# Byzantine-Tolerant Circle Formation by Oblivious Mobile Robots

Samia Souissi
Graduate School of Engineering
Nagoya Institute of Technology
Gokiso, Showa-ku, Nagoya,
466-8555, Japan.
Email:souissi.samia@nitech.ac.jp

Taisuke Izumi
Graduate School of Engineering,
Nagoya Institute of Technology
Gokiso, Showa-ku, Nagoya,
466-8555, Japan.
Email: t-izumi@nitech.ac.jp

Koichi Wada
Graduate School of Engineering
Nagoya Institute of Technology
Gokiso, Showa-ku, Nagoya,
466-8555, Japan.
Email: wada@nitech.ac.jp

***Abstract*—Consider a system that consists of a group of mobile robots roaming in the two-dimensional plane. Each robot occupies a point in the plane, and is equipped with sensors to observe the positions of the other robots. Each robot proceeds by repeatedly (1) observing the environment, (2) computing a destination based on the observed positions of robots, and (3) moving toward the computed destination. Robots are unable to communicate directly, and can only interact by observing each others' positions. In addition, all robots execute the same deterministic algorithm, however some of them can be Byzantine, and exhibit arbitrary behaviors that are not in accordance with their local algorithms. Finally, robots are oblivious (i.e., stateless), meaning that they can not remember their previous observations and actions. In this model, we address the problem of coordination between these robots from a computational viewpoint, and we focus on a basic coordination problem, namely the Byzantine-tolerant circle formation problem. In other words, we study the feasibility of positioning a group of mobile robots into forming a circle in Byzantine fault model.**

**The contribution of this paper is as follows: Let $N$ be the number of robots in the system, and let $f$ be the number of Byzantine robots. We first show that there exists no algorithm that solves the Byzantine-tolerant circle formation when $N \leq 2f + 2$ in a fully synchronous system. On the other hand, when $N > 2f + 2$, we provide an algorithm in the fully synchronous model. Then, we show that when $N \leq 3f + 2$, the problem of Byzantine circle formation has no solution in the semi-synchronous model when robots' activations follow a $k$-bounded scheduler for $k \geq 2$, that is while the slowest robot is activated once, the fastest robot is activated at most $k$ times ($k$-bounded). Finally, we conjecture that the problem is impossible in general for any $N$ when $k \geq 2$ in the semi-synchronous model even for $f = 1$.**

## I. Introduction

Cooperation and coordination of autonomous mobile robots is a very important area of study since many robotic projects tend to rely more and more on complex interactions of multiple robots and sensor nodes with very limited capabilities, rather than a single general-purpose robot. A system consisting of multiple simple robots can be more cost effective and more robust. This depends heavily on the coordination mechanisms that are used, and their ability to operate properly in spite of failures. In this paper, we address the problem of coordination between these robots from a computational viewpoint, aiming to identify the fundamental limits of what autonomous mobile robots can do in the presence of Byzantine failures. In particular, we investigate the solvability of the circle formation problem when some robots may fail in a Byzantine manner. Specifically, given a group of weak mobile robots, sharing no common coordinate system, and when some of them may be Byzantine, can correct robots self-organize into forming a circle starting from any arbitrary configuration? What is the degree of synchrony required for these robots?, and what is the bound on the number of faulty robots? This problem is called the Byzantine-tolerant circle formation problem. This problem in particular has interesting applications because the ability to form a circle means that robots are spontaneously able to reach an agreement on an origin and unit distance, albeit not on a complete coordinate system. For instance, consider the context of space exploration and the initial preparation of a zone. A group of robots could be sent and after landing at random locations, would self-organize to form the initial infrastructure for later expeditions.

***Related work.*** The problem of circle formation by autonomous mobile robots was heavily studied in the literature from an algorithmic point of view [1], [5], [6], [7], [8]. However, none of these earlier studies considered the failure of robots, and this can be explained by the difficulty of handling faults by mobile robots with weak capabilities. In particular, studies on Byzantine-prone system with mobile robots considered other problems, such as the gathering problem, where robots must self-organize and meet at some location not determined in advance in finite time, and the convergence problem, which requires that all robots asymptotically approach the same location. Agmon and Peleg [4] studied the gathering problem deterministically regarding Byzantine robots in synchronous and asynchronous settings. In particular, they showed that it is impossible to perform a successful gathering in a 3-robot asynchronous system, even if at most one of them might be Byzantine. Later on, Défago et al. [9] showed the existence of randomized solutions for the gathering problem with Byzantine robots. Bouzid et al. [10] proposed an algorithm for the convergence problem in one dimensional space with oblivious robots. Their algorithm tolerates $f$ Byzantine robots for $(2f + 1)$-sized robot networks in the fully synchronous model, and $(3f + 1)$-sized in the semi-synchronous model. In their later work [11], they proposed an optimal algorithm for

the asynchronous case, with an optimal bound of $3f+1$ that works under bounded scheduling assumptions.

To the best of our knowledge, our work is the first to consider the solvability of pattern formation problems in the presence of Byzantine faulty robots, and specifically, the circle formation problem. Also, we want to stress that previous algorithms proposed in the literature for the circle formation problem do not work properly in the presence of faulty robots, and their adaptation is not straightforward.

***Contribution.*** This paper studies the feasibility of circle formation problem by oblivious robots when they may fail in a Byzantine manner, under the fully synchronous and semi-synchronous models. First, we show that the problem is not solvable in the fully synchronous model for $N \leq 2f+2$ robots, where $f$ represents the number of Byzantine faults. Then, we give a solution to the Byzantine circle formation for $N > 2f+2$ synchronous robots. Finally, we show that when $N \leq 3f+2$, the problem has no solution in the semi-synchronous model under a $k-$bounded scheduler for $k \geq 2$, and we conjecture that the problem is impossible in general for any $N$ when $k \geq 2$ in the semi-synchronous model.

***Structure.*** The remainder of this paper is structured as follows. In Section II, we introduce the system model and the problem definition. In Section III, we show the impossibility of Byzantine-tolerant circle formation for $N \leq 2f+2$ in the fully synchronous model. In Section IV, we provide an algorithm that solves the problem for $N > 2f+2$ synchronous robots, and prove its correctness. In Section V, we show that the problem has no solution in the semi-synchronous model for $N \leq 3f+2$ and $k \geq 2$, when robots' activations follow a $k-$bounded scheduler. Finally, in Section VI, we conclude the paper.

## II. Preliminaries

### A. Robot Models:

The system consists of a set of autonomous mobile robots $\mathcal{P} = \{r_1, \cdots, r_n\}$ roaming on the two-dimensional plane devoid of any landmark. Each robot is modelled and viewed as a point in the plane, and equipped with sensors to observe the positions of the other robots. Each robot executes its own instance of the same algorithm which consists of repeatedly (1) observing the environment, (2) computing a destination, and (3) moving toward it. This is called the *cycle* of a robot. The system can be classified into three synchronization models depending on the activation of robots, and the time at which they see the environment.

***Fully-synchronous model***. The robots are fully-synchronous, in the sense that all of them are always activated at the same time, and also they observe the environment simultaneously. In other words, robots operate according to the same clock cycles. This model is the strongest one.

***Semi-synchronous*** [3]. This model is similar to the fully-synchronous model in the sense that all robots operate according to the same clock cycles, however not all robots are necessarily activated at each cycle. In other words, when robots are activated simultaneously, they see the same thing on the environment. Otherwise, robots can see the environment only when the other robots has already finished their moves.

***Asynchronous*** [2]. In this model, robots are completely asynchronous in the sense that no bounds exist on the length of their cycles, but finite.

In the semi-synchronous, and asynchronous models, the activation of robots is done through a *scheduler* that ensures the fairness of activation of robots. That is, every robot becomes active at infinitely-many times. In particular, we introduce the definition of $k$-bounded scheduler as follows:

*$k$-bounded* scheduler: between any two consecutive activations of a robot, no other robot is activated more than $k$ times.

In the three robot models, robots are *anonymous*, in the sense that they can not be distinguished by their appearance, and they do not have any kind of identifiers that can be used during their computation. In addition, there is no direct means of communication among them. Hence, the only way for robots to acquire information is by observing each other's positions. In this paper, we make the following further assumptions on robot models:

- *Oblivious*: robots are oblivious or *memory-less*, which implies that they are unable to remember their past actions and observations, and thus their computations can not be based on previous observations.
- *No common coordinate system*: robots share neither knowledge of the coordinate systems of the other robots, nor of a global coordinate one. However, robots agree on the clockwise direction.
- *Byzantine failures*: we assume that robots can be Byzantine. A Byzantine robot exhibits arbitrary behavior, and executes arbitrary steps that are not in accordance with its local algorithm. This behavior in general is modelled by means of an *adversary*, which has the ability to control the behavior of faulty and non-faulty robots. We also assume that robots can not distinguish Byzantine robots from correct ones (non faulty robots).

Finally, we assume that initially no two correct robots can be on the same location, and this is because it is impossible to separate them later deterministically since they are oblivious.

### B. Problem Definition

The problem addressed in this paper is the circle formation problem under the Byzantine fault model by a group of oblivious mobile robots. The problem is defined more rigorously as follows:

*Definition 1 ($(N,f)$-Byzantine-tolerant Circle Formation): Given a group of $N$ robots $r_1, r_2, \ldots$ and $r_N$, including at most $f$ Byzantine robots. These robots are located arbitrarily on the plane. We say that robots solve the $(N,f)$-Byzantine-tolerant circle formation problem, if at least $(N\text{-}f)$ correct robots arrange themselves at distinct positions to eventually form a non-degenerate circle (i.e., with finite radius greater than zero), and this circle is unique. Besides, after forming the circle, it remains unchanged forever. We shall call this configuration a legitimate configuration.*

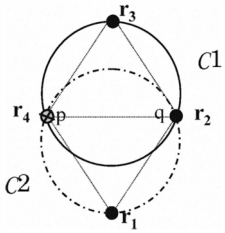



Fig. 1. Illustration: Impossibility proof of Theorem 1; e.g., $(N = 4, f = 1)$.

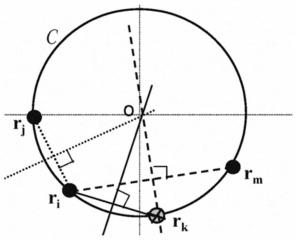



Fig. 2. Principle of Algorithm 1: preservation of circle $\mathcal{C}$ by correct robots.

When no ambiguity arises, we sometimes use the term legitimate for circle. For instance, when we say circle $\mathcal{C}$ is legitimate means that $\mathcal{C}$ contains all correct robots on its boundary. The problem of Byzantine-tolerant circle formation is trivial for cases where there are only one or two robots on the system. Therefore, in the rest of the paper we consider the cases with three or more robots.

## III. Impossibility of Byzantine-tolerant circle formation under the fully-synchronous model for $N \leq 2f + 2$

In this section, we show that the Byzantine-tolerant circle formation problem has no solution in the fully synchronous model for $N \leq 2f + 2$ oblivious robots. The proof is by contradiction. Assume that there exists an algorithm $\mathcal{A}$ that solves the problem. The proof consists of considering some configuration $\mathbb{E}$, such that $\mathbb{E}$ becomes legitimate for some choice of Byzantine robots, and $\mathbb{E}$ is not legitimate for some other choice of Byzantine robots. However, algorithm $\mathcal{A}$ stops all robots from moving, and thus it does not allow the robots to reach a legitimate configuration.

Assume that the initial configuration is a bivalent configuration. That is, there exist two circles $\mathcal{C}_1$ and $\mathcal{C}_2$ that have the same diameter, and the boundaries of these two circles intersect into two points $p$ and $q$ that are occupied by some robots. Also, $\mathcal{C}_1$ and $\mathcal{C}_2$ are completely symmetrical, in the sense that they have the same number of robots on the boundary, which is equal to $f + 2$ (with two robots are in common), and they are images of each other with respect to the line $(\overline{pq})$ (Fig. 1).

We first prove the case of $N = 2f + 2$ (e.g., $N = 4, f = 1$ in Fig. 1). First, consider that the robots that are located outside of $\mathcal{C}_1$ are Byzantine (their number is equal to $f$), then circle $\mathcal{C}_1$ becomes a legitimate configuration. Therefore, by algorithm $\mathcal{A}$, none of the robots located on the boundary of $\mathcal{C}_1$ move because they are in legitimate configuration. Similarly, consider the situation where the robots that are located outside of $\mathcal{C}_2$ are Byzantine. Then, circle $\mathcal{C}_2$ also becomes a legitimate configuration. This means that, none of the robots located on the circumference of $\mathcal{C}_2$ move by algorithm $\mathcal{A}$. It follows that, none of the robots move by algorithm $\mathcal{A}$ because they believe they have reached the legitimate configuration.

Now, assume that robots that are located outside of $\mathcal{C}_1$, and also the ones located outside of $\mathcal{C}_2$ are correct, then neither $\mathcal{C}_1$ nor $\mathcal{C}_2$ is a legitimate configuration because these robots do not belong to the same circle, however algorithm $\mathcal{A}$ stops all robots from moving. A contradiction. Thus, algorithm $\mathcal{A}$ does not exist, and the case of $N = 2f + 2$ is proved. The proof for $N < 2f + 2$ robots can be deduced easily from above, and thus omitted here. Consequently, the following theorem holds:

*Theorem 1: In the fully synchronous model, there exists no oblivious algorithm that solves the $(N, f)-$Byzantine-tolerant circle formation problem deterministically for $N \leq 2f + 2$ robots, with at most $f$ Byzantine failures.*

*Corollary 1: In the semi-synchronous, and asynchronous models, there exists no oblivious algorithm that solves the $(N, f)-$Byzantine-tolerant circle formation problem deterministically for $N \leq 2f+2$ robots, under a $k-$bounded scheduler, even for $k = 1$.*

**Algorithm 1: Byzantine-tolerant circle formation for $N > 2f + 2$ synchronous robots (code executed by robot $r_i$);**

**Begin**

1: $P :=$ the set of positions of all robots on the plane;
2: $\Delta_1, \Delta_2 :=$ perpendicular bisector of the segment $\overline{r_j r_k}$, and respectively $\overline{r_j r_m}$ formed by any three robots $r_j$, $r_k$, and $r_m$;
3: **if** $(\Delta_1 \cap \Delta_2 \neq \emptyset)$ **then**
4: $r_j$, $r_k$, and $r_m$ belong to the same circle $\mathcal{C}$, having as center the intersection point of $\Delta_1$ and $\Delta_2$;
5: **end if**
6: **if** ($\exists$ unique circle $\mathcal{C}$, such that the maximum number of robots are on $\mathcal{C}$, and they form a majority)
7: Function_Move_to_Circle($\mathcal{C}$,$P$,$r_i$);
8: **else** [target circle is the smallest enclosing circle]
9: $r_i$ computes the smallest enclosing circle $\mathcal{SEC}$ of points $P$;
10: $\mathcal{C} := \mathcal{SEC}$
11: Function_Move_to_Circle($\mathcal{C}$, $P$,$r_i$);
12: **end if**

**end**

## IV. Byzantine-tolerant Circle Formation Algorithm in the Synchronous model for $N > 2f + 2$ Robots

### A. Algorithm Description

Informally, the idea of the algorithm is as follows. First, when robots get activated, they check if there exists a unique circle $\mathcal{C}$, where the majority of robots are located on it. We mean by the majority of robots are located on $\mathcal{C}$ that the number of robots located on the boundary of $\mathcal{C}$ is strictly greater than $N/2$, and they form the *maximum* number of robots on the boundary of $\mathcal{C}$. Then, if such a circle $\mathcal{C}$ exists,

**Function_Move_to_Circle($\mathcal{C}$, $P$,$r_i$)**

**Begin**

1: $o$ := center of $\mathcal{C}$ ;
2: $N := |P|$;
3: **if** ($r_i = o$)
4: $r_i$ chooses arbitrary to belong to a radius, where at least one robot is located on that radius;
5: **end if**

6: **if** ($r_i \in boundary(\mathcal{C})$)
7: Do_Nothing();
8: **elseif** (only robot $r_i$ is located on the ray $[\overline{or_i})$ and $r_i \neq o$)
9: $r_i$ moves on the ray $[\overline{or_i})$ toward $boundary(\mathcal{C})$ ;
10: **else** [some other robots are located on the same ray radius as $r_i$]
11: $K$ := the number of robots located on the ray $[\overline{or_i})$, including the robots at the center of $\mathcal{C}$ if any;
12: $\Psi$ := the axis collocated with the ray $[\overline{or_i})$, oriented from $o$ toward the locations of robots;
13: $next_{r_i}$:= the next direct neighbor of $r_i$, which is not located on the ray $[\overline{or_i})$, in clockwise direction;
14: $\alpha_{next_{r_i}}$:= angular distance between $r_i$ and $next_{r_i}$ in clockwise orientation;
15: $\theta := \alpha_{next_{r_i}} / K$;
16: $Stargets$ := the set of $K$ targets on the boundary of $\mathcal{C}$, such that the angular distance between any two consecutive targets is $\theta$, starting from $\Psi$, and in clockwise direction;
17: Robot $r_i$ ranks the set of $Stargets$ with respect to $\Psi$;
18: Robot $r_i$ ranks the set of $K$ robots located on the ray radius $[\overline{or_i})$, with respect to $\Psi$ (Fig. 3(b));
19: $r_i$ moves to the target on the boundary of $\mathcal{C}$ that corresponds to its rank;
20: **end if**

**end**

robots that are not located on the circumference of $\mathcal{C}$ move to it without collisions with the other robots. However, if no circle with the majority of robots exist, then robots compute the smallest enclosing circle $\mathcal{SEC}$ of the observed positions of all robots on the system, and move to $\mathcal{SEC}$.

In the algorithm, we assume that a robot teleport precisely to its computed target within one cycle without stopping on the way. Such assumption is essential because removing it renders the problem very difficult to solve even in the fully synchronous model because robots are oblivious, they can not distinguish Byzantine robots from correct ones, and also the number of Byzantine robots $f$ is not known to the robots. Subsequently, if robots do not reach their destination in the target circle within one cycle, it is very difficult for correct robots to fix a target circle because the movement of Byzantine robots can change that circle. In other words, removing such assumption makes the smallest enclosing circle $\mathcal{SEC}$ useless, since it is not guaranteed that correct robots reach the boundary of $\mathcal{SEC}$ within one single cycle, and then Byzantine robots can modify $\mathcal{SEC}$ in each activation cycle, and thus correct robots will never reach $\mathcal{SEC}$ because they are oblivious.

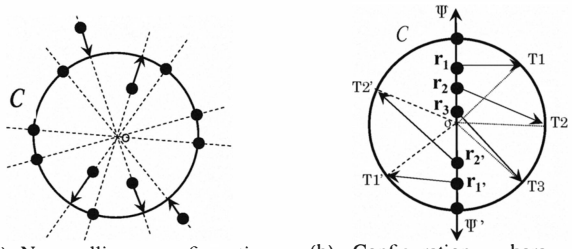
The pseudo code description of the algorithm is given in Algorithm 1, where the Function_Move_to_Circle($\mathcal{C}$, $P$,$r_i$) gives to each robot its target destination on the circle $\mathcal{C}$. We first describe how robots compute if the majority of robots are located on the same circle $\mathcal{C}$. In other words, how they compute and preserve $\mathcal{C}$ if it exists (Algorithm 1). Then, we describe how robots reach the circumference of the circle without collisions with other robots ($Function\_Move\_to\_Circle(\mathcal{C}, P, r_i)$).

Assume that $r_i$ is activated at some time $t$, then $r_i$ computes with every robot on the plane the perpendicular bisector between its position and the position of the other robot. If the perpendicular bisector between the pair of robots $(r_i, r_j)$ intersects the perpendicular bisector of a pair of robots $(r_i, r_k)$, then these three robots $r_i$, $r_j$ and $r_k$ are located on the same circle having as a center the intersection of their perpendicular bisectors (refer to Fig. 2). In order to compute the circle with the majority of robots on boundary, robot $r_i$ has to compute the perpendicular bisector between any pair of robots on the system, and find the maximum number of robots, in which their perpendicular bisectors intersect on the same point. If the number of robots that their perpendicular bisectors intersect on the same point $I$ exceeds $N/2$ robots, and $I$ is unique, then there exists a unique circle $\mathcal{C}$ having as center the point $I$, where the majority of robots are located on its boundary. We now describe how robot $r_i$ computes its destination on the target circle $\mathcal{C}$ (having as center $o$). $\mathcal{C}$ is either the circle where the majority of robots are located on its boundary, or the smallest enclosing circle $\mathcal{SEC}$. We distinguish the following cases:

1) If robot $r_i$ is located on the circumference of $\mathcal{C}$, then $r_i$ does not move.
2) If robot $r_i$ is located at the center of the target circle $\mathcal{C}$, then $r_i$ chooses arbitrary to belong to one radius of $\mathcal{C}$ where at least one robot is located on that radius. This is needed in order to avoid collisions between robots when they are moving toward $\mathcal{C}$.
3) If robot $r_i$ is located in one separate radius, that is only robot $r_i$ is located on the ray $[\overline{or_i})$, then $r_i$ simply moves linearly on the ray $[\overline{or_i})$ to the boundary of $\mathcal{C}$ (Fig. 3(a)).
4) If robot $r_i$ is collinear on the radius with other robots (Fig. 3(b)), then $r_i$ computes $next_{r_i}$; the next direct neighbor to $r_i$ in clockwise direction, such that $next_{r_i}$ is not located on the same radius as $r_i$. The target of $r_i$ on $\mathcal{C}$ (also the targets of the robots located on the same radius as $r_i$) will be within the angular sector of $\mathcal{C}$ delimited by $r_i$ and $next_{r_i}$. Let $\Psi$ be the ray collocated with the radius of $r_i$ oriented from the center $o$ toward the location of $r_i$, and let $\alpha_{next_{r_i}}$ be the angular distance between $r_i$ and $next_{r_i}$ in clockwise orientation. All robots located on $\Psi$ will be ranked according to $\Psi$, and also their targets. Let $Stargets$ be the set of targets on $\mathcal{C}$ of robots located on $\Psi$. $Stargets$ are computed starting from $\Psi$, and in clockwise direction, such that the angular distance between any two consecutive targets is equal to $\alpha_{next_{r_i}}$ divided by the number of robots on $\Psi$. $Stargets$ will be also ranked according to $\Psi$, and robot $r_i$ moves to the target on $Stargets$ that corresponds to its rank on $\Psi$.

### B. Correctness

*Lemma 1: By Algorithm* 1, *no two correct robots ever move to the same location.*




(a) Non collinear configuration of robots.

(b) Configuration where some robots are located on the same radius.

Fig. 3. Computation of robots' targets on circle.

*Proof:* The proof is straightforward. First, in non collinear configurations, robot $r_i$ is located in one separate radius of the target circle $\mathcal{C}$, and its move is linearly on its radius toward the boundary of $\mathcal{C}$, so none of the other robots intersect the path of robot $r_i$, and then it can not collide with any of the other correct robots.

In the case where some robots are located on the same radius, Algorithm 1 gives a total ordering to these robots, and also to their targets on $\mathcal{C}$. In addition, by assumption each robot must reach its target in one cycle, and robots are synchronous, then in finite time, each correct robot will be located at its final target, and thus no two correct robots move to the same location.[1] ■

*Lemma 2: By Algorithm* 1, *all correct robots reach the circumference of the same circle in a finite number of steps.*

*Proof:* We first show that all correct robots compute the same target circle $\mathcal{C}$. By assumption, all correct robots are activated together, and they execute the same algorithm, then if there exists a unique circle $\mathcal{C}$, where the majority of robots are on its boundary, then necessarily correct robots compute the same circle $\mathcal{C}$. Otherwise, all correct robots compute the smallest enclosing circle of the positions of all robots in the system, which is also unique.

We now show that all correct robots reach the circumference of $\mathcal{C}$ in one single step. By Algorithm 1, a robot $r_i$ that is located on the boundary of $\mathcal{C}$ does not move, while a robot $r_i$ that is located inside or outside $\mathcal{C}$ has a unique target on the boundary of $\mathcal{C}$. Since, the cycle of a robot is finite, robots are synchronous, and by assumption each robot has to reach its target within one cycle, then all correct robots reach the circumference of $\mathcal{C}$ in one single step. ■

*Lemma 3: By Algorithm* 1, *the target circle* $\mathcal{C}$ *computed by correct robots is stable or invariant.*

*Proof:* Assume at some time $t$, robots compute the target circle $\mathcal{C}$, then by Algorithm 1, $\mathcal{C}$ is unique at time $t$ because either $\mathcal{C}$ is the smallest enclosing circle $\mathcal{SEC}$, which is unique at time $t$, or $\mathcal{C}$ is the unique circle with majority of robots on boundary at time $t$ ($\mathcal{C}$ may contain correct and Byzantine robots).

We will also show that $\mathcal{C}$ is unique at any time $t' \geq t$. First, by Lemma 2, all correct robots reach the circumference of $\mathcal{C}$ in one step. Then, at time $t+1$, $\mathcal{C}$ becomes the *unique* circle having all correct robots on its boundary that form the maximum, and also the majority of robots.

Now, we will show that Byzantine robots can not create another circle $\mathcal{C}'$ that contains a maximum of robots on its boundary. First, if $\mathcal{C} \cap \mathcal{C}' = \emptyset$, then the number $(N-f)$ of correct robots is greater than $f$ Byzantine robots because $N > 2f+2$. Second, if $\mathcal{C} \cap \mathcal{C}' = \{p, q\}$, where the points $p$ and $q$ are occupied by correct robots, then $(N-f) > (f+2)$ robots because $N > 2f+2$. Finally, when $\mathcal{C} \cap \mathcal{C}' = \{p\}$, the same arguments holds. So, $\mathcal{C}'$ does not exist, and $\mathcal{C}$ is unique.

Finally, we show that $\mathcal{C}$ is stable or invariant at any time $t' \geq t+1$. By the algorithm, correct robots that are located on the boundary of $\mathcal{C}$ at time $t+1$ can compute the same circle $\mathcal{C}$ at any time $t' \geq t+1$ because the intersection of the perpendicular bisectors between any pair of correct robots gives the same point, which is the center of $\mathcal{C}$. In addition, by the algorithm, correct robots on the boundary of $\mathcal{C}$ do not leave $\mathcal{C}$ because they form the circle with the maximum and majority of robots on boundary. So, the movement of correct robots do not change $\mathcal{C}$. Also, as we showed earlier, the movement of Byzantine ones can not change $\mathcal{C}$. It follows that, $\mathcal{C}$ is unique, and stable. ■

From Lemma 1, Lemma 2, and Lemma 3, we derive the following theorem:

*Theorem 2: In a Byzantine-prone system, and starting from any initial configuration, Algorithm* 1 *is a fault-tolerant oblivious algorithm that solves the* $(N, f)$*-Byzantine-tolerant circle formation for* $N > 2f+2$ *synchronous robots, deterministically.*

## V. Impossibility of Byzantine-tolerant circle formation in the semi-synchronous model for $N \leq 3f+2$ under $k-$bounded scheduler for $k \geq 2$

In this section, we study the feasibility of the $(N, f)-$Byzantine tolerant circle formation in the semi-synchronous model deterministically, under a $k-$bounded scheduler. In particular, we show that it is impossible to solve the problem in the semi-synchronous model for $N \leq 3f+2$ oblivious robots when $k \geq 2$.

[1]Note that, while moving to their targets, two robots may pass by the same point on the plane, however they never end their moves on the same target location by the algorithm since by assumption on the system model, the movement of a robot is instantaneous.

*Theorem 3: In the semi-synchronous model, there exists no oblivious algorithm that solves the $(N, f)$-Byzantine-tolerant circle formation problem deterministically when $N \leq 3f + 2$ robots, for at most $f$ Byzantine failures under a $k-$bounded scheduler when $k \geq 2$.*

The idea of the proof is similar to the proof for fully synchronous model given in Section III. Consider a bivalent configuration, where we have two circles $\mathcal{C}_1$ and $\mathcal{C}_2$ that have the same diameter, and these circles intersect into two points $p$ and $q$ that are occupied by some robots. Also, $\mathcal{C}_1$ and $\mathcal{C}_2$ are completely identical in the sense that they have the same number of robots on boundary, which is equal to $f + 2$, and they are images of each other with respect to the line $(\overline{pq})$.

Initially, let the $f$ Byzantine robots be located anywhere on the plane. Let us consider the behavior of algorithm $\mathcal{A}$ given the following scenario and an activation scheduler for $k = 2$. Assume at time $t_0$, the adversary activates the Byzantine robots, and make them move to $\mathcal{C}_1$ or $\mathcal{C}_2$. Assume without loss of generality that at time $t_0$, the $f$ Byzantine robots move to the boundary of $\mathcal{C}_1$. As a result, at time $t_1$, we have $2f + 2$ robots located on the boundary of $\mathcal{C}_1$. Since, the total number of robots on the system is $3f + 2$, then only $(3f + 2) - (2f + 2) = f$ robots are not placed on the boundary of $\mathcal{C}_1$. Consequently, $\mathcal{C}_1$ becomes a legitimate configuration at time $t_1$. Now, at time $t_1$, the adversary activates all correct robots that are located on the boundary of $\mathcal{C}_1$. By algorithm $\mathcal{A}$, none of the correct robots on the boundary of $\mathcal{C}_1$ move because they believe they are in the legitimate configuration.

After that, at time $t_2$, the adversary activates again the $f$ Byzantine robots, and make them move to the boundary of the second circle $\mathcal{C}_2$. Assume at time $t_3$, these robots finish their moves. Then, at time $t_3$, we will have $2f + 2$ robots located on the boundary of $\mathcal{C}_2$, and $f$ robots outside of it. Then, $\mathcal{C}_2$ becomes a legitimate configuration. Similarly, at time $t_4 > t_3$, the adversary activates all correct robots that are located on the circumference of $\mathcal{C}_2$, and by the same arguments, none of them move. The adversary can reiterate the above scenario infinitely. Consequently, all correct robots never move by algorithm $\mathcal{A}$. This leads to a contradiction because neither $\mathcal{C}_1$ nor $\mathcal{C}_2$ is legitimate (correct robots are not located on the same circle). However, $\mathcal{A}$ stops all correct robots from moving. The same result holds when $k > 2$. Thus, algorithm $\mathcal{A}$ does not exist. The following result is derived:

*Corollary 2: In the asynchronous model, and Byzantine-prone system, there exists no oblivious deterministic algorithm that solves the circle formation problem for $N \leq 3f+2$ robots, under a $k$-bounded scheduler when $k \geq 2$.*

## VI. Conclusion

In this paper, we studied the solvability of the circle formation problem deterministically in Byzantine environment, and with oblivious robots. Table I summarizes the results of this paper. In particular, we have shown that the problem has no solution in the fully synchronous model for $N \leq 2f + 2$. Subsequently, we have proposed an algorithm that allows the robots to form the circle when $N > 2f + 2$. Then, we have shown that it is impossible to solve the Byzantine circle formation in the semi-synchronous model for $N \leq 3f + 2$ robots under a $k-$bounded scheduler when $k \geq 2$. The question remains open in the semi-synchronous model when $N > 3f+2$. We conjecture that the Byzantine circle formation problem is impossible under a $k-$bounded scheduler for $k \geq 2$ even when $f = 1$, and this is due to the fact that two competitive circles, in which correct robots are located can not be transformed to one unique circle in finite time because when $k \geq 2$, it is easy for the adversary to make Byzantine robots alternate between the two competitive circles, in such a way to stop correct robots from moving. However, we believe that the convergence problem can be solvable. Currently, we are investigating these issues.

TABLE I
Solvability of the circle formation in Byzantine prone system.

| Model | Condition | Solution |
|---|---|---|
| **Fully-synchronous** | $N \leq 2f + 2$ | Impossible (Section III) |
| **Fully-synchronous** | $N > 2f + 2$ | Possible (Section IV) |
| **Semi-synchronous** | $N \leq 3f + 2$ | Impossible for $k \geq 2$ (Section V) |
| **Semi-synchronous** | $N > 3f + 2$ | (Open question) |
| **Asynchronous** | $N \leq 2f + 2$ | Impossible (deduction) |
| **Asynchronous** | $N \leq 3f + 2$ | Impossible for $k \geq 2$ (deduction) |

## References


[1] Défago, X., Souissi, S.: Non-uniform circle formation algorithm for oblivious mobile robots with convergence toward uniformity. Theor. Comput. Sci., 396, (1-3) (2008) 97–112

[2] 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), (2001) 185–190.

[3] I. Suzuki and M. Yamashita, Distributed Anonymous Mobile Robots: Formation of Geometric Patterns, SIAM Journal of Computing, No. 28 4 (1999), 1347–1363.

[4] N. Agmon and D. Peleg, Fault-Tolerant Gathering Algorithms for Autonomous Mobile Robots, SIAM Journal of Computing, No. 36 1 (2006) 56-82.

[5] Y. Dieudonne and O. Labbani-Igbida and F. Petit, Circle Formation of Weak Mobile Robots, Proc. of 8th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS'06), LNCS. No. 4280 (2006), 262–275.

[6] Y. Dieudonné and F. Petit, Circle Formation of Weak Robots and Lyndon Words, Information Processing Letters, No. 104 4 (2007), 156–162.

[7] Y. Dieudonné and F. Petit, Swing Words to Make Circle Formation Quiescent, Forteenth International Colloquium on Structural Information and Communication Complexity (SIROCCO '07), No. 4474 (2007) 166–179.

[8] B. Katreniak, Biangular Circle Formation by Asynchronous Mobile Robots, Proc. 12th Intl. Colloquium on Structural Information and Communication Complexity (SIROCCO'05), Le Mont St-Michel, France, (2005).

[9] X. Défago, M. Gradinariu, S. Messika P. Raipin-Parvédy, Fault-Tolerant and Self-stabilizing Mobile Robots Gathering, Proc. 20th International Symposuim on Distributed Computing (DISC'06), 4167 (2006) 46–60

[10] Z. Bouzid, M. G. Potop-Butucaru and S. Tixeuil, Byzantine-Resilient Convergence in Oblivious Robot Networks, Proc. 10th International Conference on Distributed Computing and Networking (ICDCN'09), LNCS. No. 5408 (2009), 275–280

[11] Z. Bouzid, M. G. Potop-Butucaru and S. Tixeuil, Optimal Byzantine Resilient Convergence in Asynchronous Robot Networks, Proc. of 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS'09), LNCS. No. 5873 (2009), 165–179