Fault-tolerant and Self-stabilizing Mobile Robots Gathering - Feasibility Study -

Xavier Défago, Maria Gradinariu, Stéphane Messika, Philippe Raïpin-Parvédy

I R I S A
INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

PUBLICATION
INTERNE
N° 1802

FAULT-TOLERANT AND SELF-STABILIZING MOBILE ROBOTS GATHERING
— FEASIBILITY STUDY —

X. DÉFAGO M. GRADINARIU S. MESSIKA P. RAIPIN-PARVÉDY

ISSN 1166-8687

IRISA
CAMPUS UNIVERSITAIRE DE BEAULIEU - 35042 RENNES CEDEX - FRANCE

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES
Campus de Beaulieu – 35042 Rennes Cedex – France
Tél. : (33) 02 99 84 71 00 – Fax : (33) 02 99 84 71 71
http://www.irisa.fr

Fault-tolerant and Self-stabilizing Mobile Robots Gathering — Feasibility Study —

X. Défago[*] M. Gradinariu S. Messika[**] P. Raipin-Parvédy

Systèmes communicants
Projet Adept

Publication interne n ° 1802 — may 2006 — 20 pages

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.

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.

Key-words: mobile robots, fault-tolerance, self-stabilization, self-organization, impossibility results, deterministic gathering, probabilistic gathering

(Résumé : tsvp)

[*] JAIST, Japon
[**] LRI/Universite Paris Sud, France

Centre National de la Recherche Scientifique (unp 6074) Université de Rennes 1 – Insa de Rennes
Institut National de Recherche en Informatique et en Automatique – unité de recherche de Rennes

Rassemblement de robots mobiles tolerant aux fautes et auto-stabilisant. Etude de fesabilite

Résumé : Nous étudions la faisabilité (deterministe et probabiliste) du problème de rassemblement d'un ensemble de robots mobiles. Dans le modèle considéré les robots peuvent exhiber un comportement byzantin.

Mots clés : rassemblement réparti, robots mobiles, fautes byzantines, tolerance aux fautes, auto-stabilisation

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.

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 [12], 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 [8], activations are completely asynchronous, for instance allowing robots to be seen while moving.

Suzuki and Yamashita [12] 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.[1] Prencipe [9] 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.* [5] 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.* [11] consider a system in which compasses are not necessarily consistent initially. Ando *et al.* [2] propose a gathering algorithm for ATOM model with limited visibility. Cohen and Peleg [3] 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 [1]. They prove that gathering of correct robots (referred 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

[1] 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.

deterministic gathering algorithm exists in ATOM model that can tolerate a Byzantine[2] 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[3] (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 [1] 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 [6]. The convergence time of our algorithms is polynomial in the size of the network in both fault-free and crash-prone environements 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 for our results are detailed in the Annexes section.

2 Model

2.1 Robots Networks

Most of the notions presented in this section are borrowed from [12, 8, 1]. We consider a network of a finite set of robots arbitrarily deployed in a geographical 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] 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.

[3] 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.

2.2 System model

A network of robots can be modeled with a hybrid I/O automata [7]. This framework allows the modeling of systems that exhibit a discrete and continuous behavior and in particular the modeling of robots networks.

The actions executed by the automaton modeling a robot are: *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 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 execute 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 execute its actions; *k-bounded* - between two consecutive executions of a robot another robot can execute at most k actions; *bounded regular* - between two consecutive executions of a robot all the robots in the system execute their actions one and only one time; *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 distinguish two sub-classes: (1) robots physically disappear from the network. (2) robots stop all their activities, however they are still 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 [12]. 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 executed 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 [1]. 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 [8]. This model refines the atomicity of the actions executed by each robot. Hence, robots may execute in a decoupled fashion the atomic actions of a computation cycle. They may be interrupted by the scheduler during the execution of the 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 (stateless). That is, robots do not conserve any information between two computational cycles. 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.

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 [12].

Note that an algorithm that solves the gathering problem with oblivious or stateless robots is self-stabilizing [4].

4 Gathering in Fault-Free environments

In this section, we refine results showing the impossibility of gathering [10, 1] by proving first that these results hold even under more restrictive schedulers than the ones considered so far [10, 1]. 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 [9] 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 [9] 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 (ex. 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* $\longrightarrow$

$\mathcal{N}_p = observe\_neighbors()$;
select a robot $q \in \mathcal{N}_p \bigcup \{p\}$ with probability $\alpha = \frac{1}{|\mathcal{N}_p \bigcup \{p\}|}$;
move towards q;
Note that with probability $\frac{1}{|\mathcal{N}_p \bigcup \{p\}|}$ *the current position is not changed*;

Note that the deterministic gathering of two oblivious robots was proved impossible by Suzuki and Yamashita [12]. 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 ($n = 2$) 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 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 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 ATOM model under a fair bounded scheduler and without multiplicity knowledge. The algorithm converges in $O(n^2)$ rounds*[4].

[4] 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

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

Proof. In the following we use the coupling technique developed in [6]. Algorithm 4.1 can be seen as a Markov chain. Let's call it $\mathcal{A}$ in the following. 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 $Y_0 = y$); (2) If $X_t = Y_t$ then $X_{t+1} = Y_{t+1}$. Intuitivelly, 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 [6] 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 denote by $\delta(X_t, Y_t)$ the distance between X_t and Y_t (the number of robots that does not share identic 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 the scheduler choses the robot p at moment t and assume 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 the scheduler choses 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 proven in Theorem 2 [6], the coupling time for this chain is upper bounded 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) $E[\delta(X_{t+1}, Y_{t+1})] \leq \beta\delta(X_t, Y_t)$. 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 - CORDA. Obviously, all the impossibility results proved in ATOM model hold for CORDA [9].

The next lemma states that 2-gathering, probabilistically feasible in ATOM model, is impossible in CORDA model under an arbitrary scheduler [5]. We recall that in CORDA model robots can be interrupted by the scheduler during a computation cycle.

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

Lemma 4.7 *Algorithm 4.1 probabilistically solves the n-gathering problem, $n \geq 2$, in 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 a (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 the both situations.

[5] Note that 2-gathering is trivially possible in a centralized scheduler.

Lemma 5.1 *(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 *There is no deterministic algorithm that solves (n,1) gathering problem, $n \geq 4$, under a fair bounded regular centralized scheduler without additional assumptions (ex. multiplicity knowledge).*

Lemma 5.3 *There is no probabilistic algorithm that solves (n,1) gathering problem, $n \geq 3$, under a fair centralized scheduler without additional assumptions (ex. multiplicity knowledge).*

The key argument in the previous impossibility proof is the scheduler freedom to choose a robot until it is allowed (by its probabilistic algorithm) to move. In some sens the scheduler managed to derandomize the system. However, the process of the derandomization is not 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 *Algorithm 4.1 is a probabilistic solution for the gathering problem in systems with n correct robots but one and under a bounded scheduler. The algorithm converges in $(n-1)^2$ rounds in expectation.*

In the following we extend our study to systems with more than one faulty robot. In the following (n,f)-gathering refers the gathering problem in a system with n correct robots but f. If the faulty robots disappear from the system than the problem trivially reduces to the study of fault-free (n-f)-gathering. In systems where the faulty robots are still physically present in the network the problem is far from being trivial. Obviously, gathering all the robots including the faulty ones at the same position is impossible since faulty robots may not share, from the beginning, the same position.

In the following we study the feasibility of a weaker version of gathering, referred in the following, *weak gathering*. (n,f) *weak gathering* assumes that in a terminal configuration only the correct robots have to 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 *There is no probabilistic and 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.*

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

In the following 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 the location of another robot - "unbalanced action". The attraction action helps the convergence while the unbalanced action breaks the symmetry.

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 in case that all nodes have the same multiplicity returns a node in the neighborhood of p;

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

Lemma 5.6 *Algorithm 5.1 deterministically solves (n,f) weak gathering problem, $f \geq 2$, under a centralized scheduler if nodes 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
select a robot $q \in maximal\_multiplicity(\mathcal{N}_p)$ with probability $\frac{1}{|maximal\_multiplicity(\mathcal{N}_p)|}$;
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 *Algorithm 5.2 probabilistically solves the (n,f) weak gathering problem, $f \geq 2$, under an unfair scheduler if nodes 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 [1] proved that gathering in Byzantine environments is impossible in ATOM and CORDA model for the case (3,1). The impossibility proof of Peleg is given for the case of ATOM model and not hyperactive algorithms. The following lemma proves the (3,1)-gathering impossibility under the weakest scheduler, in particular the centralized, fair and regular.

Lemma 5.8 *There is no deterministic algorithm that solves the (3,1) gathering under a fair, centralized and bounded regular scheduler without additional assumptions.*

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

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

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

Note 5.2 *(n,1)-gathering for any odd* $n \geq 4$ *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 *There is no deterministic algorithm that solves (n,1)-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 *(n,1)-gathering is possible under a centralized scheduler:*

- *in systems where* $n \geq 4$ *is odd, nodes 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 *There is no deterministic algorithm that solves (n,f)-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 nodes are aware of the system multiplicity.*

Proof.

- **Even case.** As for the case (n,1) assume the system starts in the following initial configuration: nodes are arranged in two groups. Assume the same scheduler as for the (n,1) case: for each move of a correct node the scheduler chooses a byzantine node. The byzantine node will try to balance the system equilibrium hence it will move towards the old location of the correct node. In order to win the game the byzantine nodes need to move each time a correct node moves. Since in the system there are $(n-f)$ correct nodes the scheduler has to be at least $\left\lceil \frac{n-f}{f} \right\rceil$ bounded in order to allow the byzantine team to win.

- **Odd case.** For the odd case assume an initial configuration where nodes 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 nodes that influence the faith of the game is $f-1$. Therefore, in order to win the game the byzantine team needs a $k \geq \left\lceil \frac{n-f}{f-1} \right\rceil$ bounded scheduler.

□

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

6 Conclusions and Open Problems

In this paper, we studied the limits of gathering feasibility in both fault-free and fault prone environments, by considering centralized schedulersand k-bounded schedulers.

The main results we obtained are as follows. Firstly, we strengthened the impossibility results of Agmon and Peleg [1] since we show that, even in strictly stronger models, their impossibility result holds. Secondly, we outlined the essential limits where Byzantine and crash-tolerant gathering become possible. In particular, we proposed interesting lower bounds on the value that k (the scheduler bound) must take for the problem to become possible. Thirdly, we showed 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 [6]. The convergence time of our algorithms is polynomial in the size of the network in both fault-free and crash-prone environements under fair bounded schedulers. We conjecture that our bounds are optimal and hold for the case of byzantine-prone systems.

References

[1] N. Agmon and D. Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. In *Proc. 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2004)*, pages 1070–1078, New Orleans, LA, USA, January 2004.

[2] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Trans. on Robotics and Automation*, 15(5):818–828, October 1999.

[3] R. Cohen and D. Peleg. Convergence of autonomous mobile robots with inaccurate sensors and movements. In B. Durand and W. Thomas, editors, *23rd Annual Symposium on Theoretical Aspects of Computer Science (STACS'06)*, volume 3884, pages 549–560, Marseille, France, February 2006.

[4] S. Dolev. *Self-Stabilization.* MIT Press, 2000.

[5] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of asynchronous mobile robots with limited visibility. *Theoretical Computer Science*, 337:147–168, 2005.

[6] L. Fribourg, S. Messika, and C. Picaronny. Coupling and self-stabilization. *Distributed Computing*, 18(3):221–232, 2006.

[7] N. A. Lynch. *Distributed Algorithms.* Morgan Kaufmann, San Francisco, CA, USA, 1996.

[8] G. Prencipe. CORDA: Distributed coordination of a set of autonomous mobile robots. In *Proc. 4th European Research Seminar on Advances in Distributed Systems (ERSADS'01)*, pages 185–190, Bertinoro, Italy, May 2001.

[9] G. Prencipe. The effect of synchronicity on the behavior of autonomous mobile robots. *Theory of Computing Systems*, 38(5):539–558, September 2005.

[10] G. Prencipe. On the feasibility of gathering by autonomous mobile robots. In A. Pelc and M. Raynal, editors, *Proc. Structural Information and Communication Complexity, 12th Intl Coll., SIROCCO 2005*, volume 3499, pages 246–261, Mont Saint-Michel, France, May 2005.

[11] S. Souissi, X. Défago, and M. Yamashita. Eventually consistent compasses for robust gathering of asynchronous mobile robots with limited visibility. Technical Report IS-RR-2005-010, July 2005.

[12] I. Suzuki and M. Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM Journal of Computing*, 28(4):1347–1363, 1999.

7 Annexes

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 (ex. multiplicity knowledge).

Proof. With no loss of generality, consider a system with three robots: r_1, r_2 and r_3 arranged as shown on Figure 7.1. In the following, we construct an infinite execution that never converges. Consider a schedule *Sch* that invariably applies the following scenario: r_3 is chosen first, then r_1, then r_2. The schedule verifies the centralized fair bounded regular scheduler.

Consider now an initial configuration of the system such that robots r_2 and r_3 have the same initial location. Since r_3 does not have the ability to detect multiplicity, it observes the existence of two groups of robots but does not know how many robots are in each group. Since r_3 is oblivious, it cannot deterministically localize r_2 in the groups. Then, according to *Sch*, we can always find a scenario in which the system will cycle infinitely. For instance, *Sch* can generate the following change in robots topology: $(r_1, r_2r_3) \rightarrow (r_1r_3, r_2) \rightarrow (r_3, r_1r_2) \rightarrow (r_2r_3, r_1)$. It follows that the system never converges. □

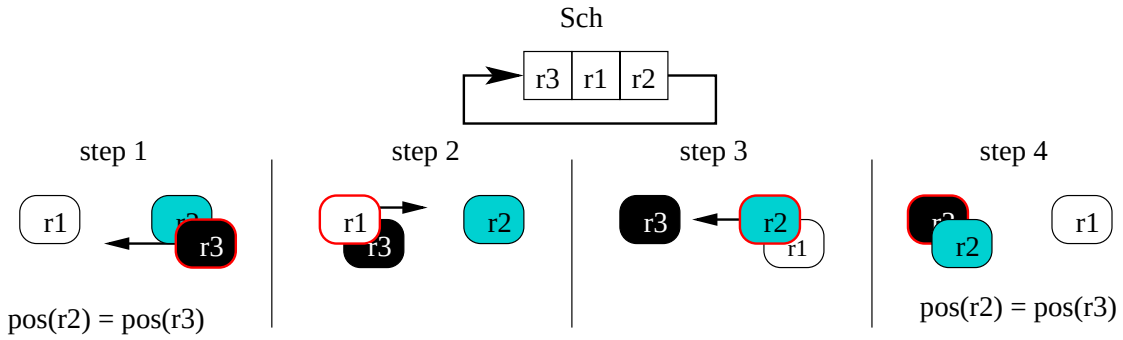

Figure 7.1: Lemma 4.1: constructing an infinite execution.

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

Proof. Consider two robots r_1 and r_2, and an arbitrary initial configuration c.

If r_1 and r_2 are gathered at the same location, they are already in a terminal configuration and thus neither will move, regardless of activations and the probability t. This leaves the case when r_1 and r_2 are not gathered initially.

Consider some step i when the two robots have distinct locations. If only one of the robots is activated by the scheduler, then there is a probability α that the robot moves, and thus the robots end up gathered in the next configuration (terminal). If both robots are activated at step i, then, they end up in a terminal configuration if either one of them is activated. This occurs with probability $2\alpha(1-\alpha)$.

Consequently, the probability of reaching a terminal configuration during step i is at least $q = \min(\alpha, 2\alpha(\alpha-1)) > 0$, regardless of the choice of the scheduler. It follows that the probability of reaching a terminal configuration at step $s > i$ is at least $p = 1-(1-q)^s$. Thus, the probability for the system to reach a terminal configuration is $\lim_{s\to\infty}(1-(1-q)^s) = 1$.

In the following we evaluate the convergence time of the algorithm. Let X_t be the probabilistic variable that models the configuration of the system at time t. Let $\mathcal{L}$ be the set of terminal configurations (robots are gathered at the same location). Let $T_{\mathcal{L}}$ the time to reach a configuration in $\mathcal{L}$, $T_{\mathcal{L}} = min\{t, X_t \in \mathcal{L}\}$. The convergence time of the algorithme (also known as the hitting time) is $E[T_{\mathcal{L}}] = \sum_{i=1}^{\infty} i(\frac{1}{2})^i = 2$. □

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).

Proof. Consider an initial configuration in which nodes form two groups, G_1 and G_2. Consider a schedule such that robots move from G_1 to G_2 and vice-versa, without ever reaching a terminal configuration. Consider now the following simple schedule: select one robot from the group with maximal multiplicity.[6] With no loss of generality, let the selected robot be from G_1. Recall that the algorithm executed by each robot is probabilistic. Therefore, it is possible that the selected robot needs to be activated several times until its coin allows it to actually move. Once the selected robot arrives in the opposite group, G_2, select one of the robots in G_2 and activate it until it is ready to move.

Even if the scheduler is fair the above schedule generates an infinite execution that does not lead to a terminal configuration. □

Lemma 4.4 Algorithm 4.1 probabilistically solves the n-gathering problem, $n \geq 3$, in ATOM model under a fair bounded scheduler and without multiplicity knowledge. The convergence time of the algorithme is $O(n^2)$ rounds.

Proof. We show that with high probability a group of at least two robots is formed and then, with high probability this group will attract the other robots (even if these robots are not aware of the system multiplicity). Assume an arbitrary initial configuration - no two robots share the same position. Let c_0 be this configuration. Initially the scheduler chooses s robots ($s \geq 1$). With probability α^s all the s robots choose as meeting point the same robot and move towards this robot. Let's call this robot the core of G. Let G be the group formed by these $s+1$ robots. Let G' be the robots which do not share the same position as the robots in G. Let c_1 be the configuration obtained after the first choice of the scheduler. Starting with c_1 consider the following scenario. Each time a robot in G is chosen by the scheduler it selects with probability α a node in G (itself for example or the core of G). Each time a robot in G' is chosen by the scheduler it selects with probability α a robot in G (for example the core of G). In the worst case, the robots in G' have to wait to enter in G until each robot, already in this set, is chosen at most k times. Note that each time a robot in G' is chosen, the size of G' probabilistically decreases and the size of G probabilistically increases. Following the above scenario, the probability to obtain a configuration c_2 such that the size of G increases by one and the size of G' decreases by one is superior to $\alpha^{k(s+1)}\alpha$. The size of G' is bounded. Therefore, by applying the previous scenario a terminal configuration is reached in a finite number of steps with positive probability, superior to $\alpha^s\alpha^{k\sum_{i=1,(n-s-1)}(s+i)}\alpha^{n-s-1} \geq \alpha^{k\frac{n(n-1)}{2}+(n-1)} = \epsilon$, where k is the scheduler bound. The probability that the system converges to a terminal configuration is $\lim_{m\to\infty} \epsilon(1 + \sum_{i=1,m}(1-\epsilon)^i) = 1$. The following lemma 4.5 provides the polynomial bound on the convergence time. □

[6] If the groups have the same multiplicity then select one of the two groups arbitrarily.

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

Proof. We exhibit a schedule such that the two robots will never reach a terminal configuration. Consider a non terminal configuration - the two robots does not share the same position. First the scheduler chooses one of the robots until it has the right to move (probabilistically or deterministically). Then, since the move and the internal computations are decoupled, the scheduler has the right to stop the robot just before it starts to move and to choose the second robot. The same scenario is applied for the second robot which will eventually be allowed to move. Recall that this scenario is applied to both deterministic and probabilistic robots. The two robots are ready to move. At this point the scheduler choose the both robots so they will exchange their places. Note that the position switch is possible since the computation of the target location was done in the observation period. The obtained configuration is symmetrical to the initial configuration. Therefore the scheduler can apply exactly the same schedule as above in order to win the game (i.e to prevent the gathering). □

Lemma 5.1 (3,1)-gathering is deterministically possible under a fair centralized regular scheduler.

Proof. If the faulty robot completely disappears from the system, the problem trivially reduces to 2-gathering under a fair centralized 1-bounded scheduler. If the robots execute the following trivial algorithm: whenever chosen by the scheduler a robot moves to the location of another robot, then a terminal configuration is reached after a scheduler choice.

Assume the (n,1) system where the faulty robot is still present in the system, however each time when chosen by the scheduler it does not move. The scheduler has a fair, centralized and 1-bounded an regular behavior. That is, between two activations of a robot, each other robot is activated once. Whatever will be the initial configuration, after the choice of a correct robot the system reaches one of the following configurations: c_1 - the correct robots share the same location; c_2 a correct robot and the faulty robot share the same location. From c_1, after the move of the second correct robot the system reaches c_2. From c_2, whatever will be the scheduler choices following the bounded regular and fair centralized specification, the system converges to a terminal configuration. Hence the system converges to a terminal configuration. Once in a terminal configuration robots do not change their position. □

Lemma 5.2 There is no deterministic algorithm that solves (n,1) gathering problem, $n \geq 4$, under a fair bounded regular centralized scheduler without additional assumptions (ex. multiplicity knowledge).

Proof. If the crashed robot totally disappears from the system then the problem is trivially reduced to gathering in a fault-free system with (n-1) robots. The impossibility result directly follows from Lemma 4.1.

Assume the crashed robot is still present in the system, however each time it is chosen by the scheduler it does not execute its code. In the following we show that we can build a schedule such that the system never converges to a terminal configuration. In the sequel we distinguish two cases: systems with an even number of robots and systems with a odd number of robots.

- Odd case. Consider the following initial configuration, see Figure 7.2: the robots are arranged in two groups of quasi-equal size. More precisely, let color black the group including the faulty robot and let color white the group including only correct robots. Define a virtual matching

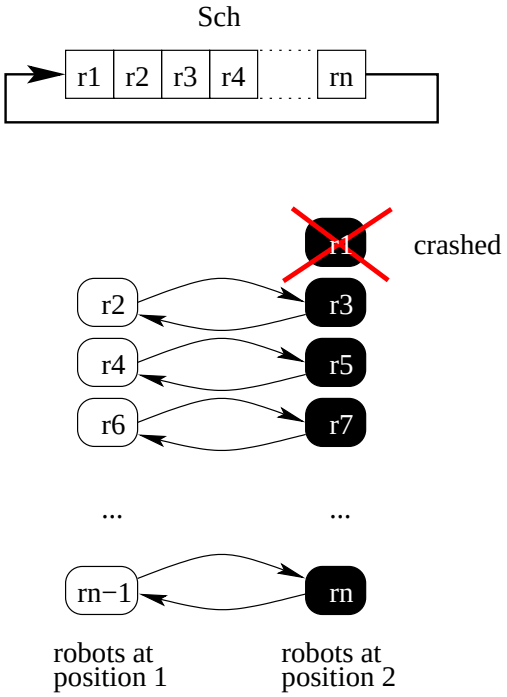

Figure 7.2: Scheduling sample.

between the correct black nodes and the white nodes such that each correct black node has as unique match a correct white robot. Define the following schedule: if a node is chosen by the scheduler in a configuration then, in the next configuration, the scheduler will chose its matching peer. We assume without restraining the generality that matchings do not change during the system execution. Moreover, since the number of nodes is odd the faulty node has no matching. From hypothesis, nodes have no additional information (ex. multiplicity knowledge). Therefore, robots whenever chosen by the scheduler move to the opposite group.

Following this schedule, after each two scheduler choices, the system reaches a symmetric configuration with respect to the initial configuration. The only difference between the two configurations being that a white and a black node have exchanged their positions. The previous scenario can be infinitely repeated. Therefore the system never converges to a terminal configuration.

- Even case. As previously, let color black the group that includes the faulty robot and let color white the group without faulty robots. Consider an initial configuration such that the two groups have equal weight. Define, a virtual matching between the black and white robots and a similar schedule as previously: each time a black robot moves its peer moves in the next configuration. In two rounds[7] the system reaches a configuration identical with the initial configuration.

Overall, the system never converges to a terminal configuration. □

[7] During a round each robot of the system is chosen by the scheduler

Lemma 5.3 There is no probabilistic algorithm that solves (n,1) gathering problem, $n \geq 3$, under a fair centralized scheduler without additional assumptions (ex. multiplicity knowledge).

Proof. Note that if the problem does not have a solution under the fair centralized scheduler then it does not have solutions under an arbitrary fair scheduler.

The scheduler may incite a correct robot to cycle between the location of another correct robot and the location of the faulty robot. Consider the following initial configuration: all correct robots are gathered at a different location than the location of the faulty robot. Without restraining the generality consider a system with three robots - 2 correct robots, r_1 and r_2, and a faulty one, r_3. Starting from this initial configuration consider the following schedule. The scheduler chooses r_1 until it is allowed to move[8]. Then, once r_1 arrives at the location of r_3, the scheduler chooses r_3. Since r_3 is the faulty node it will not move. Then the scheduler chooses r_1 until it moves back to the location of r_2. In this new configuration the scheduler repeats the same scenario with r_2. Therefore, the scheduler prevents the system to reach a terminal configuration. □

Lemma 5.4 Algorithm 4.1 is a probabilistic solution for the gathering problem in systems with n correct robots but one and under a bounded scheduler.

Proof. If the faulty robot disappears from the system then the result directly follows from Lemma 4.4 applied to a system with (n-1) robots.

Assume the faulty robot is still physically present in the system. We have to prove that with high probability all correct robots are attracted by the faulty robot. From this point further the proof is identical with the proof of Lemma 4.4. □

Lemma 5.5 There is no probabilistic and 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.

Proof. Consider without restraining the generality $f = 2$. We will prove that each of the faulty robots becomes an attraction point and the correct robots will infinitely migrate between the faulty nodes. Therefore the system never converges. Assume an initial configuration such that the faulty nodes do not share the same position and all the correct robots are quasi-equitable divided in two groups - each group including a faulty node. Let color white and black the two groups respectively. The impossibility result for the deterministic case follows using the same construction as in the proof of Lemma 5.2. In the probabilistic case, with high probability a correct robot migrates between the two groups. □

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

Proof. The idea of the proof is the following. We show that eventually a group of maximal multiplicity is created and this group further attracts all correct robots.

Assume an initial configuration such that all nodes have the same multiplicity. More precisely, nodes are evenly distributed in the plane. After the scheduler choice, the chosen robot will create with its target a group of maximal multiplicity. Let denote this group attractor. Each subsequent

[8]Since robots execute a probabilistic algorithm they might not be allowed to move by the probabilistic local computations

choice of the scheduler will increase the size of the attractor. Since the number of the correct robots is finite, eventually the system converges to a terminal configuration.

Assume that in the initial configuration all robots are organized in two or more groups with equal multiplicity. After one choice of the scheduler one of these groups increases its multiplicity, hence the above described scenario applies and the system converges to a terminal configuration. □

Lemma 5.7 Algorithm 5.2 probabilistically solves the (n,f) weak gathering problem, $f \geq 2$, under an unfair scheduler if nodes are aware of the system multiplicity.

Proof. If the system starts in a configuration with an unique group of maximal multiplicity then this group will be the attraction point for the other robots in the network. The proof is similar with the proof of Lemma 5.6. Each robot, once chosen by the scheduler, will deterministically choose as destination the group with maximal multiplicity. Since the number of robots is finite, then the system converges to a terminal configuration in a finite number of steps.

In the case when the system starts in a configuration with equal sized (multiplicity) groups, we will show that with high probability an unbalanced group is formed. Then all the other nodes will be attracted by this group. Assume without restraining the generality, the scheduler chooses a robot in each group with maximal multiplicity. Let $\alpha = \frac{1}{|maximal\_multiplicity(\mathcal{N}_p)|}$ be the probability that a robot chooses a particular target. In the following we evaluate the probability that the initial configuration does not change. With probability $s!\alpha^s$ the system reaches a new configuration which is a permutation of the old configuration (either the s robots didn't change their positions or, once all robots have reached their targets, all groups are again equilibrated). The probability that the new obtained configuration is symmetrical to the old configuration is $s!\alpha^s$. However, the probability that the previous scenario repeats as the number of scheduler choices goes to infinite is $\lim_{m\to\infty}(s!\alpha^s)^m = 0$. Eventually, with high probability the system reaches a configuration where an unique group has the maximal multiplicity. Starting from this point further the proof is similar to the previous case since nodes execute only deterministic actions. □

Lemma 5.8 There is no deterministic algorithm that solves the (3,1) gathering under a fair, centralized and bounded regular scheduler without additional assumptions.

Proof. Consider the following scheduling strategy. Every time a byzantine robot is chosen it chooses to move to an arbitrary location. In particular, it can chose to move to the previous location of the correct robot that moved in the previous round. In the following we show that the system may cycle between two non-terminal configurations. Assume 3 robots, r_1,r_2, r_3 and assume r_3 byzantine. Assume that robots initially does not share the same position. The system topology changes from the topology (r_1, r_2, r_3) to $(-, r_2, r_1r_3) \to (r_3, r_2, r_1) \to (r_3r_2, -, r_1) \to (r_1r_2r_3, -, -)$. Let's denote c the configuration corresponding to the last topology. Since r_3 is byzantine it can move and the system may reach the following topology $(r_1r_2, r_3, -)$ (robots r_1 and r_2 are gathered). If robots do not have the multiplicity knowledge they can be attracted by the byzantine node, hence the system evolves to the following topology $(r_1, r_2r_3, -)$ then $(-, r_1r_2r_3, -)$. Note that the system reached again the configuration c. Overall, the system never reaches a terminal configuration. □

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

Proof. The general proof idea is the following : the byzantine node plays the attractor role, hence the system never reaches a terminal configuration. Consider a schedule *Sch* such that after each execution of a correct robot the scheduler gives the permission to the byzantine robot to move. This schedule verifies the specification of the 2-bounded scheduler. Assume that each time a correct node chooses to move, it chooses as target the location of the byzantine node. Then, following the scheduler *Sch* the byzantine will replace the location of the node that just joined its location. Therefore, the system never converges. □

Lemma 5.10 There is no deterministic algorithm that solves (n,1)-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.

Proof. Assume by contradiction that there is an algorithm, $\mathcal{A}$, that solves the (n-1)-gathering under a k-bounded scheduler for $k \geq (n-1)$. Assume the following initial configuration: robots are organized in two groups of equal size. One of the groups contains the byzantine robot. Let refer the group containing the byzantine robot as the black group and the other group - the white group. Assume the scheduler applies iteratively the following two choices: (C1) choose a robot in the white group; (C2) choose the byzantine robot. Since $\mathcal{A}$ is a deterministic algorithm that solves gathering whenever a correct robot is chosen it moves towards another robot, otherwise the termination property is not verified. Consider in the following that the byzantine robot always moves towards the group with the weaker weight. After the choices C1 and C2 the system reaches a configuration symmetrical to the initial configuration: two groups of equal size one black and one white. In order to win the game the byzantine robot should do a move for each move of a correct robot. Therefore, for any centralized k-bounded scheduler with $k \geq (n-1)$ the byzantine robot wins and the gathering becomes impossible. □

Lemma 5.12 Algorithm 5.2, probabilistically solves the (n,f)-gathering, $n \geq 3$, problem under a bounded scheduler and multiplicity detection.

Proof. The proof is similar with the proof of Lemma 5.7. □