Direction Election in Flocking Swarms*

(Extended Abstract)

Ohad Ben-Shahar† Shlomi Dolev† Andrey Dolgin†‡ Michael Segal‡

ABSTRACT

Swarm gathering and swarm flocking may conflict each other. Without explicit communication, such conflicts may lead to undesired topological changes since there is no global signal that facilitates coordinated and safe switching from one behavior to the other. Moreover, without coordination signals multiple swarm members might simultaneously assume leadership, and their conflicting leading direction is likely to nullify a successful flocking effort. To the best of our knowledge, we present the first set of swarm flocking algorithms that maintain connectivity while electing direction for flocking. The algorithms allow spontaneous direction requests and support direction changes.

Categories and Subject Descriptors

F.2.2 [**Nonnumerical Algorithms and Problems**]: General—*Sequencing and scheduling*

General Terms

Algorithms

Keywords

Direction Election, Flocking, Swarms, Mobile Robots

1. INTRODUCTION

Coordinating motion, cooperative formation, and flocking control of multiple autonomous entities is of great theoretical and practical interest. One (now classical) approach is Reynolds's Boids [17], where each entity updates its movement based on the distances and velocities of neighboring entities in order to ensure the *alignment* and *cohesion* of the swarm while avoiding unsafe distances between the entities. The three corresponding behavioral rules that each entity follows indeed address many practical situations, however there are still several pathological cases where partition of the swarm is possible. For example, if it happens that all entities move exactly towards (or away from) their center of mass, no convergence to stable flocking can occur and it becomes impossible to break the symmetry without using randomization. In this paper, an intrinsic bounded random variable is indeed incorporated, though its primary use goes beyond symmetry breaking, handling leadership election when no explicit information exchange is allowed between the swarm's entities.

Clearly, convergence of a swarm motion while avoiding topological changes like partitioning is possible using explicit (e.g., wireless) communication, and indeed early work has explored such solutions, in particular when the communication is employed in a predefined time interval [8, 18, 24]. Later work relaxed the defined time interval to variable-length intervals [7, 11, 13] or employed probabilistic networks [22]. However, unlike previous work, we seek to solve these problems in "silent networks", where there is no explicit communication between the swarm members. Furthermore, unlike much of the previous work, we are also interested in maintaining robust convergence results despite uncertainties in both the measurement and execution of motion. All these conditions mimic constraints of artificial swarms.

Much work on the stability and convergence of flocking in silent swarms has been based on the use of *potential functions*. Doing so, several algorithms and convergence results have been obtained for different flocking mechanisms and various connectivity assumptions (e.g., [14, 15, 25, 9]). While most effort was put to investigate leaderless swarms, some research has extended this scope, presenting potential field algorithms for safe, connectivity-preserving flocking in the presence of a single leader. However, thus far no potential function, that guarantees the desired flocking behavior (and in particular, collision avoidance and connectivity preservation) when multiple leaders may co-exist, is known. Still, the case of multiple leaders co-existence has been considered in several works, typically employing local rules and information passing between neighbors [19, 10]. Only several recent papers attack the problem in the scope of "silent networks". For example, Jiang-Ping and Hai-Wen [10] show that all the agents will flock in the polytope region formed by the leaders. Su et al. [20] have studied the case of a fixed

*Research supported in part by the ICT Programme of the European Union under contract number FP7-215270 (FRONTS), US Air Force European Office of Aerospace Research and Development, grant number FA8655-09-1-3016, Microsoft, Deutsche Telekom, European project FLAVIA, Israeli Ministry of Industry, Trade and Labor (consortium CORNET), and Rita Altura Trust Chair in Computer Sciences.

†Department of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva, 84105, Israel. Email: ben-shahar@cs.bgu.ac.il, dolev@cs.bgu.ac.il, dolgina@bgu.ac.il.

‡Department of Communication Systems Engineering, Ben-Gurion University of the Negev, Beer-Sheva, 84105, Israel. Email: dolgina@bgu.ac.il, segal@cse.bgu.ac.il.

DIALM-POMC'10, September 16, 2010, Cambridge, MA, USA.
ACM 978-1-4503-0413-9/10/09.

set of leaders and showed how the swarm will steadily converge to the dynamical center of mass of the leaders. While suitable under specific assumptions (e.g., symmetry does not exist) for solving different agreement problems like swarming, schooling, flocking, or rendezvous, existing methods do not support direction election. Furthermore, the majority of previous works bear on a strict assumption that connectivity is preserved and collisions between entities are avoided all the time, while no technique to obey these properties is described.

An important issue related to flocking swarms is their temporal convergence rate to a stable configuration. Results on this problem have been obtained by Chazelle in [1, 2], who showed an exponential time bound for the convergence of the motion of a *single entity* to a fixed motion vector. Furthermore, the time bound for the convergence of the entire swarm was shown to be an iterated exponential of height logarithmic in the number of agents.

In this paper we present (what we believe are) the first practical and provable flocking schemes with silent direction election from several candidates. We develop simple and efficient algorithms for silent direction election, taking into account (bounded) environmental and parametric uncertainties, while providing a mechanism for connectivity preservation and collision avoidance. These latter capacities are obtained by introducing the notion of a *spring*, which resembles a potential function with restrictions and provides flexibility in the presence of uncertainties. The same mechanism also withstands temporary coexistence of multiple leaders. We do note that although direction election techniques do exist in the context of mobile robotics [6, 12], these methods assume explicit communication between entities, which is outside the scope of our study.

The rest of this paper is organized as follows. Section 2 briefly describes our system settings. Section 3 describes the intuition behind our new spring network approach and proves its effectiveness for direction election in autonomous mobile swarms under uncertain environment, assuming initial connectivity of swarm entities. Conclusions appear in Section 4. Some details are omitted from this preliminary extended abstract.

2. SWARM SETTINGS

We begin our theoretical discussion with several definitions and notions, the first of which relates to the different swarms configurations that may be considered in the context of flocking. In particular, we will first consider *synchronous swarms*, where all entities perform their measurements at the same points in time based on one global clock. We will then move to discuss *partially asynchronous swarms*, which relax the synchronous swarms assumption to allow each entity to perform its measurements and motion updates after some arbitrary time phase relative to global time pulses (as the local clocks measure time periods in the same rate but are not synchronized to hold the same global time). We note that the case of asynchronous network, where each entity has its own independent clock and cycle duration, is not considered here.

DEFINITION 2.1 (CYCLE). *Cycle duration dt is a unit of time all entities use for their measurement and position adjustments.*

As we emphasized in the Introduction, the focus of this paper is *silent swarms*, i.e. network of entities *without* explicit communication of information transfer. In particular, the only measurement allowed for different entities is the position of neighboring entities. Let r_i be the position vector of entity i. We therefore define:

DEFINITION 2.2 (DISTANCE). *Let r_i and r_j be the position vectors of entities i and j, respectively. The distance r_{ij} between the two entities is therefore defined as the Euclidean norm of the corresponding difference vector $r_{ij} = ||r_i - r_j||$.*

To support practical applications, we do allow some uncertainty in distance measurements and the execution of motion commands, and in particular, we assume that distance measurement errors (compared to the true distances) are bounded by a known constant, which includes a symmetry breaking randomization as well. Formally,

DEFINITION 2.3 (ERROR). *Error e_{ij} is the difference between the measurement made by entity i for the distance to its neighbor j and their actual distance. i.e. $e_{ij} = r_{ij}^{m_i} - r_{ij} + RV$, where RV stands for a random variable value, bounded by a predetermined constant, which is used to break possible pathological symmetry position of entities. The error is bounded by some known constant e, i.e. $|e_{ij}| \leq e \quad \forall i, j$.*

Due to the measurement errors, the measurement made by entity i for the distance to its neighbor j may be different from the real value r_{ij}

DEFINITION 2.4 (MEASURED DISTANCE). *Let $r_i^{m_i}$ and $r_j^{m_i}$ be the position vectors of entities i and j, respectively, measured by entity i. The distance $r_{ij}^{m_i}$ between the two entities is therefore defined as the Euclidean norm of the corresponding difference vector $r_{ij}^{m_i} = ||r_i^{m_i} - r_j^{m_i}||$.*

Given a swarms of entities, one may associate a graph topology based on distances. In particular, we define the maximum distance, under which entities can be considered connected (or neighbors in the swarm graph), and the minimum distance, under which the safety of entities is compromised.

DEFINITION 2.5 (CONNECTIVITY LIMIT). *Connectivity limit R between two entities is the maximal distance where those entities are considered connected to each other (e.g., by visibility).*

DEFINITION 2.6 (PROXIMITY LIMIT). *Proximity limit r between two entities is the minimal safety distance that prevents the entities from collision.*

Of course, the connectivity limit entails a neighborhood relationship between entities, i.e.

DEFINITION 2.7 (NEIGHBORS). *The set $N(i)$ of all neighbors of entity i is the set of all entities whose distance from entity i measured by entity i does not exceed the connectivity limit value R, i.e. $r_{ij}^{m_i} \leq R$.*

In order to obey behavioral rules such as Reynolds' cohesion and separation, we propose to describe the interaction between swarm neighbors via the following intuitive mechanism: two neighboring entities i and j, such that $i \in N(j)$ and $j \in N(i)$, are connected by (virtual) spring, which applies force on its end points (i.e. the entities) based on its length. Formally

DEFINITION 2.8 (SPRING). *Spring is a virtual structure connecting any two neighboring entities. Spring size equals the distance $r_{ij}^{m_i}$ between the two entities i and j, where $r \leq r_{ij}^{m_i} \leq R$. This important property of the spring states, that it can neither stretch above R, nor shorten under r. Each pair of entities i, j that become neighbors(i.e. $i \in N(j)$ and $j \in N(i)$) obtains a spring. The force that the spring applies on its ends is proportional to its size, i.e. the force on each end is $F = (r_{ij}^{m_i} - (R-r)/2)/2$, so the spring attains its equilibrium state in the middle between R and r.*

Note that the definition above does not necessarily mimic a physical spring. However, regardless of the implementing structure, the force applied on each entity by its springs affects it velocity, which one can define in the standard fashion, i.e.

DEFINITION 2.9 (VELOCITY). *The velocity $v_i^{m_i}$ of entity i is the time derivative of its position vector $v_i^{m_i} = \dot{r}_i^{m_i}$ and it is assumed to be constant within a clock cycle.*

Finally, we note that one of the main purposes of the algorithm described in the bulk of this paper is to preserve initial connectivity of the swarm network. We therefore define

DEFINITION 2.10 (CONNECTIVITY). *Let $G(V, E)$ be the graph whose nodes are the swarm entities and the bidirectional edges are the swarm bidirectional springs (both entities on the end of the spring are aware of its existence). The swarm is said to be connected if its corresponding graph is connected.*

DEFINITION 2.11 (NON-CRITICAL STARTING POINT). *Non-critical starting point is a sufficient condition to start using direction election algorithms. It states, that $\forall i, j | j \in N(i), r + 2e \leq r_{ij}^{m_i} \leq R - 2e$, that is all the entities are on the safe distance from each other, and also the swarm is connected.*

3. DIRECTION ELECTION IN SILENT SWARMS

3.1 Synchronous Systems

Leadership of an entity in silent swarms can only be expressed by moving in a way that violates the basic Reynold's rules. However, such arbitrary motions should be done with care in order to avoid changes to the connectivity (or more strictly, the graph topology) of the swarm. We begin our investigation of how this can be done with the case of synchronous swarms, where all entities share the same global clock and all perform their distance measurements at the same time (but otherwise exchange no other information by any other means).

3.1.1 Leaderships in flocking swarms

By Definitions (2.9) and (2.1), the velocity between subsequent measurements is assumed to be constant. We observe that when an entity j decides to lead, it may move at most $(R - r_{ij}^{m_i})/2 - 2e$ in the direction of stretching each one of the springs connected to it, and $(r_{ij}^{m_i} - r)/2 - 2e$ in the direction of shortening the springs. We note that the halving of the slack in the spring is necessary because i's neighbor on the other end of a spring may decide to lead simultaneously. For the same reason, the error bound e is doubled in all the equations. Hence, in order to make the movement in the desired direction $\widehat{DIR}$, the leader must obey

$$r_i^{m_i^{next}} \longleftarrow r_i^{m_i} + min(ST, SH) \cdot \widehat{DIR} \quad (1)$$

where

$$ST = min_{j \in N(i)}((R - r_{ij}^{m_i})/2 - 2e) \cdot \widehat{DIR} \quad (2)$$

is the minimal move among all neighbors of entity i in the direction of stretching of the spring, and

$$SH = min_{j \in N(i)}((r_{ij}^{m_i} - r)/2 - 2e) \cdot \widehat{DIR} \quad (3)$$

is the minimal move among all neighbors of entity i in the direction of shortening of the spring. Recall that e is the error bound on the measured distance of neighboring entities, and $N(i)$ is the set of neighbors of entity i. Since this policy is maintained by all pairs of neighboring entities, we are guaranteed that springs do not stretch over the connectivity limit R and at the same time do not shorten beyond the proximity limit r, assuming a non-critical starting point. Hence, the swarm remains connected and safety distances are maintained at all times.

Naturally, entities that do not wish to lead, should simply follow the regular Reynolds rules. Formally, in our spring system, this can be expressed in the following manner:

$$r_i^{m_i^{next}} \longleftarrow r_i^{m_i} + dt^2 \sum_{j \in N(i)} (r_{ij}^{m_i} - (R-r)/2)/2$$

$$dt \sum_{j \in N(i)} v_j^{m_i} - \sum_{j \neq i} Correction_i^j \widehat{r_{ij}^{m_i}} \quad (4)$$

where

$$Correction_i^j = max[0, Short_i^j, Stretch_i^j] \quad (5)$$

is the correction in the direction from $\widehat{r_{ij}^{m_i}}$ to entity's i movement in order to prevent the spring between entity i and entity j to shorten below r or stretch above R. Here we defined the shortening and stretching spring violations as

$$Short_i^j = r + (r_{ij}^{m_i} - r)/2 - |r_i^{m_i^{next}} - r_j^{m_i}| + 2e \quad (6)$$

$$Stretch_i^j = |r_i^{m_i^{next}} - r_j^{m_i}| - (R - (R - r_{ij}^{m_i})/2) + 2e \quad (7)$$

This way we make sure that entities from both ends of the spring will correct their movements in order to prevent violation of the spring Definition 2.8.

While the above set of behavioral rules for leaders and non leaders guarantees that no bidirectional edges in the swarm graph are being disconnected, it does allow the formation of new edges. Formally, each swarm member should update its neighborhood set at all cycles by executing

$$\forall j \quad r_{ij}^{m_i} > R \Rightarrow N^{next}(i) \longleftarrow N(i) - \{j\}$$

$$\forall j \quad r_{ij}^{m_i} \leq R \Rightarrow N^{next}(i) \longleftarrow N(i) \cup \{j\} \quad (8)$$

It can be realized, that due to the possible measurement errors, the newly created edges may not be bidirectional. In this situation, these new edges can be broken by the unawared neighbor. Note, that this case is not worse than the case without the newly created directed neighborship as the connectivity is maintained by the links for which both endpoint entities measure distance less than R.

To conclude this elementary case, we do note that an upper bound for the convergence rate of a swarm to a single leader may permit slight deviation from the exact speed and direction of the leader. Furthermore, in many practical situations, explicit convergence of all entities to the exact velocity of the leader is not crucial, but the movement of the swarm's center of mass in the direction of a leader, up to a small predetermined deviation, is sufficient. While the limits r and R are not approached, the center of mass of the swarm is given under the external leading force only. In this simplified case, the convergence of entities to the approximate leader velocity, up to an error, will happen in polynomial time, due to the limitations of the spring stretching and shortening by R and r. For the general case, the result is more complicated, and can be obtained numerically.

Clearly, the algorithm outlined so far would operate in the presence of multiple leaders, but it does not guarantee that these leaders would not conflict each other to stall the swarm as a whole or to act in a diverging behavior forever. Hence, some scheme of direction election should be integrated, as we begin to discuss in the next subsection.The goal of this paper is to develop efficient algorithms for the whole swarm to follow some leading direction, which may sometimes contradict the leading direction of other leaders.

3.1.2 Swarms with labelled entities

Assume first that all the entities in the swarm are *labelled* and aware of their identity. In such cases, entities can be numbered and ordered by total ordering, and each entity can then be allocated unique time slots in which only it may become a leader. Let $ORDER$ be the label of entity i. For fairness, this time slot allocation can be done via round robin, using the global clock T^{global} allocate T slots to entity i, where $(T^{global} \mod n) + 1 = ORDER$. The convergence rate T of a swarm obeying regular Reynolds rules is an intrinsic constant in all algorithms presented below and is used to represent time slot duration. Note, that convergence rate properties are straightly dependent on this constant.

Following the previous subsection, we hence obtain the following algorithm:

ALGORITHM 3.1. *Direction Election Algorithm for Flocks with Labeled Entities* **For entity i who wants to lead**

parameter:
i: entity number
input:
$r_{ij}^{m_i}$*: distances to all neighbors of entity i*
$N(i)$*: neighbor list of entity i*
Persistent variables:
n: number of entities in the swarm
T: number of cycles until complete leader following
ORDER is the leading order for entity i
Algorithm:
1: for $cycle = 1, \ldots, nT$ do
2: if $ORDER == \lfloor (cycle - 1)/T) + 1 \rfloor$ then
lead according to eq. (1)
3: else *obey Reynolds rules according to eq. (4)*
4: update *neighbor list according to eq. (8)*
5: continue *from step 1.*

In order to provide each leader a possibility to move in the leading direction, we may divide the time slot of duration T into 2 parts: one for spring network convergence between any subsequent leading time slots, and the second for leadership itself. Since the equilibrium state of each spring is in the middle between r and R, after the first part of the time slot all the springs are near the equilibrium. Note, that this is also the optimal position, to allow each leader to move in any desired direction.
Naturally, entity j that does not want to lead in its designated time slots, simply obeys Reynolds rules according to Eq. 4.

Bidirectional springs are only allowed to be formed, but they are never removed under the direction election algorithm. Since the initial swarm graph is assumed connected, and the connectivity is defined on bidirectional springs only, then connectivity is preserved. Collisions are avoided by the definition of the unidirectional and bidirectional spring, which can not become shorter than the proximity limit distance. Using this logic, we can state the following theorem:

THEOREM 3.1. *The initial connectivity of a swarm is preserved and collisions between all entities are avoided under the direction election algorithm, i.e. there is a path of springs connecting any pair of entities of a swarm during any stage of the algorithm.*

The time of starting the leading period for each entity is predetermined by its label, i.e. by its sequential number. Such a leader will lead by itself starting from its first designated time slot for duration of T cycles, a time at which the leading opportunity is passed to the next entity. Since the equilibrium state of all springs is in the middle between R and r, then any leader will obtain the possibility to move during its leading time slot, when it leads alone. Hence, the following theorem can be stated:

THEOREM 3.2. *Direction election algorithm will make the swarm follow a single leader for periodic time slots of duration at least T.*

3.1.3 Flocks of Unlabelled Entities

Assume now that the swarm consists of identical unlabelled entities which cannot be enumerated. Clearly, while leading slots cannot be allocated deterministically in this case, entities can randomly choose their leading slot. However, this random selection should be done with care to ensure a similar fair chance to all entities to lead, and to prevent two entities from leading at the same time.

ALGORITHM 3.2. *Direction Election Algorithm for Flocks with Anonymous Entities* **For entity i who wants to lead**

parameter:
i: entity number
input:
$r_{ij}^{m_i}$*: distances to all neighbors of entity i*
$N(i)$*: neighbor list of entity i*
persistent variables:
n: number of entities in the swarm
T: number of cycles until complete leader following
ORDER is the leading order for entity i
1: Uniformly generate ORDER on the range [1,P]
2: for $cycle = 1, \ldots, PT$ do
3: if $ORDER == \lfloor (cycle - 1)/T) + 1 \rfloor$ then
lead according to eq. (1)
4: else *obey Reynolds rules according to eq. (4)*
5: update *neighbor list according to eq. (8)*
6: continue *from step 1.*

Each entity runs the standard direction election algorithm for flocks with labelled entities, when a sequence of numbers $ORDER$ is randomly generated from the uniform distribution on the range $(1, \ldots, P)$, where P is the predetermined size of a period, whose optimal choice is described in eq. (11) later in the paper. Thus, $ORDER$ value may be equal for different entities, with arbitrarily small predetermined probability. This way, we obtain a small number of time slots, where multiple leaders compete for leadership. In such time slots, the swarm will flock according to the average of the leader directions [20].

The following theorem and its proof is similar to Theorem 3.1

THEOREM 3.3. *The initial connectivity of a swarm is preserved and collisions between all entities are avoided under the direction election algorithm, i.e. there is a network of bidirectional springs connecting any pair of entities of a swarm during any stage of the algorithm.*

THEOREM 3.4. *Direction election algorithm will make the swarm follow a single leader for periodic time slots of duration at least T with predetermined probability, given the period duration P.*

Proof: By construction, the leading starting time of each entity is distributed uniformly in the range $1, \ldots, P$. Let us first assume that a particular slot is selected by a *single* entity. Then, such a leader will lead by itself, starting from its time slot onward and until the slot of a next potential leader begins. Hence, in such a case, a single leader is elected and stable flocking of the swarm is achieved, for time slot of duration at least T.

Of course, it is critical to understand how likely it is for any single candidate leader to find itself selecting a leading slot without conflicting with others. The probability that at least k entities lead alone during the period P is:

$$Prob(Number^{alone}_{leading} \geq k) =$$

$$= \frac{P(P-1)\ldots(P-k+1)(P-k)^{n-k}}{P^n} \qquad (9)$$

This is due to the fact, that the number of ways to distribute k different entities into P different slots, where maximum a single entity is allowed in any slot is $P(P-1)\ldots(P-k+1)$. The total number of ways to distribute n different entities into P different slots is P^n. Increasing P for given n also increases the probability, due to the fact, that $lim_{P\to\infty} Prob(Number^{alone}_{leading} \geq k)) = 1$. Obviously, if $Prob(Number^{alone}_{leading} \geq n) = \frac{1}{q}$, then $\log_{(1/(1-q))}(n)$ sequential periods are needed to obtain the probability of any particular entity out of n to lead alone approximately approaching unity. ■

Figure 1 gives an example of the increasing probability of one or more entities to lead alone as a function of the period P duration, for number of entities $n = 10$.

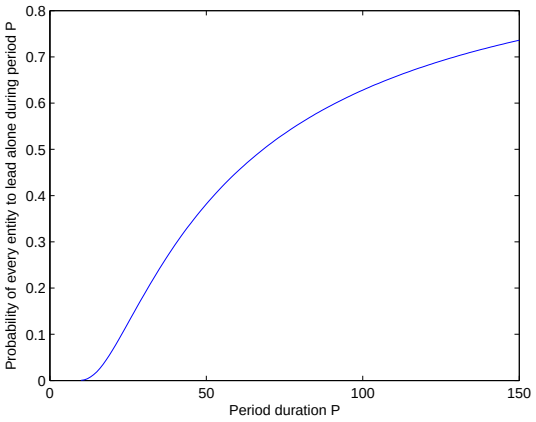

Figure 1: Probability of all n entities leading alone vs. period duration P, n=10

In order to choose the period length P consider the following argument:

$$\log Prob(Number^{alone}_{leading} \geq k) = \log P(P-1)\ldots(P-k+1)+$$

$$(n-k)\log(P-k) - n\log P \geq$$

$$n\log(P-k) - n\log P = -n\log\frac{P}{P-k} \geq -\log q \qquad (10)$$

Furthermore, one can choose P according to

$$P \geq \frac{kq^{\frac{1}{n}}}{q^{\frac{1}{n}} - 1} \qquad (11)$$

We should choose $k = n$, since this will reduce the total number of periods needed, since for larger k the lower bound for P in the eq. (10) is less strict.
In order to choose the last unknown variable q in eq. (11) we need to solve the following optimization problem:

$$minimize_q[\log_{(1/(1-q))}(n)\frac{nq^{\frac{1}{n}}}{q^{\frac{1}{n}} - 1}] \qquad (12)$$

Taking the derivative of the minimization function with respect to q we realize, that it vanishes at $q^* = RootOf[nq(q^{1/n} - 1) + (1-q)ln(-1/(-1+q))]$ (see figure 2). Thus, q^* is a single minimum of the minimization function.

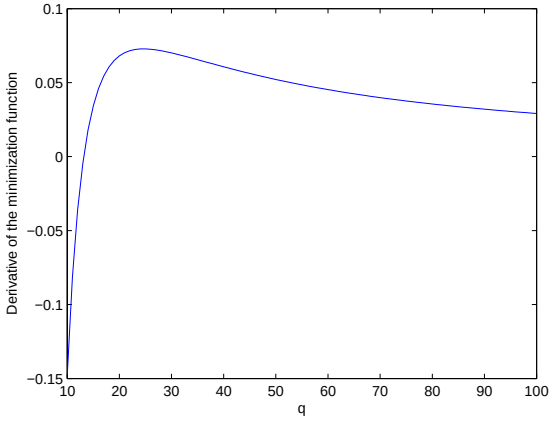

Figure 2: Derivative of the minimization function with respect to q, n=10

Figure 2 gives an example of the derivative of the minimization function with respect to q, for number of entities $n = 10$.

3.2 Semi-synchronous networks

Let us start the development of algorithms for semi-synchronous networks by allowing phase shifts between time slots of different entities. While still assuming that all the entities start counting period P simultaneously. We argue, that the direction election algorithm for flocks with labelled entities described in subsection 3.1 as well as direction election algorithm for flocks with anonymous entities described in subsection 3.2 work when no synchronization between entities exists, while an additional synchronization part in time slot T should be devoted, in order to preserve the probabilistic properties of these algorithms.

ALGORITHM 3.3. *Direction Election Algorithm for Flocks with Anonymous Entities and Unsynchronized Clocks* **For entity i who wants to lead**

parameter:
i: entity number
input:
$r_{ij}^{m_i}$: distances to all neighbors of entity i
$N(i)$: neighbor list of entity i
persistent variables:
n: number of entities in the swarm
T: number of cycles until complete leader following
$ORDER$ is the leading order for entity i
1: Uniformly generate $ORDER$ on the range [1,P]
2: for $cycle = 1, \ldots, PT$ do
3: if $ORDER == \lfloor (cycle - 1)/T) + 1 \rfloor$ then
lead according to eq. (1)
4: else *obey Reynolds rules according to eq. (4)*
5: update *neighbor list according to eq. (8)*
6: continue *from step 1.*

Indeed, allowing entities to lead only in this additional synchronization part of the time slot T, we obtain the leadership for duration of at least T, while no two entities with different $ORDER$ values can compete for leadership. In addition, eq. (5) implies, that no spring violations can occur, when cycle size dt is constant among different entities, and only the measurement time starts with a different phase. But note that this situation is covered by the synchronous case, since here the maximal possible movement size is less than the similar quantity in the synchronous case.

THEOREM 3.5. *Direction election algorithm 3.1 preserves connectivity and avoids collision between entities for partially asynchronous systems.*

Proof: We show that by running the synchronous direction election algorithm in partially asynchronous system no violation of the spring Definition 2.8 happens. Here, each entity performs its measurements with the same time periods dt, but with different starting time phase. In particular, let us assume, without loss of generality, that entity i performs its measurements at time sequence $\{0, dt, 2dt, ..., kdt, ...\}$, while entity j performs its measurements at time sequence $\{ph, dt+ph, 2dt+ph, ..., kdt+ph, ...\}$, where $0 < ph < dt$ stands for the phase shift of entity j. There are only 2 suspectable pitfalls, which should be checked: making movement as a leader, and obeying Reynolds rules. In both cases, the maximal movement for entity i starting at time 0 is $(R - r_{ij}^{m_i})/2$ in the direction of stretching the spring between i and j, and $(r_{ij}^{m_i} - r)/2$ in the direction of shortening the spring between i and j, where $r_{ij}^{m_i}$ is the length of the spring at time 0 measured by entity i. By Definitions (2.9) and (2.1) the velocity between subsequent measurements is assumed to be constant. Thus, entity i passes until the time ph a distance of exactly

$$S_i^{ph} = \frac{ph}{dt}(R - r_{ij}^{m_i})/2 \tag{13}$$

in the direction of stretching the spring between i and j, and

$$s_i^{ph} = \frac{ph}{dt}(r_{ij}^{m_i} - r)/2 \tag{14}$$

when we take the direction in which spring shortens. Then, entity j makes its movement, according to the measurement at time ph. Entity j made until the time dt a distance of exactly $S_j^{dt} = \frac{dt-ph}{dt}([R - \frac{ph}{dt}(R - r_{ij}^{m_i})/2] - r_{ij}^{m_i})/2$ in the direction of stretching the spring between i and j, and $s_j^{dt} = \frac{dt-ph}{dt}([r_{ij}^{m_i} - \frac{ph}{dt}(r_{ij}^{m_i} - r)/2] - r)/2$ in the direction of shortening the spring. Thus, at time dt, after a complete cycle of entity i, the difference between R and the maximal length of the spring between i and j is $NV^{dt} = R - L^{dt} = R - \{r_{ij}^{m_i} + (R - r_{ij}^{m_i})/2 + \frac{dt-ph}{dt}([R - \frac{ph}{dt}(R - r_{ij}^{m_i})/2] - r_{ij}^{m_i})/2\} \geq (R - r_{ij}^{m_i})/2 - ([R - \frac{ph}{dt}(R - r_{ij}^{m_i})/2] - r_{ij}^{m_i})/2 = \frac{ph}{dt}(R - r_{ij}^{m_i})/4 \geq 0$. It means that the length of the spring between i and j is less or equal than R. Also, at time dt, after a complete cycle of entity i, the difference between the minimal length of the spring between i and j and r is $nv^{dt} = l^{dt} - r = r_{ij}^{m_i} - (r_{ij}^{m_i} - r)/2 - \frac{dt-ph}{dt}([r_{ij}^{m_i} - \frac{ph}{dt}(r_{ij}^{m_i} - r)/2] - r)/2 - r \geq (r_{ij}^{m_i} - r)/2 - ([r_{ij}^{m_i} - \frac{ph}{dt}(r_{ij}^{m_i} - r)/2] - r)/2 = \frac{ph}{dt}(r_{ij}^{m_i} - r)/4 \geq 0$ Thus, the spring's length between i and j is larger than or equal to r.
Proceeding by mathematical induction, the spring size between i and j is less than or equal to R and greater or equal to r after any number of cycles. The same is true for all other springs in the network, since the spring has been chosen arbitrarily. ■

THEOREM 3.6. *Direction election algorithm will make the swarm follow a single leader at least k times in a period P for*

periodic time slots of duration at least T with predetermined probability, given P.

Proof: A time of starting the leading time slot for each leader is uniformly generated upon the range $1, \ldots, P$. Such a leader will lead alone starting from its time slot on, until the next leader in the sequence wants to lead. So, single leader is elected and stable flocking of the swarm is achieved, for time slot of duration at least T. The probability that at least k entities lead alone is equal to the probability in eq. (11), since exactly the same probabilistic rule is applied in this case. Since the equilibrium state of all springs is in the middle between R and r, then after the spare part of the time slot T, dedicated for spring network convergence, all the springs will stay near their equilibrium in the middle between r and R. Then any leader will obtain the possibility to move during its leading time slot, when it leads alone. ■

3.2.1 Real semi-synchronous networks

For the semi-synchronous networks, entities can start counting period P at different times. The next theorem shows, that this case is also fully covered by the semi-synchronous algorithms presented above.

Since the statement of liveness for networks with unsynchronized clocks Theorem 3.5 and its proof do not depend on the period starting time, then

THEOREM 3.7. *Liveness for networks with unsynchronized clocks Theorem 3.5 is applicable for Semi-synchronous Networks.*

THEOREM 3.8. *Progress for synchronous network with time shift Theorem 3.6 is applicable for semi-synchronous networks.*

Proof: Let us assume without limiting the generality that entity i starts counting its period P from a time $t = -\varepsilon$. Its $ORDER$ value is uniformly distributed on the range $[1-\varepsilon, P-\varepsilon]$, that is $Prob(ORDER = o, o \in [1-\varepsilon, P-\varepsilon]) = \frac{1}{P}$. Immediately after the period of duration P another period of duration P starts, where $ORDER$ value of entity i is also uniformly distributed, that is $Prob(ORDER = m, m \in [P-\varepsilon, 2P-\varepsilon]) = \frac{1}{P}$. Then $ORDER$ value of entity i must be uniformly distributed on the range $[1, P]$. Indeed, $Prob(ORDER = q, q \in [1, P]) = \frac{1}{P}$. The same is true for every entity in the system. Since the only assumption on period P was the uniform distribution of $ORDER$ value of each entity on the range $[1, P]$ in the statement and the proof of the Progress for Synchronous Network with Time Shift Theorem 3.4, then it is also applicable for Semi-synchronous Networks. ■

4. CONCLUDING REMARKS

In this work we enhance Reynolds Boids to enable symmetry breaking, handle measurement errors and support direction election. We prove correctness and believe that the schemes presented can be used in practice. The algorithms presented can be tuned for the cases in which various threats or flocking goals with different urgencies coexist in the system. To address this scenario we introduce priority mechanism. The priority variable has a predetermined scale of values, dividing the priority to different scenarios present in the system. Then, direction election is influenced by the priority value of each entity. For this scheduler, we have that for a random generation of $ORDER$ a period of P is multiplied by the priority category number N_P, starting from $N_P = 1$ for the highest priority.

The priority of each entity is calculated asynchronously at the time of an appropriate scenario arrival to the entity. Each entity performs such priority calculations all the time according to a predetermined formula. At this moment, new period starts for this entity, and the period duration for it is influenced appropriately.

Thus, direction election algorithm will make the swarm to follow a single leader for periodic time slots of duration at least T with arbitrarily high predetermined probability, since the time of starting the leading time slot for each leader is uniformly generated upon the range $1, \ldots, N_P P$. Such a leader will lead alone starting from his time slot on, until the next leader in the sequence wants to lead. This may happen when another leader was previously scheduled or the priority scheduling took place. So, single leader is elected and stable flocking of the swarm is achieved, for time slot of duration at least T.

The probability that at least k entities lead alone is greater or equal to the probability in eq. (11), since exactly the same probabilistic rule is applied in the case, where all entities have equal priority, otherwise, the probability is higher, since the number of entities in each period P is less than n.

5. REFERENCES

[1] Chazelle, B., "Natural Algorithms", *Proc. 20th SODA*, pp. 422-431, 2009.

[2] Chazelle, B., "The Convergence of Bird Flocking", *arXiv:0905.4241v1*, May 2009.

[3] Chazelle, B., "Analytical Tools for Natural Algorithms", *Proc. 1st ICS*, pp. 32-41, 2010.

[4] Cucker, F, Smale, S., "Emergent behavior in flocks", *IEEE Trans. Automatic Control*, Vol. 52, pp. 852-862, 2007.

[5] DeGennaro, M. C, Jadbabaie, A., "Decentralized Control of Connectivity for Multi-Entity Systems", *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 3628-3633, San Diego, CA, Dec. 2006.

[6] Fischer, M, Jiang, H., "Self-stabilizing direction election in networks of finite-state anonymous agents", *OPODIS*, pp. 395-409, 2006.

[7] Hendrickx, J.M, Blondel, V.D., "Convergence of different linear and non-linear Vicsek models", *Proc. 17th International Symposium on Mathematical Theory of Networks and Systems (MTNS2006)*, Kyoto (Japan), pp. 1229-1240, July 2006.

[8] Jadbabaie, A., Lin, J., and Morse, A. S., "Coordination of Groups of Mobile Autonomous Agents using Nearest Neighbor Rules", *IEEE Transactions on Automatic Control*, Vol. 48(6), pp. 988-1001, 2003.

[9] Ji, M., Egerstedt, M., "Distributed Formation Control while Preserving Connectedness", *IEEE Transactions On Robotics*, Vol. 23, No. 4, Aug 2007.

[10] Jiang-Ping, H., Hai-Wen, Y., "Collective coordination of multi-agent systems guided by multiple leaders", *IEEE Transactions On Robotics*, Vol. 18, No. 9, 2009.

[11] Li, S, Wang, H., "Multi-entity coordination using nearest neighbor rules: revisiting the Vicsek model", *arXiv:cs/0407021v2*, 2004.

[12] Malpani, N., Welch, J., and Vaidya, N., "Direction Election Algorithms for Mobile Ad-hoc Networks", *DIAL-M: Workshop in Discrete Algorithms and Methods for Mobile Computing and Communications*, 2000.

[13] Moreau, L., "Stability of multiagent systems with time-dependent communication links", *IEEE Transactions on Automatic Control 50*, pp. 169-182, 2005.

[14] Olfati-Saber, R, Murray, R. M., "Consensus Problems in Networks of Agents with Switching Topology and Time-Delays", *IEEE Transactions on Automatic Control*, Vol. 49(9), pp. 1520-1533, September 2004.

[15] Olfati-Saber, R., "Flocking for multi-entity dynamic systems: algorithms and theory", *IEEE Transactions on Automatic Control 51*, pp. 401-420, 2006.

[16] Olshevsky, A, Tsitsiklis, J.N., "Convergence speed in distributed consensus and averaging", *SIAM Journal on Control and Optimization*, Vol. 48, pp. 33-55, 2009.

[17] Reynolds, C., "Flocks, Birds and Schools: A Distributed Behavioral Model", *Computer Graphics*, Vol. 21, pp. 25-34, 1987.

[18] Shi, H., Wang, L., and Chu, T., "Coordination of multiple dynamic agents with asymmetric interactions", *Proc. 2005 IEEE International Symposium on Intelligent Control*, Cyprus, pp. 1423-1428, 2005.

[19] Sorensen, N., Ren, W., "A unified formation control scheme with a single or multiple leaders", *Proceedings of the American Control Conference*, New York, 2007, pp. 5412-5418

[20] Su, H., Wang, X., and Yang, W., "Flocking In Multi-Entity Systems with Multiple Virtual Leaders", *Asian Journal of Control*, Vol. 10, No. 2, pp. 238 245, March 2008.

[21] Tahbaz-Salehi, A, Jadbabaie, A., "On recurrence of graph connectivity in Vicsek's model of motion coordination for mobile autonomous agents", *Proc. 2007 American Control Conference*, New York, NY, pp. 699-704, 2007.

[22] Tang, G, Guo, L., "Convergence of a class of multi-entity systems in probabilistic framework", *Journal of Systems Science and Complexity*, Vol. 20, pp. 173-197, 2007.

[23] Tanner, H. G., Jadbabaie, A., and Pappas, G. J., "Stable flocking of mobile agents, Part I: fixed topology", *Proc. IEEE Conference on Decision and Control*, Maui, Hawaii, pp. 2010-2015, 2003.

[24] Tanner, H. G., Jadbabaie, A., and Pappas, G. J., "Stable flocking of mobile agents, Part II: dynamic topology", *Proc. IEEE Conference on Decision and Control*, Maui, Hawaii, pp. 2016-2021, 2003.

[25] Tanner, H. G., Jadbabaie, A., and Pappas, G. J., "Flocking in Fixed and Switching Networks", *IEEE Transactions On Automatic Control*, Vol. 52, No. 5, May 2007.

[26] Vicsek, T., Czirok, A., Jacob, E. B., Cohen, I., and Schochet, O., "Novel Type of Phase Transitions in a System of Self-Driven Particles", *Physical Review Letters*, Vol. 75, pp. 1226-1229, 1995.

[27] Zavlanos, M. M, Pappas, G. J., "Potential Fields for Maintaining Connectivity of Mobile Networks", *IEEE Transactions on Robotics*, Vol. 23(4), pp. 812-816, Aug. 2007.