Fault-tolerant and Self-stabilizing Mobile Robots Gathering

— Feasibility Study —

Xavier Défago[1,⋆], Maria Gradinariu[2], Stéphane Messika[3], and Philippe Raipin-Parvédy[2,4]

[1] School of Information Science, JAIST, Ishikawa, Japan
defago@jaist.ac.jp
[2] IRISA/Université de Rennes 1, France
mgradina@irisa.fr
[3] LRI/Université Paris Sud, France
messika@lri.fr
[4] France Telecom R&D, France
philippe.raipin@orange-ft.com

Abstract. Gathering is a fundamental coordination problem in cooperative mobile robotics. In short, given a set of robots with arbitrary initial location and no initial agreement on a global coordinate system, gathering requires that all robots, following their algorithm, reach the exact same but not predetermined location. Gathering is particularly challenging in networks where robots are oblivious (i.e., stateless) and the direct communication is replaced by observations on their respective locations. Interestingly any algorithm that solves gathering with oblivious robots is inherently self-stabilizing.
In this paper, we significantly extend the studies of deterministic gathering feasibility under different assumptions related to synchrony and faults (crash and Byzantine). Unlike prior work, we consider a larger set of scheduling strategies, such as bounded schedulers, and derive interesting lower bounds on these schedulers. In addition, we extend our study to the feasibility of probabilistic gathering in both fault-free and fault-prone environments. To the best of our knowledge our work is the first to address the gathering from a probabilistic point of view.

1 Introduction

Many applications of mobile robotics envision groups of mobile robots self-organizing and cooperating toward the resolution of common objectives. In many cases, the group of robots is aimed at being deployed in adverse environments, such as space, deep sea, or after some natural (or unnatural) disaster. It results that the group must self-organize in the absence of any prior infrastructure (e.g., no global positioning), and ensure coordination in spite of faulty robots and unanticipated changes in the environment.

⋆ Work supported by MEXT Grant-in-Aid for Young Scientists (A) (Nr. 18680007).

The *gathering problem*, also known as the *Rendez-Vous* problem, is a fundamental coordination problem in cooperative mobile robotics. In short, given a set of robots with arbitrary initial location and no initial agreement on a global coordinate system, gathering requires that all robots, following their algorithm, reach the exact same location—one not agreed upon initially—within a *finite* number of steps, and remain there.

Similar to the Consensus problem in conventional distributed systems, gathering has a simple definition but the existence of a solution greatly depends on the synchrony of the systems as well as the nature of the faults that may possibly occur. In this paper, we investigate some of the fundamental limits of deterministic and probabilistic gathering in the face of different synchrony and fault assumptions.

To study the gathering problem, we consider a system model first defined by Suzuki and Yamashita [1], and some variants with various degrees of synchrony. In this model, robots are represented as points that evolve on a plane. At any given time, a robot can be either idle or active. In the latter case, the robot observes the locations of the other robots, computes a target position, and moves toward it. The time when a robot becomes active is governed by an activation daemon (scheduler). In the original definition of Suzuki and Yamashita, called the ATOM model, activations (i.e., look–compute–move) are atomic, and the scheduler is assumed to be fair and distributed, meaning that each robot is activated infinitely often and that any subset of the robots can be active simultaneously. In the CORDA model of Prencipe [2], activations are completely asynchronous, for instance allowing robots to be seen while moving.

Suzuki and Yamashita [1] proposed a gathering algorithm for non-oblivious robots in ATOM model. They also proved that gathering can be solved with three or more oblivious robots, but not with only two.[5] Prencipe [3] studied the problem of gathering in both ATOM and CORDA models. He showed that the problem is impossible without additional assumptions such as being able to detect the multiplicity of a location (i.e., knowing the number of robots that may simultaneously occupy that location). Flocchini *et al.* [4] proposed a gathering solution for oblivious robots with limited visibility in CORDA model, where robots share the knowledge of a common direction as given by some compass. Based on that work, Souissi *et al.* [5] consider a system in which compasses are not necessarily consistent initially. Ando *et al.* [6] propose a gathering algorithm for ATOM model with limited visibility. Cohen and Peleg [7] study the problem when robots' observations and movements are subject to some errors.

None of the previously mentioned works addressed the gathering feasibility in fault-prone environments. One of the first steps in this direction was done by Agmon and Peleg [8]. They prove that gathering of correct robots (referred

[5] With two robots, all configurations are symmetrical and may lead to robots endlessly swapping their positions. In contrast, with three or more robots, an algorithm can be made such that, at each step, either the robots remain symmetrical and they eventually reach the same location, or symmetry is broken and this is used to move one robot at a time.

in this paper *weak gathering*) can be achieved in the ATOM model even in the face of the crash of a single robot. Furthermore, they prove that no deterministic gathering algorithm exists in ATOM model that can tolerate a Byzantine[6] robot. Finally, they consider a stronger daemon, called fully synchronous, in which all robots are always activated simultaneously, and show that weak gathering can be solved in that model when the number of Byzantine robots is less than one third of the system.

Contribution. In this paper, we further study the limits of gathering feasibility in both fault-free and fault prone environments, by considering centralized schedulers[7] (i.e., activations in mutual exclusion) and k-bounded schedulers, that is, schedulers ensuring that between any two consecutive activations of a robot, no other robot is activated more than k times.

The main results we obtain are as follows. Firstly, we strengthen the impossibility results of Agmon and Peleg [8] since we show that, even in strictly stronger models, their impossibility result holds. Secondly, we outline the essential limits where Byzantine and crash-tolerant gathering become possible. In particular, we propose interesting lower bounds on the value that k (the scheduler bound) must take for the problem to become possible. Thirdly, we show in what situations randomized algorithms can help solve the problem, and when they cannot. To the best of our knowledge our work is the first to study the feasibility of probabilistic gathering in both fault-free and fault-prone systems. Additionally we evaluate the convergence time of our probabilistic gathering algorithms under fair schedulers using the coupling technique developed in [9]. The convergence time of our algorithms is polynomial in the size of the network in both fault-free and crash-prone environments under fair bounded schedulers. We conjecture that our bounds are optimal and hold for the case of Byzantine-prone systems.

Structure of the paper. The rest of the paper is structured as follows. Section 2 describes the robots network and system model. Section 3 formally defines the gathering problem. Section 4 propose possibility and impossibility results for deterministic and probabilistic gathering in fault-free environments. Section 5.1 and 5.2 extend the study in Section 4 to crash and Byzantine prone environments. Due to space limitations, most of the proofs are omitted, but they are included in the full version [10].

2 Model

2.1 Robots Networks

Most of the notions presented in this section are borrowed from [1, 2, 8]. We consider a network of a finite set of robots arbitrarily deployed in a geographical

[6] A Byzantine robot is a faulty robot that can behave arbitrarily, possibly in a way to prevent the other robots from gathering in a stable way.

[7] The rationale for considering a centralized daemon is that, with communication facilities, the robots can synchronize by running a mutual exclusion algorithm, such as token passing.

area. The robots are devices with sensing, computational and motion capabilities. They can observe (sense) the positions of other robots in the plane and based on these observations they perform some local computations. Furthermore, based on the local computations robots may move to other locations in the plane.

In the case robots are able to sense the whole set of robots they are referred as robots with *unlimited visibility*; otherwise robots have limited visibility. In this paper, we consider that robots have unlimited visibility.

In the case robots are able to distinguish if there are more than one robot at a given position they are referred as robots with *multiplicity knowledge*.

2.2 System Model

A network of robots that exhibit a discrete behaviour can be modeled with an I/O automaton [11]. A network of robots that exhibit a continous behaviour can be modeled with a hybrid I/O automaton [12]. This framework allows the modeling of systems that exhibit both a discrete and continuous behavior and in particular the modeling of robots networks.

The actions performed by the automaton modeling a robot are as follows:

- *Observation (input type action).*
  An observation returns a snapshot of the positions of all the robots in the visibility range. In our case, this observation returns a snapshot of the positions of all the robots;
- *Local computation (internal action).*
  The aim of a local computation is the computation of a destination point;
- *Motion (output type action).*
  This action commands the motion of robots towards the destination location computed in the local computation action.

The local state of a robot at time t is the state of its input/output variables and the state of its local variables and registers. A network of robots is modeled by the parallel composition of the individual automata that model one per one the robots in the network. A configuration of the system at time t is the union of the local states of the robots in the system at time t. An execution $e = (c_0, \ldots, c_t, \ldots)$ of the system is an infinite sequence of configurations, where c_0 is the initial configuration[8] of the system, and every transition $c_i \rightarrow c_{i+1}$ is associated to the execution of a subset of the previously defined actions.

Schedulers. A scheduler decides at each configuration the set of robots allowed to perform their actions. A scheduler is fair if, in an infinite execution, a robot is activated infinitely often. In this paper we consider the fair version of the following schedulers:

- *centralized*: at each configuration a single robot is allowed to perform its actions;

[8] Unless stated otherwise, this paper makes no specific assumption regarding the respective positions of robots in initial configurations.

– *k-bounded*: between two consecutive activations of a robot, another robot can be activated at most k times;
– *bounded regular*: between two consecutive activations of a robot, all the robots in the system perform their actions once and only once.
– *arbitrary*: at each configuration an arbitrary subset of robots is activated.

Faults. In this paper, we address the following failures:

– *crash failures*: In this class, we further distinguish two subclasses: (1) robots physically disappear from the network, and (2) robots stop all their activities, but remain physically present in the network;
– *Byzantine failures*: In this case, robots may have an arbitrary behavior.

2.3 Computational Models

The literature proposes two computational models: ATOM and CORDA. The ATOM model was introduced by Suzuki and Yamashita [1]. In this model each robot performs, once activated by the scheduler, a *computation cycle* composed of the following three actions: observation, computation and motion. The atomic action performed by a robot in this model is a computation cycle. The execution of the system can be modeled as an infinite sequence of rounds. In a round one or more robots are activated and perform a computation cycle. The ATOM model was refined by Agmon and Peleg [8]. The authors distinguish the case of hyperactive systems where all robots are activated simultaneously and non-hyperactive systems where a strict subset of robots are simultaneously activated.

The CORDA model was introduced by Prencipe [2]. This model refines the atomicity of the actions performed by each robot. Hence, robots may perform in a decoupled fashion, the atomic actions of a computation cycle. They may be interrupted by the scheduler in the middle of a computation cycle. Moreover, while a robot performs an action A, where A can be one of the following atomic actions: observation, local computation or motion, another robot may perform a totally different action B.

In this paper, we consider both models, refined with the scheduling strategies presented above. Moreover, we consider that robots are oblivious (i.e., stateless). That is, robots do not conserve any information between two computational cycles.[9] We also assume that all the robots in the system have unlimited visibility.

3 The Gathering Problem

A network of robots is in a *terminal (legitimate) configuration* with respect to the gathering requirement if all the robots share the same position in the plane. Let denote by $\mathcal{P}_{Gathering}$ this predicate.

[9] One of the major motivation for considering oblivious robots is that, as observed by Suzuki and Yamashita [1], any algorithm designed for that model is inherently self-stabilizing.

An algorithm solves the gathering problem in an oblivious system if the following two properties are verified:

- **Convergence** Any execution of the system starting in an arbitrary configuration reaches in a finite number of steps a configuration that satisfies $\mathcal{P}_{Gathering}$.
- **Termination** Any execution starting in a terminal configuration with respect to the $\mathcal{P}_{Gathering}$ predicate contains only legitimate configurations.

Gathering is difficult to achieve in most of the environments. Therefore, weaker forms of gathering were studied so far. An interesting version of this problem requires robots to *converge* toward a single location rather than reach that location in a finite time. The convergence is however considerably easier to deal with. For instance, with unlimited visibility, convergence can be achieved trivially by having robots moving toward the barycenter of the network [1].

Note that an algorithm that solves the gathering problem with oblivious or stateless robots is self-stabilizing [13].

4 Gathering in Fault-Free Environments

In this section, we refine results showing the impossibility of gathering [3, 8] by proving first that these results hold even under more restrictive schedulers than the ones considered so far [3, 8]. Interestingly, we also prove that some of these impossibility results hold even in probabilistic settings. Additionally, to circumvent these impossibility results, we propose a probabilistic algorithm that solves the fault-free gathering in both ATOM and CORDA models, under a special class of schedulers, known as k-bounded schedulers. In short, a k-bounded scheduler is one ensuring that, during any two consecutive activations of any robot, no other robot is activated more than k times.

4.1 Synchronous Robots – ATOM model

Note 4.1. Prencipe [3] proved that there is no deterministic algorithm that solves gathering in ATOM and CORDA models without additional assumptions, such as the ability to detect multiplicity.

The following lemma shows that the impossibility result of Prencipe [3] holds even under a weaker scheduler—the centralized fair bounded regular scheduler. Intuitively, a schedule of this particular scheduler is characterized by two properties: each robot is activated infinitely often and between two executions of a robot every robot in the network executes its actions exactly once. Moreover, in each configuration a single robot is allowed to execute its actions.

Lemma 4.1. *There is no deterministic algorithm that solves gathering in the ATOM model for $n \geq 3$ under a centralized fair bounded regular scheduler, without additional assumptions (e.g., multiplicity knowledge).*

Algorithm 4.1 Probabilistic gathering for robot p.

Functions:

observe_neighbors :: returns the set of robots within visibility range of robot p (the set of p's neighbors). Note that, in a system with unlimited visibility, *observe_neighbors* returns all the robots in the network.

Actions:

$\mathcal{A}_1$ :: *true* $\longrightarrow$
$\mathcal{N}_p = observe\_neighbors()$;
with probability $\alpha = \frac{1}{|\mathcal{N}_p \bigcup \{p\}|}$ do
select a robot $q \in \mathcal{N}_p \bigcup \{p\}$;
move towards q;
Remark: with probability $1 - \alpha$*, the position remains unchanged*;

Note that the deterministic gathering of two oblivious robots was proved impossible by Suzuki and Yamashita [1]. The scenario is the following: the two robots are always activated simultaneously. Consequently, they continuously swap positions, and the system never converges. In the following, we prove that, for the case of two robots, there exists a probabilistic solution for gathering in the ATOM model, under any type of scheduler. Algorithm 4.1 describes the probabilistic strategy of a robot. When chosen by the scheduler, a robot decides, with probability α, whether it will actually compute a location and move whereas, with probability $1 - \alpha$, the robot will remain stationary. The following lemma shows that Algorithm 4.1 reaches a terminal configuration with probability 1.

Lemma 4.2. *Algorithm 4.1 probabilistically solves the* 2*-gathering problem in the ATOM model under an arbitrary scheduler. The algorithm converges in 2 steps in expectation.*

The next lemma extends the impossibility result proved in Lemma 4.1 to probabilistic algorithms under unfair schedulers.

Lemma 4.3. *There is no probabilistic algorithm that solves the* n*-gathering problem, for* $n \geq 3$*, in ATOM model, under a fair centralized scheduler without additional assumptions (e.g., multiplicity knowledge).*

The key issue leading to the above impossibility is the freedom that the scheduler has in selecting a robot r until its probabilistic local computation allows r to actually move. The scenario can however no longer hold with systems in which the scheduler is k-bounded. That is, in systems where a robot cannot be activated more than k times before the activation of another robot. In this type of game robots win against the scheduler and the system converges to a terminal configuration.

Lemma 4.4. *Algorithm 4.1 probabilistically solves the* n*-gathering problem,* $n \geq 3$*, in the ATOM model under a fair* k*-bounded scheduler and without multiplicity knowledge.*

Lemma 4.5. *The convergence time of Algorithm 4.1 under fair bounded schedulers is n^2 rounds*[10] *in expectation.*

Proof. In the following, we use the coupling technique developed in [9]. Algorithm 4.1 can be seen as a Markov chain. Let's call it $\mathcal{A}$ hereafter. A coupling for Algorithm 4.1, is a Markov chain $(X_t, Y_t)_{t=1}^{\infty}$ with the following properties: (1) each of the variables (X_t), (Y_t) is a copy of the Markov chain $\mathcal{A}$ (given initial configurations $X_0 = x$ and $Y_0 = y$); and (2) if $X_t = Y_t$ then $X_{t+1} = Y_{t+1}$. Intuitively, the coupling time is the expected time for the two processes X_t and Y_t to reach the agreement property ($X_t = Y_t$). As shown in Theorem 1 [9] the coupling time is also an upper bound for the hitting time or convergence time of a self-stabilizing algorithm.

Assume (X_t) and (Y_t) are two copies of the Markov chain modeling Algorithm 4.1. Let us denote by $\delta(X_t, Y_t)$ the distance between X_t and Y_t (the number of robots that do not share identical positions in X_t and Y_t). In the worst case, $\delta(X_t, Y_t) = n$ (where n is the number of robots in the network). In the following we show that, with positive probability, the distance between X_{t+1} and Y_{t+1} decreases. Assume that the scheduler chooses robot p at instant t, and assume that p does not share the same position in X_t and Y_t. With positive probability, $X_{t+1}(p) = Y_{t+1}(p)$. Assume that the scheduler chooses two or more robots in t. Since the scheduler is bounded, within a round of size R, $\delta(X_{t+R}, Y_{t+R}) \leq \delta(X_t, Y_t) - 1$. Following the result proved in Theorem 2 [9], the coupling time for this chain is bounded from above by $\frac{B}{1-\beta}$. Where B is the maximal value of the distance metric (in our case this value is n) and β is the constant such that for all (X_t, Y_t) we have $E[\delta(X_{t+1}, Y_{t+1})] \leq \beta\delta(X_t, Y_t)$. In our case, $\beta = \frac{n-1}{n}$. So, the hitting (convergence) time for Algorithm 4.1 is n^2 rounds in expectation. □

4.2 Asynchronous Robots – CORDA model

In the following, we analyze the feasibility of gathering in a stronger model, namely, CORDA. Obviously, all the impossibility results proved in the ATOM model hold for CORDA [14].

The next lemma states that 2-gathering, while probabilistically feasible in ATOM model, is impossible in the CORDA model under an arbitrary scheduler.[11] We recall that, in the CORDA model, robots can be interrupted by the scheduler during a computation cycle.

Lemma 4.6. 2*-gathering is impossible in the CORDA model under an arbitrary scheduler.*

Now, instead of an arbitrary scheduler, we consider a k-bounded scheduler, and obtain the following possibility result.

[10] A round is the longest fragment of an execution between two successive actions of the same process. Following the variant of the chosen k-bounded scheduler a round can have k steps or kn steps.

[11] Note that 2-gathering is trivially possible under a centralized scheduler.

Lemma 4.7. *Algorithm 4.1 probabilistically solves the n-gathering problem, $n \geq 2$, in the CORDA model under a k-bounded scheduler and without multiplicity knowledge.*

5 Fault Tolerant Gathering

5.1 Crash Tolerant Gathering

In the following we extend the study of the gathering feasibility to fault-prone environments. In this section (n, f) denotes a system with n correct robots but f and the considered faults are the crash failures. As mentioned in the model, Section 2, in an (n, f) crash-prone system there are two types of crashes: (1) the crashed robots completely disappear from the system, and (2) the crashed robots are still physically present in the system, however they stop the execution of any action. In the sequel we analyze both situations.

Lemma 5.1. *In a crash-prone system, $(3, 1)$-gathering is deterministically possible under a fair centralized regular scheduler.*

The following lemma proves that the previous result does not hold in systems with more than three robots. More precisely, this lemma expands the impossibility results proved in Lemma 4.1 and 4.3 to crash-prone environments.

Lemma 5.2. *In a crash-prone system, there is no deterministic algorithm that solves the $(n, 1)$-gathering problem, $n \geq 4$, under a fair bounded regular centralized scheduler without additional assumptions (e.g, multiplicity knowledge).*

Lemma 5.3. *In a crash-prone system, there is no probabilistic algorithm that solves the $(n, 1)$-gathering problem, $n \geq 3$, under a fair centralized scheduler without additional assumptions (e.g., multiplicity knowledge).*

The key argument in the previous impossibility proof is that the scheduler has the possibility to choose a robot until that robot is allowed to move (by its probabilistic algorithm). In some sense, the scheduler managed to derandomize the system. However, the process of derandomization is no longer possible with a bounded scheduler. The following lemma proves that $(n, 1)$-gathering is probabilistically possible under a bounded scheduler and without additional assumptions.

Lemma 5.4. *In a crash-prone system, Algorithm 4.1 is a probabilistic solution for the gathering problem in systems with n correct robots but one and under a bounded scheduler.*

In the following, we extend our study to systems with more than one faulty robot. Hereafter, (n, f)-gathering refers to the gathering problem in a system with n correct robots but f. If the faulty robots disappear from the system, then the problem trivially reduces to the study of a fault-free gathering with $n-f$ correct robots. In contrast, in systems where faulty robots remain physically

present in the network after crashing, the problem is far from being trivial. Obviously, gathering all the robots including the faulty ones, is impossible since faulty robots may possibly have crashed at different locations.

From this point on, we study the feasibility of a weaker version of gathering, referred to as *weak gathering*. The (n, f)-*weak gathering* problem requires that, in a terminal configuration, only the *correct* robots must share the same position. The following lemma proves the impossibility of deterministic and probabilistic weak gathering under centralized bounded and fair schedulers and without additional assumptions.

Lemma 5.5. *In a crash-prone system, there is neither a probabilistic nor a deterministic algorithm that solves the (n, f)-weak gathering problem, $n \geq 3$ and $f \geq 2$, under a fair centralized regular scheduler without additional assumptions.*

Algorithm 5.1 Deterministic fault-tolerant weak gathering for robot p

Functions:
observe_neighbors :: returns the set of robots within the vision range of robot p (the set of p's neighbors);
maximal_multiplicity :: returns a robot in the group with the maximal multiplicity; or, if several such groups exists, makes an arbitrary choice among them;

Actions:
$\mathcal{A}_1$ :: *true* $\longrightarrow$
$\mathcal{N}_p = observe\_neighbors()$;
$q = maximal\_multiplicity(\mathcal{N}_p)$;
move towards q;

An immediate consequence of the previous lemma is the necessity of an additional assumption (e.g., multiplicity knowledge), even for probabilistic solutions under bounded schedulers.

In the sequel, we identify the conditions under which the weak gathering accepts deterministic and probabilistic solutions. Algorithm 5.1 proposes a deterministic solution for the weak gathering that works under both centralized and bounded schedulers. The idea of the algorithm is the following: a robot, once chosen by the scheduler, moves to the group with the maximal multiplicity – "attraction action". In case that all groups have the same multiplicity, the chosen robot will go to the location of another robot – "unbalanced action". The attraction action helps the convergence while the unbalanced action breaks the symmetry.

Lemma 5.6. *In a crash-prone system, Algorithm 5.1 deterministically solves the (n, f)-weak gathering problem, $f \geq 2$, under a centralized scheduler if robots are aware of the system multiplicity.*

Algorithm 5.2 Probabilistic fault-tolerant gathering for robot p with multiplicity knowledge

Functions:
observe_neighbors :: returns the set of robots within the vision range of robot p (the set of p's neighbors);
maximal_multiplicity :: returns the set of robots with the maximal multiplicity;

Actions:

$\mathcal{A}_1$ :: *true* $\longrightarrow$
  $\mathcal{N}_p = observe\_neighbors()$;
  if $p \in maximal\_multiplicity(\mathcal{N}_p) \wedge |maximal\_multiplicity(\mathcal{N}_p)| > 1$ then
    with probability $\frac{1}{|maximal\_multiplicity(\mathcal{N}_p)|}$ do
      select a robot $q \in maximal\_multiplicity(\mathcal{N}_p)$;
      move towards q;
  else
    select a robot $q \in maximal\_multiplicity(\mathcal{N}_p)$;
    move towards q;

In the following we show that (n, f)-weak gathering can be solved under arbitrary schedulers using a probabilistic algorithm, Algorithm 5.2, and multiplicity knowledge. Algorithm 5.2 works as follows. When a robot is chosen by the scheduler it moves to the group with maximal multiplicity. When all groups have the same size, then the robot tosses a coin to decide if it moves or holds the current position.

Lemma 5.7. *In a crash-prone system, Algorithm 5.2 probabilistically solves the (n, f)-weak gathering problem, $f \geq 2$, under an unfair scheduler if robots are aware of the system multiplicity.*

5.2 Byzantine Tolerant Gathering

In the following we study the gathering feasibility in systems prone to Byzantine failures. In the sequel (n, f) denotes a system with n correct robots but f. Agmon and Peleg [8] proved that gathering in Byzantine environments is impossible in ATOM and CORDA models for the case $(3, 1)$. The impossibility proof is given for the case of the ATOM model and algorithms that are not hyperactive. The following lemma proves the $(3, 1)$-gathering impossibility under the weakest scheduler, in particular the centralized, fair and regular.

Lemma 5.8. *In a Byzantine-prone system, there is no deterministic algorithm that solves $(3, 1)$-weak gathering under a fair, centralized and bounded regular scheduler without additional assumptions.*

Note 5.1. Note that Algorithm 5.1 solves the Byzantine $(3, 1)$-weak gathering under a centralized regular scheduler and multiplicity knowledge. The cycle created in the impossibility proof is broken because the Byzantine robot cannot play the attractor role.

The following lemma shows that if the scheduler is relaxed, the $(3,1)$-weak gathering becomes impossible even if robots are aware of the system multiplicity.

Lemma 5.9. *In a Byzantine-prone system, there is no deterministic algorithm that solves the $(3,1)$-weak gathering, even when robots are aware of the system multiplicity, under a centralized fair k-bounded scheduler with $k \geq 2$.*

Note 5.2. Byzantine $(n,1)$-weak gathering for any odd $n > 4$ is possible under any fair centralized scheduler and multiplicity knowledge. The algorithm is trivial: a robot moves to the group with maximal multiplicity.

The following lemma establishes a lower bound for the bounded centralized scheduler that prevents the deterministic gathering.

Lemma 5.10. *In a Byzantine-prone system, there is no deterministic algorithm that solves $(n,1)$-weak gathering, with $n \geq 2$ even, under a centralized k-bounded scheduler for $k \geq (n-1)$. This result holds even when robots are aware of the system multiplicity.*

Corollary 5.1. *Byzantine $(n,1)$-weak gathering is possible under a centralized scheduler:*

- *in systems where $n \geq 4$ is odd, robots have multiplicity knowledge and the scheduler is fair, or*
- *in systems where $n \geq 2$ is even and the scheduler is k-bounded with $k \leq (n-2)$.*

The following lemma states the lower bound for a bounded scheduler that prevents deterministic gathering.

Lemma 5.11. *In Byzantine-prone systems, there is no deterministic algorithm that solves (n,f)-weak gathering, $f \geq 2$, under a centralized k-bounded scheduler with $k \geq \left\lceil \frac{n-f}{f} \right\rceil$ when n is even, and with $k \geq \left\lceil \frac{n-f}{f-1} \right\rceil$ when n is odd, even when the robots can detect multiplicity.*

Proof.
- **Even case.** Similar to the $(n,1)$ case above, assume that the system starts in an initial configuration in which all robots are arranged in two groups. Assume the same scheduler as in the $(n,1)$ case: for each move of a correct robot the scheduler chooses a Byzantine robot. The Byzantine robot will try to balance the system equilibrium hence it will move towards the old location of the correct robot. In order to win the game the Byzantine robots need to move each time a correct robot moves. Since there are $n-f$ correct robots in the system, the scheduler has to be bounded by no less than $\left\lceil \frac{n-f}{f} \right\rceil$ for the Byzantine team to win.
- **Odd case.** For the odd case assume an initial configuration where robots but one (a Byzantine one) are arranged in two groups. When chosen by the scheduler the Byzantine robot not member of a group moves such that the equilibrium between the two groups does not change. Let denote G_1 and

G_2 the two groups. Consider the following schedule. Every time a correct robot, member of G_i, moves, a Byzantine robot moves as well in the opposite direction. Hence the system equilibrium does not change. The game is similar to the even case. The only difference is that the number of Byzantine robots that influence the faith of the game is $f-1$. Therefore, in order to win the game, the Byzantine team needs a k-bounded scheduler bounded by $k \geq \left\lceil \frac{n-f}{f-1} \right\rceil$.

□

Lemma 5.12. *In systems with Byzantine faults, Algorithm 5.2 probabilistically solves the (n, f)-weak gathering, $n \geq 3$, problem under a bounded scheduler and multiplicity detection.*

6 Conclusion

The results presented here extend that of prior work on the possibility and impossibility of gathering in fault-free and both crash-prone and Byzantine-prone systems. For instance, we strengthen several prior impossibility results by showing that they still hold against weaker schedulers, and under various failure models. We also mark out more accurately the limit between possibility and impossibility by deriving appropriate upper and lower bounds.

To the best of our knowledge, this is actually the first study that considers probabilistic solutions to solve the gathering problem. Here, we identify conditions under which a probabilistic solution exists, as well as conditions for which not even a probabilistic solution exists.

The main results of the paper are summed up in Table 1 for fault-free systems; in Table 2 for crash-prone systems; and in Table 3 for the weak gathering problem in Byzantine-prone systems.

As an open question, some of the impossibility proofs only consider the use of randomization for determining whether a robot takes actions or not when it is activated. One can argue that using randomization in a different way may possibly change some of the lower bounds presented here. We conjecture that the bounds will hold even if randomization is used differently.

References

1. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: Formation of geometric patterns. SIAM Journal of Computing **28**(4) (1999) 1347–1363
2. Prencipe, G.: CORDA: Distributed coordination of a set of autonomous mobile robots. In: Proc. 4th European Research Seminar on Advances in Distributed Systems (ERSADS'01), Bertinoro, Italy (2001) 185–190
3. Prencipe, G.: On the feasibility of gathering by autonomous mobile robots. In Pelc, A., Raynal, M., eds.: Proc. Structural Information and Communication Complexity, 12th Intl Coll., SIROCCO 2005. Volume 3499 of LNCS., Mont Saint-Michel, France, Springer (2005) 246–261

Table 1. Summary of the main results in fault-free environments.

ATOM	CORDA	mult.	no mult.	centralized	regular	k-bounded	arbitrary	unfair	Conditions	Solution	Source
●			●				●	○	–	*Impossible*	Prencipe [3] (Note 4.1)
●	○		●	●	●	○	○	○	$n \geq 3$	*No deterministic*	Lemma 4.1
●		●		○	○	○	●		$n = 2$	Probabilistic	Lemma 4.2
●	○		●	●					$n \geq 3$	*No probabilistic*	Lemma 4.3
●		○	●		○	●			$n \geq 3$	Probabilistic	Lemma 4.4
	●	●					●	○	$n = 2$	*Impossible*	Lemma 4.6
○	●	○	●		○	●			–	Probabilistic	Lemma 4.7
"●" means explicit; "○" means implicit; negative results are in italic											

Table 2. Summary of the main results in crash-prone systems.

ATOM	CORDA	mult.	no mult.	centralized	regular	k-bounded	arbitrary	unfair	Conditions	Solution	Source
●			●	●	●				$n = 3$, $f = 1$	Deterministic	Lemma 5.1
●	○		●	●	●	○	○	○	$n \geq 4$, $f \geq 1$	*No deterministic*	Lemma 5.2
●	○		●	●			○	○	$n \geq 3$, $f \geq 1$	*No probabilistic*	Lemma 5.3
●			●		●	●			$f = 1$	Probabilistic	Lemma 5.4
●	○		●	●	●	○	○	○	$n \geq 3$, $f \geq 2$, weak	*Impossible*	Lemma 5.5
●		●		●					$f \geq 2$, weak	Deterministic	Lemma 5.6
●		●		○	○	○	○	●	$f \geq 2$, weak	Probabilistic	Lemma 5.7
'●" means explicit; "○" means implicit; negative results are in italic											

Table 3. Summary of the main results in Byzantine-prone systems.

ATOM	CORDA	mult.	no mult.	centralized	regular	k-bounded	arbitrary	unfair	Conditions	Solution	Source
●	○		●				●	○	$n = 3$, $f = 1$	*No deterministic*	Agmon–Peleg [8]
●	○		●	●	●	○	○	○	$n = 3$, $f = 1$	*No deterministic*	Lemma 5.8
●		●		●	●				$n = 3$, $f = 1$	Deterministic	Note 5.1
●	○	●	○	●		●	○	○	$n = 3$, $f = 1$, $k \geq 2$	*No deterministic*	Lemma 5.9
●		●		●					n odd, $n > 4$, $f = 1$	Deterministic	Note 5.2
●	○	●	○	●		●	○	○	n even $n \geq 2$, $f = 1$, $k \geq n-1$	*No deterministic*	Lemma 5.10
●	○	●	○	●		●	○	○	$f \geq 2$, $k \geq \begin{cases} \frac{n-f}{f} & \text{if } n \text{ even} \\ \frac{n-f}{f-1} & \text{if } n \text{ odd} \end{cases}$	*No deterministic*	Lemma 5.11
●		●			○	●			$n \geq 3$, weak	Probabilistic	Lemma 5.12
"●" means explicit; "○" means implicit; negative results are in italic											

4. Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Gathering of asynchronous mobile robots with limited visibility. Theoretical Computer Science **337** (2005) 147–168
5. Souissi, S., Défago, X., Yamashita, M.: Eventually consistent compasses for robust gathering of asynchronous mobile robots with limited visibility. Research Report IS-RR-2005-010, JAIST, Ishikawa, Japan (2005)
6. Ando, H., Oasa, Y., Suzuki, I., Yamashita, M.: Distributed memoryless point convergence algorithm for mobile robots with limited visibility. IEEE Trans. on Robotics and Automation **15**(5) (1999) 818–828
7. Cohen, R., Peleg, D.: Convergence of autonomous mobile robots with inaccurate sensors and movements. In Durand, B., Thomas, W., eds.: 23rd Annual Symposium on Theoretical Aspects of Computer Science (STACS'06). Volume 3884 of LNCS., Marseille, France, Springer (2006) 549–560
8. Agmon, N., Peleg, D.: Fault-tolerant gathering algorithms for autonomous mobile robots. In: Proc. 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2004), New Orleans, LA, USA (2004) 1070–1078
9. Fribourg, L., Messika, S., Picaronny, C.: Coupling and self-stabilization. Distributed Computing **18**(3) (2006) 221–232
10. Défago, X., Gradinariu, M., Messika, S., Raipin-Parvédy, P.: Fault-tolerant and self-stabilizing mobile robots gathering: Feasibility study. Tech. Rep. PI-1802, IRISA, Rennes, France (2006)
11. Lynch, N.A.: Distributed Algorithms. Morgan Kaufmann, San Francisco, CA, USA (1996)
12. Lynch, N.A., Segala, R., Vaandrager, F.W.: Hybrid I/O automata. Information and Computation **185**(1) (2003) 105–157
13. Dolev, S.: Self-Stabilization. MIT Press (2000)
14. Prencipe, G.: The effect of synchronicity on the behavior of autonomous mobile robots. Theory of Computing Systems **38**(5) (2005) 539–558