \mathcal{M}_{i-1})$. □

Lemma 15. *$\forall i \in \{0,1\}$, if $C \in \mathtt{Conf}(\mathcal{M}_i, \mathcal{S}_i)$ and r_1 executes $\mathtt{Move}(\mathfrak{s})$, then eventually we reach either a configuration in $\mathtt{Conf}(\mathcal{M}_i, \mathcal{M}_i)$ or a configuration in $\mathtt{Conf}(\mathcal{S}_{i+1}, \mathcal{M}_i)$.*

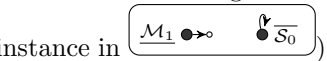
Proof. If r_2 remains idle and r_1 comes closer by executing $\mathtt{Move}(\mathfrak{s})$, then one or both robots eventually increase their levels. By definition of $\mathfrak{s}(d_1)$ (using the same proof as in Lemma 8), r_1 cannot go from level $\mathcal{M}_i$ to level $\mathcal{M}_{i+1}$ directly. Hence, either r_1 enters level $\mathcal{S}_{i+1}$ (in this case, r_2 simultaneously enters $\mathcal{M}_i$), and we reach configuration $\mathtt{Conf}(\mathcal{S}_{i+1}, \mathcal{M}_i)$, or only r_2 enters $\mathcal{M}_i$ and we reach configuration $\mathtt{Conf}(\mathcal{M}_i, \mathcal{M}_i)$. □

Main Proof of Correctness. We can characterize a configuration by only looking at where is located r_1 (right or left), whether r_2 has the same orientation has r_1, and the level of each robot (in $\mathcal{S}_0$, $\mathcal{S}_1$, $\mathcal{M}_0$ or $\mathcal{M}_1$). For simplicity, we use the notation  to denote the configuration where the left robot is in level $X \in \{\mathcal{S}_0, \mathcal{S}_1, \mathcal{M}_0, \mathcal{M}_1\}$ and the right robot has level Y. We add a line over the level of r_1 and a line under the level of r_2. Finally, if both robots have different orientations, we add a minus in front of r_2's level. To help the reader, we also draw the destination of each robot in the configuration.

For instance a configuration $C \in$  is a configuration where r_1 is located on the right, r_2 on the left (they have the same orientation of the line), r_1 is in $\mathcal{S}_0$ and r_2 in $\mathcal{M}_1$. Recall that, since r_1 has a level greater than r_2, if r_1 is in S_i for some $i \in \mathbb{Z}$ ($i \equiv 0 \mod 2$), then r_2 must be in M_{i-1}. Since $C \in \texttt{Conf}(\mathcal{S}_0, \mathcal{M}_1)$ and r_2 executes $\texttt{Move}(\mathfrak{s})$, by Lemma 14, we eventually reach either a configuration in  or in  (the same is true if a single robot is activated, since r_1 remains idle).

As a second example, $C \in$  is a configuration where r_1 is on the left and r_2 sees itself on the left as well. Again, if r_1 is in M_i for some $i \in \mathbb{Z}$ ($i \equiv 0 \mod 2$), then r_2 is in M_{i-1}. If both robots are activated, by Lemma 12 the next configuration is in  . If a single robot is activated, either the robots gather in one round (if r_1 is activated) or we can reach configuration  (if only r_2 is activated).

The information of where r_1 is located (indicated by an over-line) does not impact the movement of the robots, however, it limits the possibilities for the reached configuration. For instance if both robots are in $\mathcal{M}_i$ and make a $\texttt{Move}(\mathfrak{s})$ then we are sure that r_1 is the first to reach $\mathcal{S}_{i+1}$.

Using these notations, we can construct the a graph depicting the transitions between the different sets of configurations. Each arc is proved by one of the previous Lemmas, whose number is indicated on the arc. It is easy to see that if robots agree, resp. do not agree, on the orientation of the line, then the same is true in any reached set of configurations. This implies that we can split the graph in two, one that consider only sets of configuration where the robots agree on the orientation of the line, Figure 2, and one when they do not, Figure 3. The dashed arcs correspond to the transitions that can occur when a single robot is activated. Of course, when a single robot is dictated to move (for instance in ) then activating only this robot results in the same configuration. But with a fair scheduler, the other robot is eventually activated. Also, in this case, activating only the moving robot is similar to activating both robot, so when this happen we only draw the plain arc. Finally, for clarity, we do do represent the dashed arcs corresponding to the case where a single robot is activated and executes $\texttt{Move}$(Other), as robots gather in one round in this case.

Given the previous lemmas that proves the possible transitions between set of configuration, the graphs have been generated by an algorithm (available online [2]). It is easy to check that both graphs are in fact Directed Acyclic Graphs (DAG) with a single sink, the gathered configuration. This means that regardless of the starting configuration, we eventually reach the gathered configuration.

Theorem 2. *When $\rho \in [\rho_{\min}, \rho_{\max}]$, Algorithm 2 solves the rendezvous in SSYNC.*

Proof. Using Lemma 4, we know that eventually robots are and remains at distance at most δ so that we can consider that the movements are rigid. From there, we showed in Figures 2 and Figure 3 that regardless of the configuration and regardless of the orientation of the robots, we eventually reach the gathered configuration. □

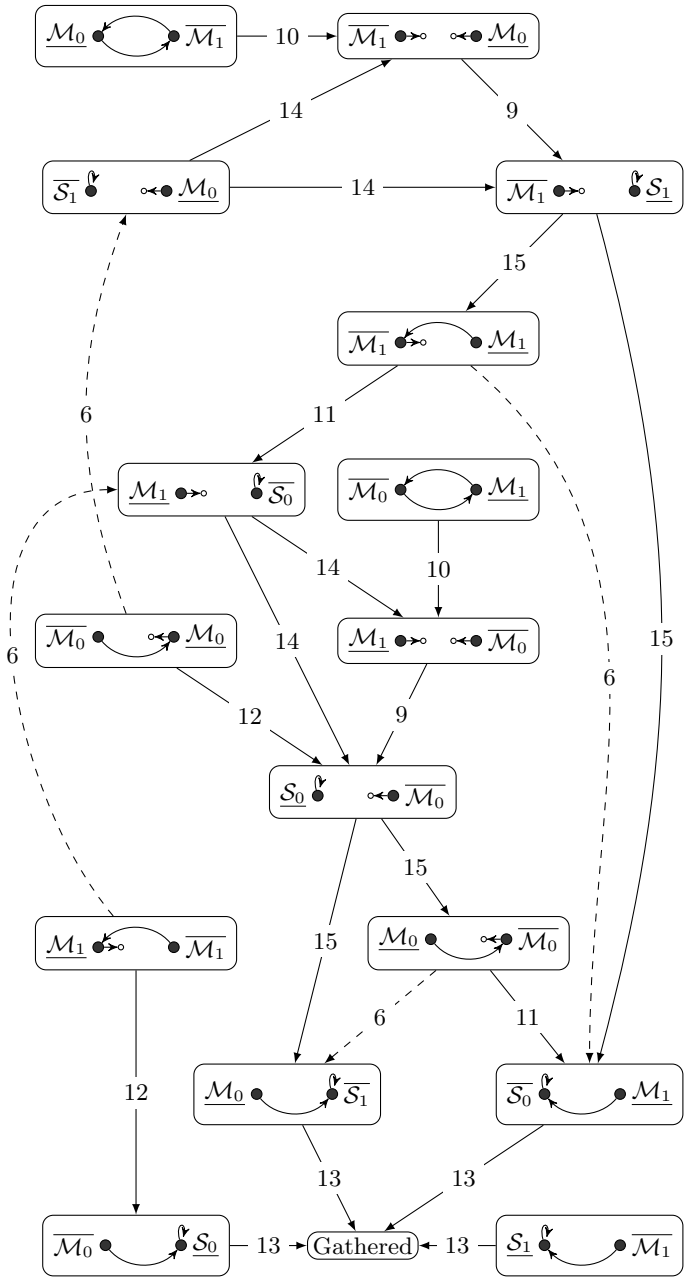


Fig. 2. The DAG of configurations and the transitions between them, when the robots have the same orientation. The number on the edges are the numbers of the Lemmas proving the transition

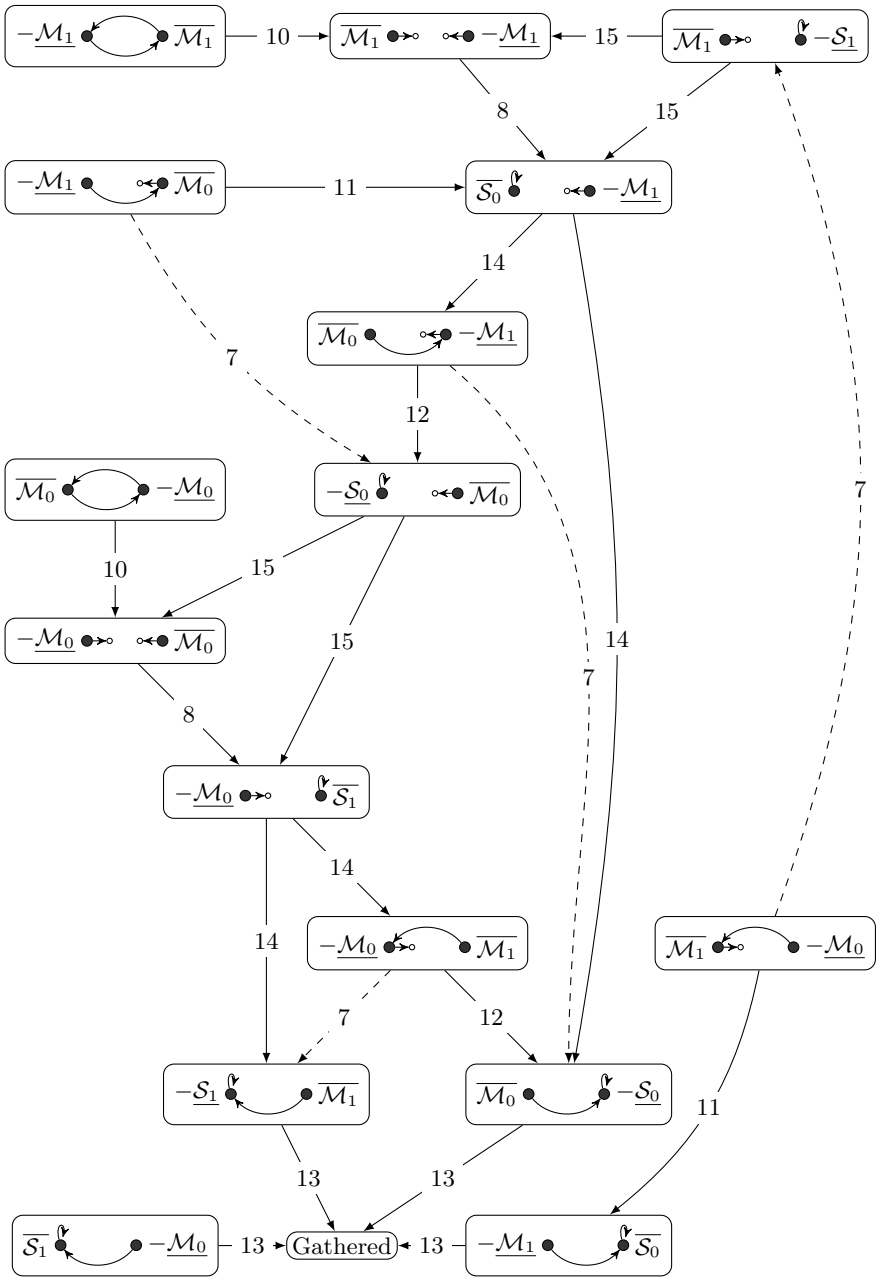


Fig. 3. The DAG of configurations and the transitions between them, when the robots have the opposite orientations. The number on the edges are the numbers of the Lemmas proving the transition.

5 Concluding Remarks

We introduced the possibility to use different unit distances to break symmetries for networks consisting on deterministic oblivious robots that operate in the Look-Compute-Move model. As a case study, we considered the rendezvous problem, that is notoriously impossible to solve in the semi-synchronous execution model, when robots share the same notion of unit distance. By contrast, we proved that when robots have different unit distances (and are unaware of the actual ratio between the unit distances), rendezvous becomes possible in the same model.

A natural open question is to consider the completely asynchronous model (ASYNC). Is it possible to solve rendezvous in ASYNC without any additional capability (no access to a randomness source, nor persistent memory) or constraints (compasses may be fully symmetric) other that the difference in unit distance? Observe that even if $\rho = 2$, the problem seems difficult, as our semi-synchronous algorithm for this special case does not solve rendezvous in ASYNC (one can construct an infinite execution where robots observe one another alternatively as they are moving, and thus never actually reach the other robot destination).

References

1. Quentin Bramas, Anissa Lamani, and Sébastien Tixeuil. Stand up indulgent rendezvous. In *International Symposium on Stabilizing, Safety, and Security of Distributed Systems*, pages 45–59. Springer, 2020.
2. Quentin Bramas, Anissa Lamani, and Sébastien Tixeuil. The agreement power of disagreement: graph generation, September 2021. https://doi.org/10.5281/zenodo.5541136.
3. Pierre Courtieu, Lionel Rieg, Sébastien Tixeuil, and Xavier Urbain. Impossibility of gathering, a certification. *Inf. Process. Lett.*, 115(3):447–452, 2015.
4. Shantanu Das, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Masafumi Yamashita. Autonomous mobile robots with lights. *Theor. Comput. Sci.*, 609:171–184, 2016.
5. Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro (Eds.). *Distributed Computing by Mobile Entities-Current Research in Moving and Computing.* Lecture Notes in Computer Science, 11340. Springer, 2019.
6. Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Gathering of asynchronous robots with limited visibility. *Theor. Comput. Sci.*, 337(1-3):147–168, 2005.
7. Adam Heriban, Xavier Défago, and Sébastien Tixeuil. Optimally gathering two robots. In *Proc. 19th Intl. Conf. on Distributed Computing and Networking, ICDCN*, pages 3:1–3:10, January 2018.
8. Taisuke Izumi, Samia Souissi, Yoshiaki Katayama, Nobuhiro Inuzuka, Xavier Défago, Koichi Wada, and Masafumi Yamashita. The gathering problem for two oblivious robots with unreliable compasses. *SIAM J. Comput.*, 41(1):26–46, 2012.
9. Takashi Okumura, Koichi Wada, and Yoshiaki Katayama. Brief announcement: Optimal asynchronous rendezvous for mobile robots with lights. In *Proc. 19th Intl. Symp. on Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 484–488, November 2017.
10. Samia Souissi, Xavier Défago, and Masafumi Yamashita. Using eventually consistent compasses to gather memory-less mobile robots with limited visibility. *ACM Trans. Auton. Adapt. Syst.*, 4(1):9:1–9:27, 2009.
11. Ichiro Suzuki and Masafumi Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM Journal on Computing*, 28(4):1347–1363, 1999.
12. Giovanni Viglietta. Rendezvous of two robots with visible bits. In *Proc. 9th Intl. Symp. on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics, (ALGOSENSORS)*, pages 291–306, September 2013.