Progress on a Perimeter Surveillance Problem

Jeremy Avigad [1]   Floris van Doorn [2]

[1]Carnegie Mellon University

[2]University of Pittsburgh

AFOSR Review Meeting
August 4, 2020

Perimeter surveillance

A trend in control theory addresses problems of this form:

- A collection of agents (small, medium, or large) is trying to coordinate.
- Each agent acts individually.
- Information is communicated between them.
- They want to achieve some global goal.

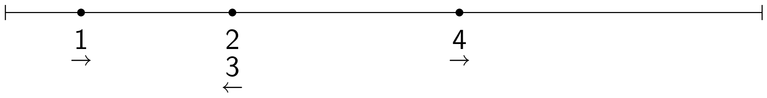
Decentralized control is a catchphrase. So is *UAV (unmanned aerial vehicle)*, or *drone*.

In 2008, Kingston, Beard, and Holt published a paper, "Decentralized Perimeter Surveillance Using a Team of UAVs," based on Kingston's 2007 dissertation.

Perimeter surveillance

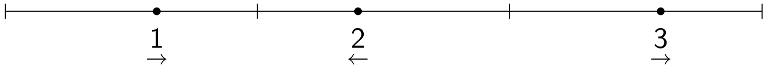
A simple model:

- n drones are "surveilling" the unit interval $[0, 1]$.
- They all move with speed one unit per unit time in either direction.
- Each one has an estimate $((a, m), (b, n))$ where:
  - a is the left border of the interval
  - m is the number of drones to the left
  - b is the right border
  - n is number of drones to the right
- The data keeps changing, so it might be wrong.
- They recognize the border when they hit it.
- They can only share information when they meet.



Perimeter surveillance

The ideal behavior: divide $[0, 1]$ into n equal intervals.


$\underset{\rightarrow}{1}$
$\underset{\leftarrow}{2}$
$\underset{\rightarrow}{3}$

Each drone has the correct data, and goes back and forth between the endpoints of its interval.

Kingston, Beard, and Holt were looking for an online algorithm that would behave reasonably if the borders or number of drones changes.

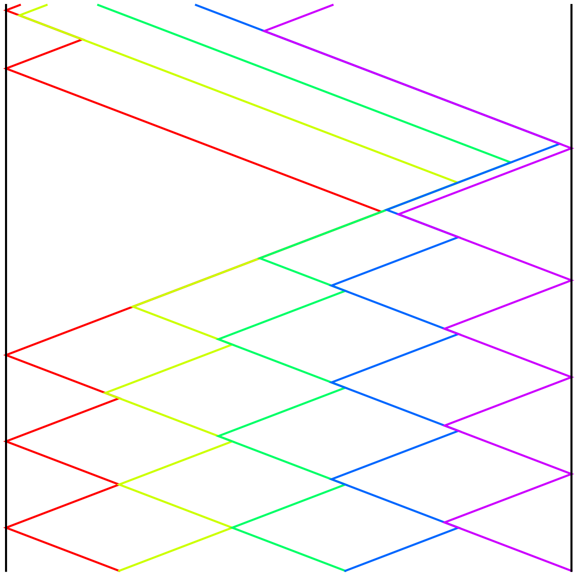
They proposed an algorithm to attain (and maintain) the ideal behavior, and studied its properties.

The algorithm

Drones continue in the direction they are going until there is an "event":

- If a drone hits the left border, it updates its left information and turns around, and similarly for the right border.
- If two drones meet, the left one adopts the right information from the one to the right, and vice-versa, and head towards the resulting common endpoint.
- If a group of drones is traveling together, when it reaches the common endpoint, they split.

The algorithm



The question

Suppose the drones in some configuration in [0,1] and follow the algorithm.

What is the longest amount of time it can take to achieve synchronization?

KBH divided the behavior of the algorithm into two phases, and claimed:

- Within 3 units of time, all the drones have the correct information.
- Once they have the correct information, they are all synchronized within 2 units of time.

In each case, they gave an argument based on a hypothesized worst case.

The fly in the ointment

Davis, Humphrey, and Kingston, “When Human Intuition Fails: Using Formal Methods to Find an Error in the ‘Proof’ of a Multi-agent Protocol” (2019) used formal methods to test the claims.

Results:

- They provided counterexamples to the purported bound on phase 1, with $n = 3$, in which the drones don’t all have correct information until time $3\frac{1}{2}$.
- For $n = 3$, their tool gave an upper bound of $3\frac{2}{3}$ (though they had to limit the number of drones in their estimates).

At that point, there was no rigorous upper bound on either phase that is independent of n, nor on the total time to synchronization.

Our results

Floris van Doorn and I have made some progress.

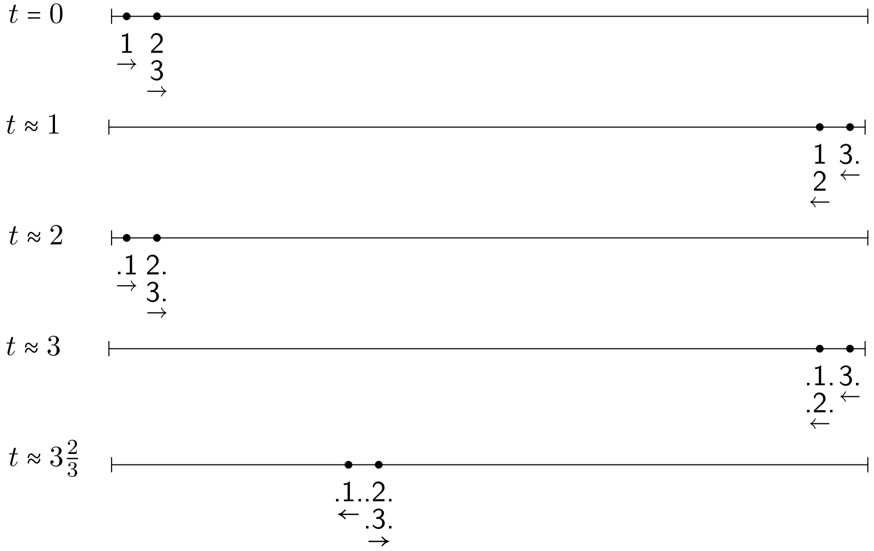
- We have improved the lower bounds:
    - For $n \geq 3$, it can take up to $4 - 1/n$ units of time before all the drones have correct information.
    - For $n \geq 3$, it can take up to $5 - 3/n$ units of time before all the drones are fully synchronized.
- We proved sharp upper bounds on phase 2:
    - Once all the drones have complete information, all drones are synchronized within $2 - 1/n$ units of time.
    - This bound is sharp (and KBH got the worst-case behavior right).
- We have made progress towards proving upper bounds on phase 1.

Lower bounds

We prove the lower bounds with explicit counterexamples.

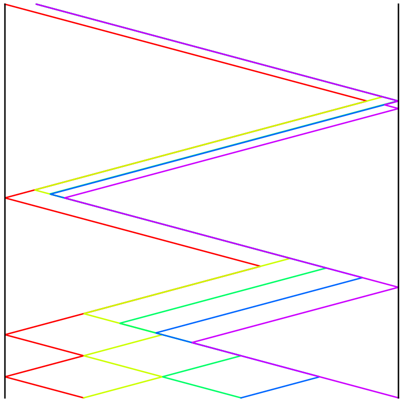
They were discovered interactively with Mathematica, fixing various parameters and using its *Solve* function.

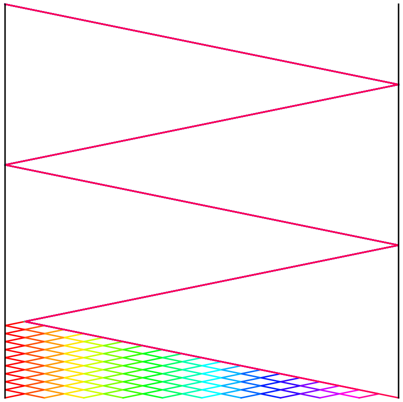
Lower bounds



Lower bounds

Near worst-case examples with $n = 5$ and $n = 20$.





Remember: correct information at $4 - 1/n$, synchronization at $5 - 3/n$.

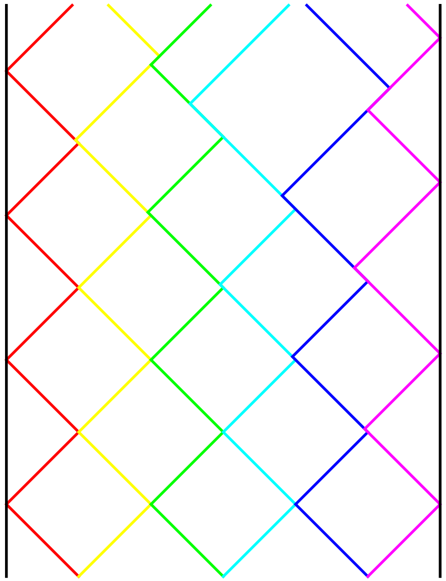
Upper bounds for phase 2

Theorem

Assuming all the drones have correct information, they are all synchronized at time $2 - 1/n$*.*

The algorithm simplifies: when two drones meet, they travel together to their common endpoint, and split.

Upper bounds for phase 2



Upper bounds for phase 2

Why is this so hard?

Ideas:

- Show that some measure is decreasing.
- Show that configurations can be “reduced” to simpler ones.
- Show that configurations can be reduced to simpler ones in a larger state space.
- Use induction on the number of drones.
- Focus on some salient feature of the history.

None of these seem to work.

Upper bounds for phase 2

The trick: the behavior of the a consecutive pair of drones is much more constrained once they have met at least once.

Lemma

For every $j < n$, drones j and $j + 1$ have met by time 1.

Lemma

Once they have met, they synchronize within time $1 - 1/n$.

It was surprising to us that this worked. You ignore almost everything about the first unit of time, but then convergence is quick.

Upper bounds for phase 2

A time t, say a drone is

- *left synchronized* if it never goes to the left of its left endpoint after time t, and
- *right synchronized* if it never goes to the right of its right endpoint.

A drone is *synchronized* at time t if it is left and right synchronized.

It suffices to show all the drones are left synchronized by time $2 - 1/n$.

Lemma

Let $j < n$. Suppose that at time t, drones $1, \ldots, j$ are left synchronized and drone j and $j + 1$ have met. Then at time $t + 1/n$, drone $j + 1$ is left synchronized as well.

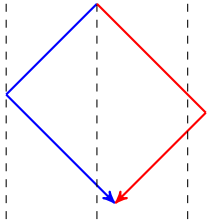
Upper bounds for phase 2

To prove that key lemma, we rely on this observation:

Lemma

Suppose at time t, drone j is left synchronized and drone j and $j+1$ separate at their common endpoint. Then drone $j+1$ is left synchronized.

The idea:



KBH use this to show that eventually all the drones are left synchronized.

Upper bounds for phase 2

Lemma

Suppose $j < n$ and at time t, drones $1, \dots, j$ are left synchronized and drone j and $j + 1$ have met. Then at time $t + 1/n$, drones $j + 1$ is left synchronized as well.

Proof (sketch). Suppose drone j is left synchronized.

- If it is moving to the right, it will be at its right endpoint within time $1/n$. At that point drone $j + 1$ is left synchronized.
- If drone j is moving to the left and it is together with drone $j + 1$, drone $j + 1$ is left synchronized at time t.
- If drone j is moving to the left and it is not together with drone $j + 1$, we can use the fact that they have already met to show that drone j will return to its right endpoint before drone $j + 1$, so drone $j + 1$ is left synchronized at time t.

Towards an upper bound for phase 1

Bounding phase 1 is harder than bounding phase 2, because drones exchange information and so the estimates change.

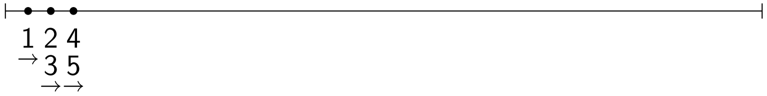
Write $(\alpha, \beta) = ((a, m), (b, n))$ for position information.

For a drone with this information:

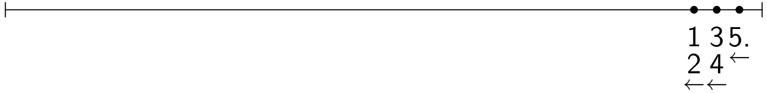
- The size of its interval is
$$I(\alpha, \beta) = \frac{b - a}{m + n + 1}.$$
- Its left endpoint is
$$a + mI(a, b) = b - (n + 1)I(a, b) = \frac{a(n + 1) + bm}{m + n + 1}.$$
- Its right endpoint is
$$a + (m + 1)I(a, b) = b - nI(a, b) = \frac{an + b(m + 1)}{m + n + 1}.$$

Towards an upper bound for phase 1

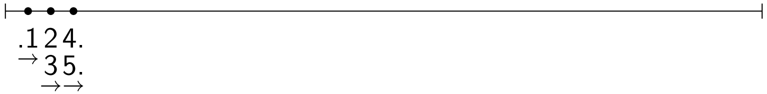
Imagine this situation. Five drones start to the left:



The pairs think their common border is all the way to the right:

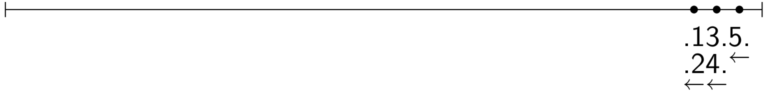


Now the pairs think their common border is to the left:

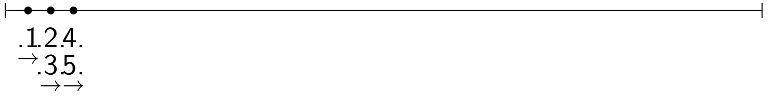


Towards an upper bound for phase 1

Now the pairs think their common endpoint is all the way to the right:



Now pairs think their common endpoint is all the way to the left:



Are there values of $((a, m), (b, n))$ for each drone that can bring this about?

We have an algebraic calculation that limits this sort of behavior.

Towards an upper bound for phase 1

Note that if drones with information

$$((a_0, m_0), (b_0, n_0)) \quad \text{and} \quad ((a_1, m_1), (b_1, n_1))$$

meet, the updated positions are

$$((a_0, m_0), (b_1, n_1 + 1)) \quad \text{and} \quad ((a_0, m_0 + 1), (b_1, n_1)).$$

These $+1$s are messy! Let's assume the ms and the n are large.

If $\alpha = (a, m)$ and $\beta = (b, n)$, (re)define

$$I(\alpha, \beta) = \frac{b - a}{m + n}$$

and

$$P(\alpha, \beta) = a + mI(\alpha, \beta) = b - nI(\alpha, \beta) = \frac{am + bn}{m + n}.$$

Towards an upper bound for phase 1

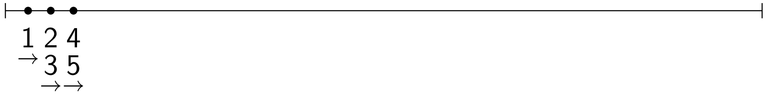
Suppose the drones start (roughly) with information

$$(\alpha_0, \beta_0), (\alpha_1, \beta_1), (\alpha_2, \beta_2).$$

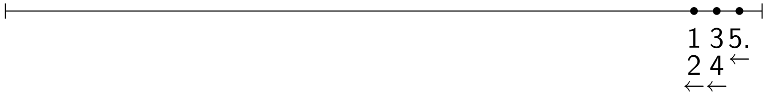
Let's look at what happens after the updates.

Towards an upper bound for phase 1

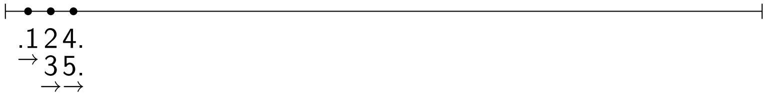
Starting estimates: $(\alpha_0, \beta_0), (\alpha_1, \beta_1), (\alpha_2, \beta_2)$



Estimates: $(\alpha_0, \beta_1), (\alpha_1, \beta_2), (\alpha_2, 1)$



Estimates: $(0, \beta_1), (\alpha_0, \beta_2), (\alpha_1, 1)$



Towards an upper bound for phase 1

Is it possible to have:

- $P(\alpha_0, \beta_0)$, $P(\alpha_1, \beta_1)$, and $P(\alpha_2, \beta_2)$ close to 1,
- $P(\alpha_0, \beta_1)$ and $P(\alpha_1, \beta_2)$ close to 0, and
- $P(\alpha_0, \beta_2)$ close to 1?

Lemma

Suppose

$$\max(P(\alpha_0, \beta_1), P(\alpha_1, \beta_2)) \leq \min(P(\alpha_1, \beta_1), P(\alpha_0, \beta_2))$$

then

$$P(\alpha_0, \beta_1) = P(\alpha_1, \beta_2) = P(\alpha_1, \beta_1) = P(\alpha_0, \beta_2)$$

Towards an upper bound for phase 1

We have three proofs of this lemma.

One, suggested to us by Reid Barton, interprets the data as weights and centers of mass.

With the $+1$s, however, the lemma is false.

Conclusions

This problem is *evil*.

It is a lot harder than we thought it would be.

We are hopeful that the method used to bound the phase 2 behavior, plus some algebraic specifics, will enable us to bound phase 1 behavior.

We think we are learning interesting things about how to reason about mixtures of continuous and discrete behavior.

Conclusions

We have formalized a statement of the upper bound on phase 2 in Lean:

```
def synchronized_positions
  (start : ∀ i : fin n, drone_info i) (t : ℝ) : Prop :=
∀ i : fin n, is_left_synchronized start i t ∧
   is_right_synchronized start i t

theorem upper_bound_phase_2 (st : valid_state n) :
  synchronized_positions
    (st.to_drone_info 0) (2 - 1 / n) :=
by sorry
```

It deserves a formal proof.