

On Practical, Safe, and Convergent Protocols for Agent Formations

Vladimir Estivill-Castro¹, Esteve Fernandez², and René Hexel¹

¹Griffith University, Nathan, Queensland 4111, Australia
 {v.estivill-castro,r.hexel}@griffith.edu.au

²Universitat Pompeu Fabra, Barcelona, 08018, Catalunya, Spain.
 esteve.fernandez@gmail.com

Abstract

We investigate the current state of the art in terms of formally verifiable protocols to form a formation of robots under unreliable communication. We show that in practical terms, it is possible to obtain more efficient protocols with more appealing properties.

Keywords: Agents, Robotics, Formations, Formal-verification, Unreliable communication, time-trigger architecture.

1 Introduction

The research areas of agent technologies and field robotics are overlapping as we see the emerging applications [Olson *et al.*, 2013] of teams of robots collectively performing tasks. The research in agent technology, so far, has focussed largely on negotiation protocols, task distribution, and the like, while the field of robotics has found robust answers to localisation, sensor interpretation and fusion. In this paper we are interested in protocols for a team of robots so that they keep a formation; namely, they achieve and maintain a regular arrangement of their positions. This is perhaps a simpler task than actually distributing tasks, collectively planning, or some other high level collaboration challenge, but very clearly highlights the difficulties and properties underpinning real-world communication. Consequently, we hope to critically scrutinise the protocol for its demonstrable correctness properties under the circumstances that are common to field robotics, which include finite communication speed and latency (and even lost communication) between the robots, or failure of a robot.

To commence our discussion, we will consider a simple formation problem in one dimension, which will help to clearly define the context. Consider a set $A := \{A_0, \dots, A_{n-1}\}$ of n agents/robots. Each robot A_i has localisation sensors that enable it, at time t to hold a belief $position_t(A_i) \in \mathbb{R}$ regarding its position on a line. The robots also have actuators (motors) that enable them to move along this line. That is, they can

effectively carry out a motion such as `move.to(x)`, which moves, in finite time, the robot from its current position to position x on the line. In terms of communication, robots can broadcast messages to all other robots, but there is no guarantee that such messages arrive (some robots may get the message, while others may not). Each robot also knows its number (its unique identifier). The challenge then is to design a protocol that would spread the robots evenly, making $d(A_i, A_{i+1}) = d(A_{i-1}, A_i)$, for $i = 1, \dots, n - 2$.

This challenge has been studied in the literature [Chandy *et al.*, 2011; 2008] and in particular, from the perspective of correctness, the following 2-phase protocol has been established as the point of reference [Chandy *et al.*, 2011; 2008]. We refer to it as MEAN_OF_2_NEIGHBOURS (MO2N). Each round of the protocol executes the following two phases.

COMMUNICATION_PHASE :

Each robot A_i broadcasts a message with its current position and its id.

MOTION_PHASE :

For $i = 1, \dots, n - 2$, if A_i receives the messages from A_{i-1} and A_{i+1} , then it moves to $(position_t(A_{i-1}) + position_t(A_{i+1}))/2$. Otherwise, it does not move.

This protocol can formally be verified for correctness under several assumptions. We will examine these assumptions and the properties of this protocol. Several of the properties suggest this algorithm is optimal in some senses. However, we can propose alternatives that significantly improve upon these properties for teams of collaborating robots. Thus, in Section 2 we scrutinise the properties of the protocol and contrast the theoretical model with practical requirements for implementation. Section 3 suggests an alternative protocol, and we establish several improvements (termination and better progress per round). However, in Section 4 we show that the duration of the COMMUNICATION_PHASE relative to the MOTION_PHASE is an important trade-off. Thus, in Section 5 we show another important property for our protocol (competitiveness). We confirm further the mer-

its of the alternative protocol in Section 6, and finish the paper with concluding remarks in Section 7.

2 Analysis of Properties

For convenience, each iteration of the two phases of MO2N shall be called a round. The MO2N algorithm has several interesting properties, and has been used as an illustration of the machinery for formal verification [Chandy *et al.*, 2011; 2008]. It is important to understand the assumption (or theoretical model) under which the formal verification has been carried out, as when we apply the protocol in practice, we need to observe what correspondence there is between the practical setting and the theoretical model. The assumptions allow for distributed failures on channels which may lose messages (that is, robots may fail for some time, or any channel may fail for some time, but failures are never permanent). These proofs have been made under the assumption that the robots move instantaneously, and that the broadcasting is instantaneous as well. Thus, in the analysis of the protocol (and this is a theoretical formulation) there is no need for agreement on when the MOTION_PHASE ends or when the COMMUNICATION_PHASE ends.

We now proceed to list the properties we find beneficial, although not all of them have been analysed before in the literature. We also explore how to maintain these interesting properties in practice.

Since instantaneous communication or motion is clearly only available in a hypothetical scenario, the first property of interest is what we call the **ease of implementation**. For a real-world implementation, two key aspects that matter are a *finite message delivery time* (due to latency and bandwidth constraints, message delivery is not instantaneous) and, as completion time is of concern, *bounded message delivery* (i.e. an upper bound deadline after which agreement has been established whether a message has been successfully delivered or not). Protocols implemented under a time-triggered architecture provide such deterministic message delivery under a given fault and load hypothesis¹ [Kopetz and Bauer, 2003].

Thus, while in theory, there is no need to specify in practice the duration of a round and its two phases, in practice, there is need to specify this, and using a time-trigger architecture, the deterministic arguments of the theoretical model are preserved. That is, under a time-triggered architecture, the duration d_{CP} of the

COMMUNICATION_PHASE can be defined ahead of time, as well as the duration d_{MP} of the MOTION_PHASE. Therefore, the COMMUNICATION_PHASE can be divided into as many time slots as there are agents, where each agent A_i broadcasts in the i -th time slot and listens in all other slots $j \neq i$. The MOTION_PHASE enables robots who are moving to perform their individual move for the duration of the time-slice of the phase. They either halt because they reach their destination during this round, or because the time-slice for them to move is over. This also accommodates robots having different moving speeds.

The second property is **uniformity**. All robots perform the same algorithm. They only need to know their robot number and their location.

The third property is **statelessness**. A robot does not have to remember anything about the actions performed in previous rounds of the algorithm. Each iteration does not require any memory of the past, only the present location and the robot id (which is constant) are required. In fact, the protocol does not need to remember how many robots are involved, except for the time slicing in the broadcasting. This means that a robot can crash, reboot, and re-integrate at any time in accordance with the communication protocol [Kopetz and Bauer, 2003].

The fourth property is **locality**. Robots do not need global information regarding the state of other robots.

The fifth property is **movement safety**. This relates to the use of on-board localisation sensors only, and no other means to identify one another. In MO2N, because each robot is moving at most half the distance to its neighbouring robot, it can be certain the path to its destination is clear of obstacles.

A sixth property that we find relevant here is what we call **default behaviour**. If an any point in time, robot A_i malfunctions permanently, then robot A_i+1 would no longer move, but the robots in the range A_{i+1}, \dots, A_{n-1} will spread themselves evenly among the current positions of A_{i+1}, \dots, A_n (and similarly for the robots in A_0, \dots, A_{i-1}).

The three formal properties discussed recently [Chandy *et al.*, 2011; 2008] for this protocol are under an assumption of fairness. Namely, in case of a temporary fault (e.g. messages getting lost, or a robot restarts), the corresponding failures are also temporary (and will disappear within a finite time after the triggering fault has disappeared).

The seventh property is **convergence**. I.e., all robots get arbitrarily close to the desired formation over time (this property is formulated by establishing a metric between the current vector of positions of the robots and the target vector, and showing that, for any $\epsilon > 0$ there would be a round number in the future where the position vector of the robots would be no more than ϵ away

¹It is important to note here that a similar argument can be made probabilistically, based on the corresponding, stochastic, quality of service (QoS) attributes of traditional, event-triggered communication protocols (however, the corresponding probabilities and confidence intervals are significantly more complex to establish and are not covered here).

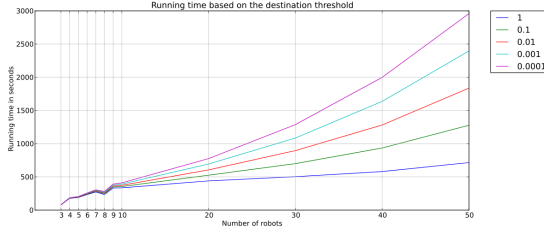


Figure 1: Running time under a Gamma distribution when the algorithm MO2N stops once all robots are within a threshold of their destination.

from the target vector).

The eighth property is **stability**. This means that the formation of the robots remains arbitrary close to the target (in the language of the metric, once the vector of positions is within ϵ of the target, it remains within ϵ).

The ninth property is the **separation** of the communication phase and the action phase. That is, the protocol does not concurrently have the robots moving while they are listening/broadcasting. This allows for deterministic modelling of the behaviour of the collective of robots, where under the same conditions, the robots will reproducibly converge in the same way, without the possibility of introducing (inherently nondeterministic) race conditions. This separation property is also nice in the sense that each robot/agent can schedule *a priori* when network activity will occur.

Another (tenth) property is **termination**, and unfortunately while the MO2N protocol converges towards its goal, it cannot be shown to terminate. Fig. 1 shows the running time when several robots are placed along a straight line initially under a Gamma distribution (so they are clustered in one extreme) and the precision threshold with respect to their final destination is the parameter distinguishing between the different curves. The smaller the precision threshold value, the more time it takes for the protocol to place the robots within the precision value of their target destination.

3 Alternative protocols

One disappointing aspect of the above protocol is the issue of termination. Let $position_t(A_i)$ be the position of agent A_i at time t . Consider a setting where $position_0(A_0) = 0$, $position_0(A_{n-1}) = n - 1$, and $position_0(A_i) = n - 2 + i/n$, for $i = 1, \dots, n - 2$.

The first thing to notice is that with failures or without any failures (no robots collapsing, no messages lost), the target position of each robot with id in $\{1, \dots, n - 2\}$ is always strictly less than its current position. More formally,

$$\forall t \geq 0, \forall i \in \{1, \dots, n - 2\}, i < position_t(A_i). \quad (1)$$

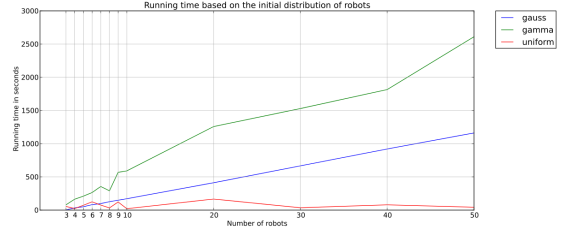


Figure 2: The MTTIS algorithm when the initial configuration of robots is from different distributions.

The proof requires an elaborate induction; that is not hard, but we trust the intuition is clear: in each round, robot A_1 only travels very close to half the distance to its destination (and similarly to Zeno's paradox, there is no round where it actually gets there).

This motivates us to suggest the following alternative protocol which we name MOVE_TO_TARGET_IF_SAFE (MTTIS). For the moment, assume the robots know the total number of robots involved.

COMMUNICATION_PHASE :

Robots broadcast their current position, their id, and how many robots they believe are involved.

MOTION_PHASE :

Each robot computes its target position $target_i = position_0(A_0) + i(position_0(A_{n-1}) - position_0(A_0))/n$.

1. If A_i receives a message from its neighbour in the direction of $target_i$, and the position of such a neighbour is at least a safe distance δ beyond $target_i$, then move to $target_i$.
2. If A_i receive a message from its neighbour in the direction of $target_i$, and it is on the way to $target_i$, then move to a position safely close (at a safety distance δ , so bodies would not collide) to the position of the neighbour in the direction of $target_i$.
3. If the message from the neighbour in the direction of $target_i$ is lost or the distance to the neighbour is already as close as the safety distance δ , then do not move.

We claim that this alternative protocol has all the properties as MO2N but additionally has the property of **termination**. And also, it has a faster convergence rate than MO2N. Figure 2 displays the running time of the algorithm MTTIS when the robots are initially placed randomly under the uniform distribution, a Gaussian and the Gamma distribution. Clustered robots in an initial position requires more time. While for a uniform distribution, the running time remains constant as the number of robots increases, for a Gaussian distribu-

tion, the running time grows linearly, and the Gamma distribution shows a larger running time to convergence.

Moreover, while MTTIS preserves the separation of the communication phase and the move phase, it does this without the restrictions imposed by MO2N. While MO2N requires the whole collective of robots to stop after each movement phase in order to establish all neighbour distances for all robots, and then calculate a new movement (which, for a robot, can potentially go in the opposite direction of its previous move), such a restriction does not exist for MTTIS. To understand why, let us examine the two possible decisions each robot A_i , individually makes after each communication phase. The robot A_i may either determine that it is safe to move all the way to its final destination. This determination is final and will not be affected by any future communication. Alternatively, A_i may find that it is not safe to move all the way towards its destination. In this case, there are two scenarios: the move phase will end before the next communication round or moving the robot will take longer than the time between the start of two communication rounds. The case where the move phase ends before the next round of communication (or the robot has determined it is not safe to move at all), the situation is the same as with MO2N: the robot stops and waits for the next round of communication. The other case, where moving the robot over a safe distance will take longer than one round of communication, is more interesting. In that case, any new message received from its neighbouring robots can only increase the safe distance the robot is allowed to travel, but never decrease that distance. Consequently, with MTTIS, the robot does not have to stop in order to communicate and update its target position. Instead the robot can continue moving and, while doing so, update its current target position (increase its overall travel distance for the round) if it receives new information from a neighbouring robot that it is safe to do so. Therefore, robots that need to travel long distances will not delay any other robots (or future communication rounds) as it is possible for these robots to move and communicate at the same time without increasing the run time or negatively affecting the convergence or stability of the algorithm.

Clearly, the proposed MTTIS requires less running time than MO2N because it actually terminates, but another argument is the following.

Proposition 1 *In any configuration, for the same duration of the MOTION_PHASE, the algorithm MTTIS will move all agents towards its target by at least as much as MO2N.*

Proof: Assume that for a given formation, the safety distance δ is less than half the distance between any pair of robots. This assumption holds since

MO2N could otherwise move robots to a position closer than δ , causing collisions. For any distance $d_{i,i\pm 1} := d(\text{position}(A_i), \text{position}(A_{i\pm 1}))$ MO2N will move A_i a maximum of $\frac{d_{i,i\pm 1}}{2}$ towards its target position target_i . With MTTIS, the maximum distance is $\min\{d_{i,i\pm 1} - \delta, d(\text{position}(A_i), \text{target}_i)\}$, which either moves the robot all the way to the target or as close as possible towards its neighbour in the direction of the target. Since, under the initial safety assumption, we have established that $d_{i,i\pm 1} - \delta \geq \frac{d_{i,i\pm 1}}{2}$, each robot will move towards its target by at least as much as MO2N. \square

Perhaps more important than the formal proof is the observation that, in non-trivial scenarios, the safety distance δ (e.g. the width of a robot) is going to be much smaller than $\frac{d_{i,i\pm 1}}{2}$, in which case the above inequality becomes $d_{i,i\pm 1} - \delta \gg \frac{d_{i,i\pm 1}}{2}$. In other words, MTTIS will reach its destination much faster than MO2N converges towards a small ϵ near that destination.

There is one more property with which to contrast the earlier protocols. We refer to this property as **monotonicity**. We say that a protocol is monotonic, if for each agent A_i , the distance to its target destination does not increase in any round. Namely $d(\text{position}_t(A_i), \text{target}_i) \geq d(\text{position}_{t+1}(A_i), \text{target}_i)$. It is not hard to see that there are examples of MO2N where an agent actually travels in the wrong direction, away from its final destination, before it eventually starts moving in the right direction. However, it is also clear that our alternative is indeed monotonic for any subset of A , as each agent always takes steps in the direction of its target.

4 The relevance of the duration of the move phase

The implementation of the two algorithms reveals a very interesting trade-off between the amount of communication between the group of agents and the amount of activity performed during the motion (action) phase. As we have introduced, the original protocol for which properties are formally proved (MO2N) does not specify the duration of the action phase, and assumes communication happens instantaneously. However, any practical implementation must establish a duration for the action phase and a minimum duration for the communication phase. If the duration of the action phase is short, then an interesting phenomenon occurs: the robots have barely moved as there is, in fact, little new information and the communication phase essentially does not add value to the effort of each individual agent. The protocol is inefficient when the duration of the move phase is too small. Most of the time is spent on communicating about an environment that has not changed much. So, consider for example a duration for the move phase that

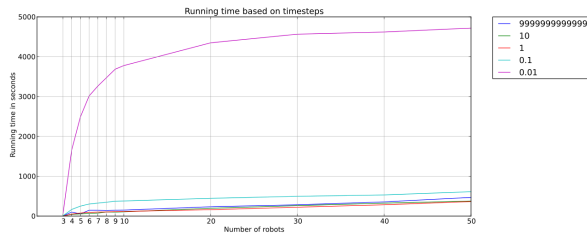


Figure 3: The running time of MTTIS algorithm when the initial configuration of robots is from a Gaussian distribution and we vary the duration of the MOTION_PHASE.

is smaller than the duration of the communication phase and that barely enables the robots to move. Then, each round is mostly communication and no effective work gets done. We actually believe this is a more generic observation that just for the protocols above, but to the best of our knowledge we have not found studies in the literature about this trade-off.

The other extreme is when the duration of the action phase is much much longer and robots and agents may long have stopped moving or carry out movements that lead them far astray, but they only are able to correct after the next communication phase. The protocol becomes inefficient as no work is performed for a period of time (the difference between the time allotted to movement and the actual movement time) or significant work (movement) is performed that actually has to be reversed when the next round provides the necessary information to continue with the structuring of the formation. In a more general context, any two-phase protocol (communication and action) among collaborating agents faces this trade-off. Too frequent communication (small duration for the action phase) is wasteful as robots waste communication resources and time that should be employed in progressing with the task at hand, and there is little or no information as the world has not changed because no work has been performed. On the other hand, a long duration of the action phase runs the risk that work and actions of other agents change the environment and work needs coordination, or the work done individually may be wasteful, or may even need to consume resources to be undone.

Figure 3 shows how dramatically it could be to have a very small duration for the MOTION_PHASE. The running time is several orders of magnitude worse. At the other extreme, a very large duration for the MOTION_PHASE is also not optimal. The trade-off is more visible if the robots are initially placed under a severely non-uniform distribution (Gamma distribution). Figure 4 shows the number of messages sent while Figure 5 shows the running time as we vary the

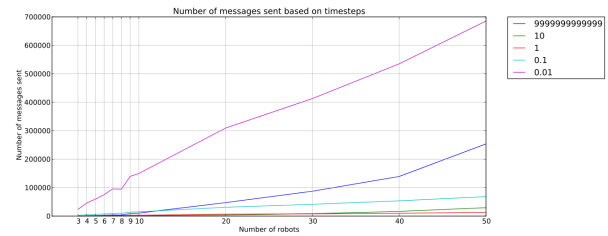


Figure 4: The number of messages by the MTTIS algorithm when the initial configuration of robots is from a Gamma distribution and we vary the duration of the MOTION_PHASE.

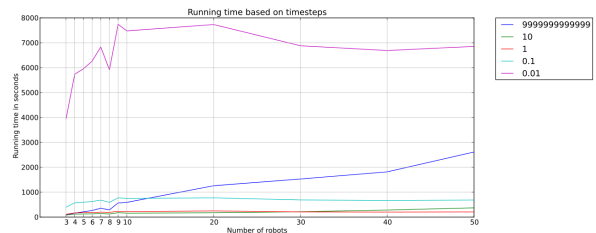


Figure 5: The running time by the MTTIS algorithm when the initial configuration of robots is from a Gamma distribution and we vary the duration of the MOTION_PHASE.

duration of the MOTION_PHASE. Again, a very short duration produces very slow convergence and lots of messages, but also a too-large duration phase is not optimal.

For algorithms, such as MO2N, that follow a strict two-phase protocol, this trade-off is a hard one. If the distribution of agents is unknown, the efficiency of the algorithms is largely determined by the quality of the underlying assumption, i.e. how well the distribution model (and corresponding calculation of expected phase duration) matches reality. The MTTIS algorithm does not face such a strong restriction, i.e. the robots will not have to stop moving in order to update their status. Therefore, the worst-case scenario here is that one or more robots would already have stopped ahead of the next communication phase. Consequently, more frequent communication tends to improve (and will not cause to deteriorate) the efficiency of the action phase. This makes the modelling of the system much simpler, as the complex interaction between the durations of the communication and action phases gets de-coupled. As a result, the timing of the communication phase can be optimised towards the properties of the communication channel. More frequent communication has no negative impact on the action phase.

5 Competitiveness

Competitive analysis [Borodin and El-Yaniv, 1998] is a method for analysing online algorithms. The aim is to qualify the performance of an online algorithm (an algorithm that can not use the future sequence of events, but must act properly on any such future sequence) by comparing it to the performance of an optimal offline algorithm that can view the sequence of events in advance. An algorithm is competitive if its competitive ratio (the ratio between its performance and the offline algorithm's performance) is bounded for all possible future sequences. In this section, we aim at presenting and argument for the competitiveness of MTTIS. We will show competitiveness among the algorithms that use a time-triggered architecture where there is a communication phase and a motion phase. In fact, we have already argued why the motion phase must not overlap with the communication phase in MO2N, while MTTIS only has a one-way dependency (i.e. the motion phase cannot commence until relevant position information has been received, but updated information can, from the perspective of the motion phase, be communicated at any point in time). Thus, any algorithm has rounds that include a communication phase and then an action phase. We let $d_{CP}^{(t)}$ be the duration of the communication phase in round t while $d_{MP}^{(t)}$ denotes the duration of the motion phase in round t . Since the focus are time-triggered protocols, $d_{CP}^{(t)}$ is a constant for all $t \in \{1, \dots, T\}$ (where T is the total number of rounds).

5.1 Off-line optimality with respect to the duration of the motion phase

We will make some other clarifying assumptions along the way. The first assumption is that all robots move at a constant common velocity v . We also need some notation. Clearly, the placement of the i^{th} robot defines the distance it must travel from its initial position $position_0(A_i)$ to its final destination, $target_i$. If we denote by d_M the largest of these distances, that is,

$$d_M = \max_{i=0, \dots, n-1} |target_i - position_0(A_i)|,$$

then the following proposition on a lower bound for the running time follows.

Proposition 2 *Any formation protocol has a running time at least equal to or larger than $v \cdot d_M$. Moreover, the total duration of all the motion phases of any formation-forming algorithm is at least $v \cdot d_M$.*

Next, we let $\delta > 0$ be the minimum distance there must be between any two robots to ensure safety from collision. We, therefore, do not accept initial configurations where robots are such that $d_r < \delta$, where

$$d_r = \min_{i \neq j \in \{0, \dots, n-1\}} |position_0(A_j) - position_0(A_i)|.$$

Thus, $position_0(A_{n-1}) - position_0(A_i) \geq n\delta$. Moreover, we also consider invalid any formation-forming algorithm that, during any t^{th} motion phase, moves an agent A_i at $position_t(A_i)$ in the direction of $position_t(A_{i-1})$ when $position_t(A_i) - position_t(A_{i-1}) = \delta$. That is, any two agents can not move concurrently, even if in the same direction, if they are separated by δ . This is because, if A_i (during such hypothetical motion phase) is moving in the direction of $position_t(A_{i-1})$, there is the risk that agent A_{i-1} breaks/shuts down and actually does not move. Thus, A_i would be moving beyond what is safe, and does not know/discover this until the next communication phase when A_{i-1} is not heard or A_{i-1} announces a new position at time $t + 1$ that is unsafe. Conversely, this also applies if the movement of A_{i-1} is in the direction of $position_t(A_i)$. Therefore, for each pair of robots, there needs to be a free space $\epsilon > 0$ such that the distance between these robots is $\delta + \epsilon$ and the total distance of the pair travelling towards each other in the corresponding round does not exceed ϵ . In summary, any formation-forming algorithm can only allow movement of robot A_i in the motion phase when there is free space to move into and that space has been computed at the time of the previous communication.

Now consider a hypothetical variation of the MTTIS algorithm that could vary the interval between communication rounds and (lets assume it has an oracle) carry out a communication phase in the exact moment the first robot that is moving has finished moving during a motion phase because it cannot safely move any further (but not because it has reached its final destination). Note that a robot A_i in the MTTIS algorithm stops moving during the motion phase only because of three reasons.

Destination. The agent A_i has reached its destination $target_i$, or

Duration. The agent A_i has exhausted the duration d_{MP} of the motion phase, or

Boundary. The agent A_i has reached the boundary of where it is safe to move in this motion phase.

We call this hypothetical variation OFF-LINE BY BOUNDARY MTTIS. This OFF-LINE BY BOUNDARY MTTIS algorithm may have some robots that do not move in a particular round, because there is no free space to move in the direction of their target (or some have already reached their destination).

Consider a scenario where no failures occur (there is no packet loss during communication, and every robot succeeds in travelling the corresponding distance in its motion phase). We can claim several optimality properties about the OFF-LINE BY BOUNDARY MTTIS algorithm in this no-failure scenario.

Proposition 3 *In the no-failure scenario, if in the initial configuration of the robots, the distance between*

any two robots is larger than δ , then the OFF-LINE BY BOUNDARY MTTIS algorithm has a total duration of its motion phase that is equal to the optimum total motion duration $v \cdot d_M$.

Proof: Because all robots are more than δ apart, none of them will start waiting in the OFF-LINE BY BOUNDARY MTTIS algorithm (unless it is at its final destination). Moreover, because we are in a no-failure scenario, even if robots stop (a safe distance from the position they learned their neighbour was in the previous COMMUNICATION_PHASE), they would find themselves at distances larger than δ in the next round. Thus, unless robots are at their destination, they will move again in the next MOTION_PHASE. Every robot not at their final destination will make progress at each MOTION_PHASE at speed v in the direction of their final destination. Since we are only concerned with the running time of the total MOTION_PHASE, such time equals $d_M \cdot v$. \square

The above proposition shows that if the cost of communication can be considered negligible (more precisely, communication duration $d_{CP} = 0$) then the OFF-LINE BY BOUNDARY MTTIS protocol is optimal.

However, in what follows we show an example of a strict two-phase version of the algorithm (i.e. communication and motion does not overlap), where the duration of communication is not zero. In such situation, there is a trade-off depending on the duration d_{MP} of the motion phase relative to the communication duration d_{CP} .

Input configuration 1 Consider $n = 10$ robots A_0, \dots, A_9 with $\text{position}_0(A_9) = 100$ and $\text{position}_0(A_0) = 0$. Also, let $\delta = 1$ be the minimum safe distance and suppose $\text{position}_0(A_8) = 99$, $\text{position}_0(A_7) = 98 - \epsilon$, $\text{position}_0(A_6) = 97 - 2\epsilon$, $\text{position}_0(A_5) = 96 - 3\epsilon$, and in general $\text{position}_0(A_i) = 100 - (n - i - 1) - (n - i - 2)\epsilon$, for $i = 1, \dots, 8$. The common speed of the robots is 1 unit per unit of time.

In Input configuration 1, the OFF-LINE BY BOUNDARY MTTIS protocol moves all robots for distance ϵ for as many rounds until A_8 reaches its final destination (at 88.88). Thus, the duration of the OFF-LINE BY BOUNDARY MTTIS protocol is at least $d_{CP} \cdot \#\text{communication rounds} + (99 - 88.88)v$. The number of communication rounds is $(99 - 88.88)/(v\epsilon)$. Thus, by choosing ϵ small enough (or the input configuration large enough so that ϵ units is still a move for all robots), one concludes the following.

Proposition 4 When the cost of communication is not zero, there are initial configurations in the no-failure scenario that result in unbounded running time for the OFF-LINE BY BOUNDARY MTTIS protocol (although the total

motion phase time is optimal, the communication cost is unbounded).

Now consider another version of off-line MTTIS where the duration of the motion phase is as long as it takes for the first of the moving robots to reach its final destination (we refer to this formation-forming variant as OFF-LINE BY DESTINATION MTTIS). Note that in any configuration of the robots, there is always at least one robot that can travel to its final position. In Input configuration 1, this version of the offline MTTIS protocol will incur one communication phase for each robot i from $i = 1, \dots, 8$ (a total of nine). The first duration phase would be until robot 1 reaches its position at 11.11; that is $\text{position}_0(A_1) - 11.11 = 92 - (7)\epsilon - 11.11$ time units. The second duration phase would be until robot 2 reaches its position at 22.22 and robot 2 only moved ϵ in the first round. Thus, the second round lasts

$$\text{position}_0(A_2) - \epsilon - 22.22 = 93 - (6)\epsilon - \epsilon - 22.22.$$

It is not hard to see that the total duration of the motion phases is at least

$$\sum_{i=1}^8 [91 + i - 7\epsilon - 11.11(i)].$$

This is

$$\begin{aligned} 8[91 - 7\epsilon] - 10.11 \sum_{i=1}^8 (i) &= 8[91 - 7\epsilon] - 10.11(8)9/2 \\ &= 8(91) - 7\epsilon - 8(91/2) \\ &= 4(91) - 7\epsilon. \end{aligned}$$

The fact that the total duration of the motion phase is bounded in the OFF-LINE BY DESTINATION MTTIS protocol and the total duration of its communication phase is also bounded would make it preferable to the OFF-LINE BY BOUNDARY MTTIS protocol (although the total duration of the motion phase is much worse).

5.2 Lower bounds on the communication phase

Any formation-forming algorithm that terminates in T rounds has a running time defined as follows.

$$\begin{aligned} \text{total running time} &= \sum_{t=1}^T [d_{CP}^{(t)} + d_{MP}^{(t)}] \quad (2) \\ &= T \cdot d_{CP} + \sum_{t=1}^T d_{MP}^{(t)}. \end{aligned}$$

The following proposition gives a lower bounds in the number T of rounds that multiplies the constant cost of the communication phase. Let $\text{blockers}_{\text{low}}(i)$ be

the number of agents whose initial position is between the final destination $target_i$ of agent A_i and the initial position $position_0(A_i)$ of agent A_i , if $target_i < position_0(A_i)$. That is,

$$\begin{aligned} blockers_{low}(i) &= \\ &= \|\{j | target_i \leq position_0(A_j) \wedge j < i\}\|. \end{aligned}$$

Similarly, we denote by $blockers_{high}(i)$ be the number of agents whose initial position is between the final destination $target_i$ of agent A_i and the initial position $position_0(A_i)$ of agent A_i , if $position_0(A_i) < target_i$.

$$\begin{aligned} blockers_{high}(i) &= \\ &= \|\{j | position_0(A_j) \leq target_i \wedge i < j\}\|. \end{aligned}$$

Proposition 5 *Any formation forming algorithm must perform a number of rounds that is at least*

$$\max_{i \in \{0, \dots, n-1\}} \{1, blockers_{low}(i), blockers_{high}(i)\}.$$

In fact, we can prove a stronger statement. Let $blockers(i, goal)$ be the number of agents between $position_0(A_i)$ and an arbitrary position $goal$, then any algorithm must perform at least $\max\{1, blockers(i, goal)\}$ communication rounds.

Proof: We prove, without loss of generality, the claim for when $target_i < position_0(A_i)$ (as the symmetric case $position_0(A_i) < target_i$ follows analogously). Thus, suppose $target_i < position_0(A_i)$.

The proof is by induction. Assume that there are no other agents between the initial position $position_0(A_i)$ and the destination $target_i$. At least one communication round would be required for robot A_i to find this out.

Assume that there is one agent (it must be robot A_{i-1}) between the initial position $position_0(A_i)$ of agent A_i and its destination $target_i$. There would be at least one round of communication by which robot A_i learns that it has free space up to $position_0(A_{i-1}) + \delta$. Since the boundary of this free space is larger than $target_i$, there would be at least a second round of communication before robot A_i can potentially find $target_i$ is within free space.

Now the inductive step. Assume there are $k = blockers_{low}(i)$ agents between $position_0(A_i)$ of agent A_i and its destination $target_i$. This implies that agent A_{i-1} has at least $k - 1 = blockers_{low}(i) - 1$ agents between itself and the a goal position $target_i$ it needs to pass by in order to get to $target_{i-1}$. By the induction hypothesis, at least k rounds would be required to clear all agents between agent A_{i-1} and the position $target_i$ (and possibly more). But this implies that at least $blockers_{low}(i)$ rounds would be required (possibly more) for agent A_i to establish free space to its final destination. \square

5.3 A competitive MTTIS

From the description of the cost (see Equation (3)) it is clear that the challenge is the balance between the cost d_{CP} of the communication phase and the choice of the duration for the motion phase. Suppose the duration of the motion phase is to be chosen as a constant for all rounds; then the analysis above on OFF-LINE BY BOUNDARY MTTIS suggest that the duration of the motion phase should approximate the duration of the communication phase from above (OFF-LINE BY BOUNDARY MTTIS is best when the duration of the motion phase is much larger than d_{CP}). However, Input configuration 1 shows a case when very little parallel movement is desirable, in favour of sporadic communication. In a sense, the duration of the motion phase approximates the duration of the communication phase from below. We use this as intuition to suggest that the duration of the motion phase should be set to the duration of the communication phase.

In this section we show that if $d_{MP}^{(t)} = d_{CP}$, then we have a competitive algorithm that is off-line (there is no oracle anymore). Let $d = v \cdot d_{CP}$ be the distance a robot travels in a motion phase whose duration is the duration of the communication phase.

Proposition 6 *In the no-failure scenario, if in the initial configuration of the robots, the distance between any two robots is larger than $d + \delta$, then the MTTIS algorithm with $d_{MP}^{(t)} = d_{CP}$ has a competitive running time.*

Proof: Because the distance between any two robots is $d + \delta$, and we are in the no-failure scenario, every robot has free space during the motion phase to carry out a motion until it reaches its destination or until the duration of the motion phase expires. Therefore, all robots are always moving in parallel, and in a similar manner to the OFF-LINE BY BOUNDARY MTTIS, the total duration of the motion phases across all rounds is optimal. In particular, the total duration of the motion phases is $d_{MP} \cdot d_M/d$; and since $d_{MP} = d_{CP}$ we have the total duration of all morion phases is $d_{CP} \cdot d_M/d$.

We now analyse the cost of the communication phase. Because all robots move d units in each round, the total number of rounds T used by the algorithm in this case is bounded by d_M/d . And thus, the total time spent in communication is $d_{CP} \cdot d_M/d$. Therefore, even if an off-line algorithm did not spend any time in communication, it would need to spend $v \cdot d_M = (d_{CP}/d)d_M$ time. Therefore, the total running time of the on-line MTTIS protocol with $d_{MP}^{(t)} = d_{CP}$ is no more than twice the best possible off-line algorithm. \square

To complete the proof that MTTIS is competitive in all cases we now proceed to analyse the situation

when the initial configuration has at least two consecutive robots who are closer than $d + \delta$. Clearly, the monotonicity of MTTIS ensures that if we reach a configuration as needed by Proposition 6, then the minimum separation is maintained (in the no-failure scenario). Therefore, it suffices to show that when robots are clustered together, they get separated by the MTTIS protocol in competitive time if $d_{MP}^{(t)} = d_{CP}$. This task is accomplished by the following proposition. We say we have a $(d + \delta)$ -cluster of size k in a configuration if we have $A_i, A_{i+1}, \dots, A_{i+k-1}$ agents so that $position_0(A_{i+j}) - position_0(A_{i+j-1}) < d + \delta$ for $j = 1, \dots, k-1$, $position_0(A_i) - position_0(A_{i-1}) \geq d + \delta$ and $position_0(A_{i+k}) - position_0(A_{i+k-1}) \geq d + \delta$.

Proposition 7 *Let $A_i, A_{i+1}, \dots, A_{i+k-1}$ be the largest $(d + \delta)$ -cluster in an initial configuration. Then, after k rounds of the MTTIS algorithm with $d_{MP}^{(t)} = d_{CP}$, the cluster has disappeared.*

Proof: The proof is similar to the proof of Proposition 2. If the cluster has only 2 agents, clearly in the next round phase, one of them will be able to travel d in the motion phase and dissipate the cluster.

For the induction step, if the cluster has $k + 1$ agents, at least one of agents on the side of the cluster (if not two, but the possibility exists that the cluster is at one extreme of the interval $[target_0, target_{n-1}]$) will travel d in the next round during the motion phase, reducing the cluster by one. But then, by the induction hypothesis, the cluster of size k will be dissolved in k steps. \square

Using this proposition, it is now feasible to prove the competitiveness of MTTIS when $d_{MP}^{(t)} = d_{CP}$. Consider all of the intervals of the form $[target_i, target_j]$ with $i < j$ and find the one that has the largest $d + \delta$ -cluster (again, if there is no cluster, Proposition 6 shows competitiveness). If the largest cluster is of size k and is in an interval $[target_{i_c}, target_{j_c}]$ with $i_c < j_c$, then by Proposition 2 any formation-forming algorithm would have to perform k rounds and a communication cost of at least $k \cdot d_{CP}$. In k rounds, the MTTIS protocol would incur a total motion cost of $k \cdot d_{MP}^{(t)} = k \cdot d_{CP}$ and an equal total communication cost. Thus MTTIS with $d_{MP}^{(t)} = d_{CP}$ is no worse than twice any (hypothetical) communicationless offline formation-forming protocol.

6 Illustrating the convergence rate

We have already indicated that MO2N requires a clear separation between the COMMUNICATION_PHASE and the MOTION_PHASE. The previous section shows that under this clear separation, MTTIS has the desirable property of competitiveness. But we have already indicated that, unlike MO2N, MTTIS is not constrained by this restriction. In fact, during MTTIS's

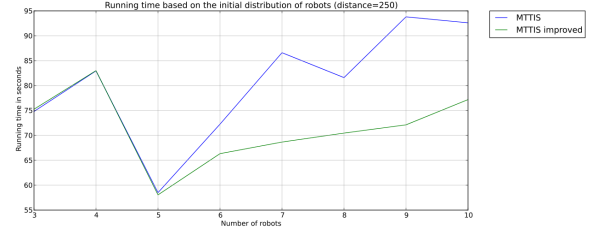


Figure 6: Under initial position uniformly distributed, the running time of MTTIS algorithm is improved by allowing the free space learned under the COMMUNICATION_PHASE to be enlarged during the MOTION_PHASE.

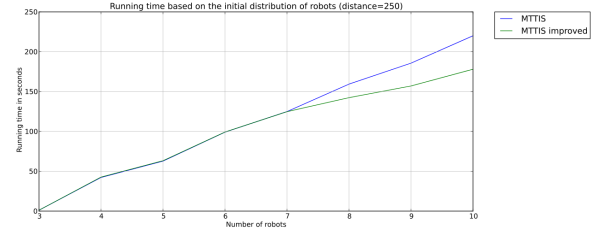


Figure 7: The running time of MTTIS algorithm is improved by allowing the free space learned under the COMMUNICATION_PHASE to be enlarged during the MOTION_PHASE. Data comes from a Gaussian distribution.

MOTION_PHASE, a robot may learn that the safe and free space it was certain about has become even larger as other robots may also have moved. Clearly, the suggestion to enlarge the destination, if larger free space is discovered during a move, does not undermine any of the theoretical properties. In the experiments that follow, we show that this idea of MTTIS creates further improvements. We implement this idea in the simplest of modalities. During the MOTION_PHASE, any robot A_i in between a neighbouring robot $A_{i\pm 1}$ and the destination $target_{i\pm 1}$, broadcasts when it vacates the interval between the last known $position_{i\pm 1}$ and the $target_{i\pm 1}$ (this happens when it goes past $target_{i\pm 1}$).

The simulation of this modality immediately produces faster running times. Fig. 6 shows the results with an initial configuration drawn out of a uniform distribution. Even as robots are, on average, equally spaced, we have significant improvement with this modest variation (note the variation could be introduced in every round as we already alluded to, but we are here minimising bandwidth usage). Clearly, the more robots, the bigger the opportunity for this improvement. Fig. 7 also shows the improvement as the number of robots increases, but here it is more infrequent that a robot clears the space it was

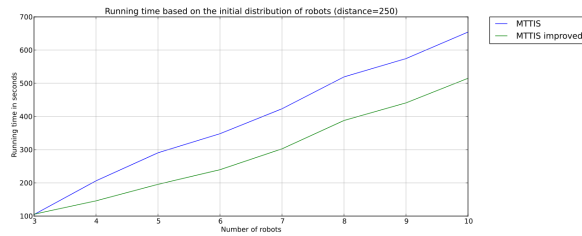


Figure 8: The running time of the MTTIS algorithm is improved by allowing the free space learned under the COMMUNICATION_PHASE to be enlarged during the MOTION_PHASE. Data comes from an initial Gamma distribution.

blocking for a neighbour. Again, the more robots, the larger the impact of the improvement. Finally, Fig. 8 shows that the improvement is truly effective in this case, as immediate neighbours clear space for others much more frequently.

7 Reflection

Creating formations has been studied in the literature of agents and applied to agents in games [Millington and Funge, 2009]. They are usually based on some modelling of physical properties. Models of systems of forces, for example, use an attraction-repulsion model. If there is a spring between agent A_i and A_{i+1} , for $i = 0, n-2a$, then a point of equilibria is a formation in a line with equal spacing. These methods have been used in the graph drawing community for a long time [Di Battista *et al.*, 1994], and more recently in the robotics community [Cifuentes *et al.*, 2012]. The criteria by which the formation is structured has not been the main interest here (but our principles apply universally). We are more interested in the mathematical verification and formal properties of the algorithm as each robot schedules time-slices for communication or sensing of positions of others with the performing of its own movement actions.

In real applications one would expect the duration d_{CP} of the COMMUNICATION_PHASE to be much shorter than the duration d_{MP} of the MOTION_PHASE. A motion may require enabling and powering the motors, overcoming inertia, and many other aspects that, in robotics, may take much longer than communication. However, in some environments, communication may actually be lengthy as well. At RoboCup, interference from many Wi-Fi networks results in significant packet loss and significant delays. This paper has focussed on the fundamental properties and mathematical verification of the protocols, while keeping regard for practice beyond what previously has been the case.

In particular, one aspect that this work has left untouched is that the communication during the communi-

cation phase may be sufficient to dynamically set up the duration d_{MP}^t of the next MOTION_PHASE (rather than set them up all uniform). Our work here clearly suggests that when robots are well separated, it is advantageous to actually have long motions and there would be little need for communication; while when robots are clustered, frequent communication seems suitable. However, the MOTION_PHASE should actually be long enough (at least as long as the communication phase) to cause someone on the cluster to spread out (even if there is no parallel movement).

A video of a simulation scenarios using the Webots simulator is available at <http://youtu.be/T3WcKntnlb4>. The simulation includes randomly shutting down a robot and a high level of packets being lost.

References

- [Borodin and El-Yaniv, 1998] Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. Cambridge University Press, New York, NY, USA, 1998.
- [Chandy *et al.*, 2008] K. M. Chandy, S. Mitra, and C. Pilotto. Convergence verification: From shared memory to partially synchronous systems. In F. Cassez and C. Jard, editors, *Formal Modeling and Analysis of Timed Systems, 6th International Conference, FORMATS*, volume 5215 of *Lecture Notes in Computer Science*, pages 218–232, Saint Malo, France, September 15th–17th 2008. Springer.
- [Chandy *et al.*, 2011] K. M. Chandy, B. Go, S. Mitra, C. Pilotto, and J. White. Verification of distributed systems with local-global predicates. *Formal Asp. Comput.*, 23(5):649–679, 2011.
- [Cifuentes *et al.*, 2012] S. Cifuentes, J. M. Giron-Sierra, and J. F. Jiménez. Robot navigation based on discrimination of artificial fields: Application to robot formations. *Advanced Robotics*, 26(5-6):627–652, 2012.
- [Di Battista *et al.*, 1994] G. Di Battista, P. Eades, R. Tamassia, and I.G. Tollis. Algorithms for drawing graphs: an annotated bibliography. *Comput. Geom.*, 4:235–282, 1994.
- [Kopetz and Bauer, 2003] H. Kopetz and G. Bauer. The time-triggered architecture. *Proceedings of the IEEE*, 91(1):112–126, 2003.
- [Millington and Funge, 2009] I. Millington and J. Funge. *Artificial Intelligence for Games, Second Edition*. Morgan Kaufmann, 2009.
- [Olson *et al.*, 2013] E. Olson, J. H. Strom, R. Goedel, R. D. Morton, P. Pradeep Ranganathan, and A. Richardson. Exploration and mapping with autonomous robot teams. *Commun. ACM*, 56(3):62–70, 2013.