When can multi-agent rendezvous be executed in time linear in the diameter of a plane configuration ?

Presenting: Peter Hegarty
Co-authors: Anders Martinsson, Dmitry Zhelezov

Department of Mathematics, Chalmers/Gothenburg University, Sweden

Wednesday, 05 January, 2016

- Points in the plane represent idealized mobile agents.

- Points in the plane represent idealized mobile agents.
- Their goal is to **rendezvous**, that is, meet up at a single point.

- Points in the plane represent idealized mobile agents.
- Their goal is to **rendezvous**, that is, meet up at a single point.

The following are the “classical” assumptions:

- Points in the plane represent idealized mobile agents.
- Their goal is to **rendezvous**, that is, meet up at a single point.

The following are the “classical” assumptions:

- **RP-1:** Identical agents (capabilities).

- Points in the plane represent idealized mobile agents.
- Their goal is to **rendezvous**, that is, meet up at a single point.

The following are the “classical” assumptions:

- **RP-1:** Identical agents (capabilities).
- **RP-2:** Fixed maximum speed of motion.

- Points in the plane represent idealized mobile agents.
- Their goal is to **rendezvous**, that is, meet up at a single point.

The following are the “classical” assumptions:

- **RP-1:** Identical agents (capabilities).
- **RP-2:** Fixed maximum speed of motion.
- **RP-3:** Fixed visibility range.

- Points in the plane represent idealized mobile agents.
- Their goal is to **rendezvous**, that is, meet up at a single point.

The following are the "classical" assumptions:

- **RP-1:** Identical agents (capabilities).
- **RP-2:** Fixed maximum speed of motion.
- **RP-3:** Fixed visibility range.
- **RP-5:** No (active) communication between agents.

- Points in the plane represent idealized mobile agents.
- Their goal is to **rendezvous**, that is, meet up at a single point.

The following are the “classical” assumptions:

- **RP-1:** Identical agents (capabilities).
- **RP-2:** Fixed maximum speed of motion.
- **RP-3:** Fixed visibility range.
- **RP-5:** No (active) communication between agents.

Now if there were a global point of reference, then **RP-2** would be the only real constraint and rendezvous could be executed in time bounded by the diameter of the configuration.

- Points in the plane represent idealized mobile agents.
- Their goal is to **rendezvous**, that is, meet up at a single point.

The following are the "classical" assumptions:

- **RP-1:** Identical agents (capabilities).
- **RP-2:** Fixed maximum speed of motion.
- **RP-3:** Fixed visibility range.
- **RP-5:** No (active) communication between agents.

Now if there were a global point of reference, then **RP-2** would be the only real constraint and rendezvous could be executed in time bounded by the diameter of the configuration.

Motivating Question: How well can we do in the absence of an a priori global reference point ?

- If agents must act locally, then it is reasonable to also assume

  RP-4: The visibility graph is initially connected.

- If agents must act locally, then it is reasonable to also assume

  RP-4: The visibility graph is initially connected.
- The number N of agents is a lower bound, up to a universal constant factor, for any procedure which takes account of time for computations, since eventually each agent will have to take the locations of all others into account.

- If agents must act locally, then it is reasonable to also assume
  RP-4: The visibility graph is initially connected.
- The number N of agents is a lower bound, up to a universal constant factor, for any procedure which takes account of time for computations, since eventually each agent will have to take the locations of all others into account.
  It is common in the literature to impose **synchronization** and consider only the number of "rounds" of computation.

- If agents must act locally, then it is reasonable to also assume

  RP-4: The visibility graph is initially connected.
- The number N of agents is a lower bound, up to a universal constant factor, for any procedure which takes account of time for computations, since eventually each agent will have to take the locations of all others into account.

  It is common in the literature to impose **synchronization** and consider only the number of "rounds" of computation.
- It seems intuitively plausible that, for **generic configurations**, even very simple algorithms might perform well.

- If agents must act locally, then it is reasonable to also assume

  RP-4: The visibility graph is initially connected.
- The number N of agents is a lower bound, up to a universal constant factor, for any procedure which takes account of time for computations, since eventually each agent will have to take the locations of all others into account.

  It is common in the literature to impose **synchronization** and consider only the number of "rounds" of computation.
- It seems intuitively plausible that, for **generic configurations**, even very simple algorithms might perform well.
- For example, let us consider a generic configuration of points in a **disc**, i.e.: the points are initially placed uniformly and independently at random in the disc.

- If agents must act locally, then it is reasonable to also assume
  RP-4: The visibility graph is initially connected.
- The number N of agents is a lower bound, up to a universal constant factor, for any procedure which takes account of time for computations, since eventually each agent will have to take the locations of all others into account.
  It is common in the literature to impose **synchronization** and consider only the number of "rounds" of computation.
- It seems intuitively plausible that, for **generic configurations**, even very simple algorithms might perform well.
- For example, let us consider a generic configuration of points in a **disc**, i.e.: the points are initially placed uniformly and independently at random in the disc.
  Note that, if the disc has radius r, then **RP-4** holds asymptotically almost surely (a.a.s.) iff $N = \Omega(r^2 \log r)$.

- A very simple and natural algorithm is: In each round, every point moves to the average of the locations of its current neighbours.

- A very simple and natural algorithm is: In each round, every point moves to the average of the locations of its current neighbours.
  This is the so-called **Hegselmann-Krause (HK) dynamics**.

- A very simple and natural algorithm is: In each round, every point moves to the average of the locations of its current neighbours.
  This is the so-called **Hegselmann-Krause (HK) dynamics**.
- It is known (there are simple examples) that the HK-dynamics can disconnect the visibility graph, but simulations suggest that for generic connected configurations on a disc

- A very simple and natural algorithm is: In each round, every point moves to the average of the locations of its current neighbours.
  This is the so-called **Hegselmann-Krause (HK) dynamics**.
- It is known (there are simple examples) that the HK-dynamics can disconnect the visibility graph, but simulations suggest that for generic connected configurations on a disc

  (1): Rendezvous (consensus) is a.a.s. achieved.

- A very simple and natural algorithm is: In each round, every point moves to the average of the locations of its current neighbours.
  This is the so-called **Hegselmann-Krause (HK) dynamics**.
- It is known (there are simple examples) that the HK-dynamics can disconnect the visibility graph, but simulations suggest that for generic connected configurations on a disc

  (1): Rendezvous (consensus) is a.a.s. achieved.
  (2): The expected number of rounds scales as the **area** of the disc.

- A very simple and natural algorithm is: In each round, every point moves to the average of the locations of its current neighbours.
  This is the so-called **Hegselmann-Krause (HK) dynamics**.
- It is known (there are simple examples) that the HK-dynamics can disconnect the visibility graph, but simulations suggest that for generic connected configurations on a disc

  (1): Rendezvous (consensus) is a.a.s. achieved.
  (2): The expected number of rounds scales as the **area** of the disc.
- To prove rigorously either of these statements seems to be a very subtle problem and would be a big breakthrough.

- A very simple and natural algorithm is: In each round, every point moves to the average of the locations of its current neighbours.
  This is the so-called **Hegselmann-Krause (HK) dynamics**.
- It is known (there are simple examples) that the HK-dynamics can disconnect the visibility graph, but simulations suggest that for generic connected configurations on a disc

  (1): Rendezvous (consensus) is a.a.s. achieved.
  (2): The expected number of rounds scales as the **area** of the disc.
- To prove rigorously either of these statements seems to be a very subtle problem and would be a big breakthrough.
- Is it possible to beat the area, that is the **square** of the diameter, for generic configurations on a disc ?

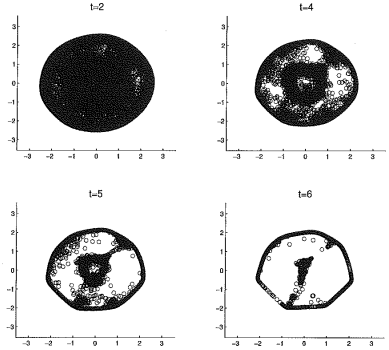

Figure 5: An evolution of 25600 uniformly i.i.d agents on a disc with area 40. At $t = 5$ we see that two "circles" have formed inside one another.

- An alternative approach is to see by how much the classical rendezvous assumptions need to be weakened before we can obtain a linear-in-diameter time algorithm.

- An alternative approach is to see by how much the classical rendezvous assumptions need to be weakened before we can obtain a linear-in-diameter time algorithm.
- Suppose we had global visibility (RP-3 completely removed). Then there seem to be quite natural ways to accomplish rendezvous efficiently, if we also allow

- An alternative approach is to see by how much the classical rendezvous assumptions need to be weakened before we can obtain a linear-in-diameter time algorithm.

- Suppose we had global visibility (RP-3 completely removed). Then there seem to be quite natural ways to accomplish rendezvous efficiently, if we also allow

  (1): agents can generate random bits **(RP-6)**

- An alternative approach is to see by how much the classical rendezvous assumptions need to be weakened before we can obtain a linear-in-diameter time algorithm.

- Suppose we had global visibility (RP-3 completely removed). Then there seem to be quite natural ways to accomplish rendezvous efficiently, if we also allow

  (1): agents can generate random bits **(RP-6)**
  (2): they have a simple **signalling** mechanism, e.g.: 1 = green, 0 = blue. This is a weakening of RP-5, we allow some rudimentary communication using a finite and bounded number of **colour signals (frequencies)**.

- An alternative approach is to see by how much the classical rendezvous assumptions need to be weakened before we can obtain a linear-in-diameter time algorithm.

- Suppose we had global visibility (RP-3 completely removed). Then there seem to be quite natural ways to accomplish rendezvous efficiently, if we also allow

  (1): agents can generate random bits **(RP-6)**
  (2): they have a simple **signalling** mechanism, e.g.: $1 =$ green, $0 =$ blue. This is a weakening of RP-5, we allow some rudimentary communication using a finite and bounded number of **colour signals (frequencies)**.

  Then the algorithm would essentially be:

(1): everyone signal green

(1): everyone signal green
(2): generate random bits and change your signal to blue once you generate a zero.

(1): everyone signal green
(2): generate random bits and change your signal to blue once you generate a zero.
(3): as long as you remain green, **scan** for other greens

(1): everyone signal green
(2): generate random bits and change your signal to blue once you generate a zero.
(3): as long as you remain green, **scan** for other greens
(4): if you're still green and see no other green, then turn red to indicate you're the **leader = global reference point**

(1): everyone signal green
(2): generate random bits and change your signal to blue once you generate a zero.
(3): as long as you remain green, **scan** for other greens
(4): if you're still green and see no other green, then turn red to indicate you're the **leader = global reference point**
(5): follow the leader, i.e.: everyone move to his location.

(1): everyone signal green
(2): generate random bits and change your signal to blue once you generate a zero.
(3): as long as you remain green, **scan** for other greens
(4): if you're still green and see no other green, then turn red to indicate you're the **leader = global reference point**
(5): follow the leader, i.e.: everyone move to his location.

- After fixing some technicalities, it is easy to show that a unique leader will a.a.s. emerge after $O(\log N)$ rounds of bit generation.

(1): everyone signal green
(2): generate random bits and change your signal to blue once you generate a zero.
(3): as long as you remain green, **scan** for other greens
(4): if you're still green and see no other green, then turn red to indicate you're the **leader = global reference point**
(5): follow the leader, i.e.: everyone move to his location.

- After fixing some technicalities, it is easy to show that a unique leader will a.a.s. emerge after $O(\log N)$ rounds of bit generation. So this will be negligible compared to the walking time in step (5), because of RP-2, as long as we don't have an extremely dense configuration, i.e.: $N = o(e^r)$.

Let us summarize the new assumptions:

Let us summarize the new assumptions:

- Agents are capable of generating random bits **(RP-6)**

Let us summarize the new assumptions:

- Agents are capable of generating random bits **(RP-6)**
- We weaken RP-5 to allow the kinds of interaction referred to as **signalling** and **scanning** above.

Let us summarize the new assumptions:

- Agents are capable of generating random bits **(RP-6)**
- We weaken RP-5 to allow the kinds of interaction referred to as **signalling** and **scanning** above.
  However, we really want to retain bounded visibility (RP-3), it is the essence of this distributed control problem.

Let us summarize the new assumptions:

- Agents are capable of generating random bits **(RP-6)**
- We weaken RP-5 to allow the kinds of interaction referred to as **signalling** and **scanning** above.
  However, we really want to retain bounded visibility (RP-3), it is the essence of this distributed control problem.
  Hence, we also permit a third capability called **tracking** by which agents with bounded visibility can all locate and move towards a chosen "leader".

Let us summarize the new assumptions:

- Agents are capable of generating random bits **(RP-6)**
- We weaken RP-5 to allow the kinds of interaction referred to as **signalling** and **scanning** above.
  However, we really want to retain bounded visibility (RP-3), it is the essence of this distributed control problem.
  Hence, we also permit a third capability called **tracking** by which agents with bounded visibility can all locate and move towards a chosen "leader".
  In the previous example, replace the last step (5) by:
  (5)*: If you see a neighbour signalling red, then signal red yourself and track the movements of those neighbours who signalled red before you.

Let us summarize the new assumptions:

- Agents are capable of generating random bits **(RP-6)**
- We weaken RP-5 to allow the kinds of interaction referred to as **signalling** and **scanning** above.
  However, we really want to retain bounded visibility (RP-3), it is the essence of this distributed control problem.
  Hence, we also permit a third capability called **tracking** by which agents with bounded visibility can all locate and move towards a chosen "leader".
  In the previous example, replace the last step (5) by:
  (5)*: If you see a neighbour signalling red, then signal red yourself and track the movements of those neighbours who signalled red before you.

  The assumptions regarding signalling, scanning and tracking we summarize as **RP-5***.

Question: Find or prove impossible a randomized algorithm which a.a.s. achieves rendezvous in linear time, assuming only RP:1-4, 5*, 6 and some assumption on the density of a configuration:

Question: Find or prove impossible a randomized algorithm which a.a.s. achieves rendezvous in linear time, assuming only RP:1-4, 5*, 6 and some assumption on the density of a configuration:
- the density should be high enough so that the graph diameter is a.a.s. of the same order of magnitude as the Euclidean diameter

Question: Find or prove impossible a randomized algorithm which a.a.s. achieves rendezvous in linear time, assuming only RP:1-4, 5*, 6 and some assumption on the density of a configuration:
- the density should be high enough so that the graph diameter is a.a.s. of the same order of magnitude as the Euclidean diameter
- it should not be too high: the previous example suggests at least subexponential in the diameter.

Question: Find or prove impossible a randomized algorithm which a.a.s. achieves rendezvous in linear time, assuming only RP:1-4, 5*, 6 and some assumption on the density of a configuration:
- the density should be high enough so that the graph diameter is a.a.s. of the same order of magnitude as the Euclidean diameter
- it should not be too high: the previous example suggests at least subexponential in the diameter.

Result: *Assume in addition that each agent possesses unlimited memory* **(RP-7)**. *Then there is a randomized algorithm* $\mathcal{A}$ *such that the following holds: There are absolute constants* C_1, C_2 *such that, if* $f : \mathbb{N} \to \mathbb{N}$ *is a function satisfying* $C_1 r^2 \log r < f(r) = o(r^3)$ *and* $f(r)$ *points are placed uniformly and independently at random in the interior of a closed disc* $\mathcal{D} = \mathcal{D}_r$ *of radius* r *in* $\mathbb{R}^2$ *and proceed to execute the algorithm* $\mathcal{A}$, *they will a.a.s. rendezvous in time at most* $C_2 r$.

The assumptions on the function f guarantee that the visibility graph G has some nice properties a.a.s.:

The assumptions on the function f guarantee that the visibility graph G has some nice properties a.a.s.:

"Lemma 2": The graph diameter is $O(r)$ and every vertex has $\Theta\left(\frac{f(r)}{r^2}\right)$ neighbours.

The assumptions on the function f guarantee that the visibility graph G has some nice properties a.a.s.:

"Lemma 2": The graph diameter is $O(r)$ and every vertex has $\Theta\left(\frac{f(r)}{r^2}\right)$ neighbours.

"Lemma 3": Define the **graph boundary** δG to consist of those vertices not surrounded by a polygon of neighbours. Then
(i) δG contains $\Theta(r)$ vertices
(ii) every point of δG is very close to the Euclidean boundary.

The assumptions on the function f guarantee that the visibility graph G has some nice properties a.a.s.:

"Lemma 2": The graph diameter is $O(r)$ and every vertex has $\Theta\left(\frac{f(r)}{r^2}\right)$ neighbours.

"Lemma 3": Define the **graph boundary** δG to consist of those vertices not surrounded by a polygon of neighbours. Then
(i) δG contains $\Theta(r)$ vertices
(ii) every point of δG is very close to the Euclidean boundary.

- Lemma 2 is easy and requires only the lower bound on f.

The assumptions on the function f guarantee that the visibility graph G has some nice properties a.a.s.:

"Lemma 2": The graph diameter is $O(r)$ and every vertex has $\Theta\left(\frac{f(r)}{r^2}\right)$ neighbours.

"Lemma 3": Define the **graph boundary** δG to consist of those vertices not surrounded by a polygon of neighbours. Then
(i) δG contains $\Theta(r)$ vertices
(ii) every point of δG is very close to the Euclidean boundary.

- Lemma 2 is easy and requires only the lower bound on f.
- Lemma 3 is harder though the proof is a "standard" second moment analysis. The upper bound on f is needed for part (i).

ALGORITHM:

ALGORITHM:

Step 1: Choose a leader. Lemma 3 is important for this step.

ALGORITHM:

Step 1: Choose a leader. Lemma 3 is important for this step.

- Each agent **scans** its neighbourhood in an attempt to rule out its being on δG.

ALGORITHM:

Step 1: Choose a leader. Lemma 3 is important for this step.

- Each agent **scans** its neighbourhood in an attempt to rule out its being on δG.
- The surviving agents each generate a sequence of random bits, one for each of its neighbours.

ALGORITHM:

Step 1: Choose a leader. Lemma 3 is important for this step.

- Each agent **scans** its neighbourhood in an attempt to rule out its being on δG.
- The surviving agents each generate a sequence of random bits, one for each of its neighbours.
- Each agents stores his number with a chosen accomplice, then performs a walk around the (approximate) Euclidean boundary, comparing his number with all others. To execute this step requires exchange of various **colour signals**.

ALGORITHM:

Step 1: Choose a leader. Lemma 3 is important for this step.

- Each agent **scans** its neighbourhood in an attempt to rule out its being on δG.
- The surviving agents each generate a sequence of random bits, one for each of its neighbours.
- Each agents stores his number with a chosen accomplice, then performs a walk around the (approximate) Euclidean boundary, comparing his number with all others. To execute this step requires exchange of various **colour signals**.
- With high probability, a unique agent returns to his own number in time $O(r)$ without having encountered any larger one **(Lemma 1)**. He signals he is the leader.

ALGORITHM:

Step 1: Choose a leader. Lemma 3 is important for this step.

- Each agent **scans** its neighbourhood in an attempt to rule out its being on δG.
- The surviving agents each generate a sequence of random bits, one for each of its neighbours.
- Each agents stores his number with a chosen accomplice, then performs a walk around the (approximate) Euclidean boundary, comparing his number with all others. To execute this step requires exchange of various **colour signals**.
- With high probability, a unique agent returns to his own number in time $O(r)$ without having encountered any larger one **(Lemma 1)**. He signals he is the leader.

Step 2: Everyone moves to the leader's location.

ALGORITHM:

Step 1: Choose a leader. Lemma 3 is important for this step.

- Each agent **scans** its neighbourhood in an attempt to rule out its being on δG.
- The surviving agents each generate a sequence of random bits, one for each of its neighbours.
- Each agents stores his number with a chosen accomplice, then performs a walk around the (approximate) Euclidean boundary, comparing his number with all others. To execute this step requires exchange of various **colour signals**.
- With high probability, a unique agent returns to his own number in time $O(r)$ without having encountered any larger one **(Lemma 1)**. He signals he is the leader.

Step 2: Everyone moves to the leader's location.

- Employs **tracking** mechanism and Lemma 2 is important here.

Our algorithm has various shortcomings which any further work should address:

Our algorithm has various shortcomings which any further work should address:

- We ignore the issue of storage capacity.

Our algorithm has various shortcomings which any further work should address:

- We ignore the issue of storage capacity.
- The upper bound on the density is polynomial in the diameter, hence far from the original hope that subexponential might do. Can this be improved ?

Our algorithm has various shortcomings which any further work should address:

- We ignore the issue of storage capacity.
- The upper bound on the density is polynomial in the diameter, hence far from the original hope that subexponential might do. Can this be improved ?
- The algorithm should in fact be robust if we modify the **shape** of the region, however we do require that there is “some nice” shape a priori.

Our algorithm has various shortcomings which any further work should address:

- We ignore the issue of storage capacity.
- The upper bound on the density is polynomial in the diameter, hence far from the original hope that subexponential might do. Can this be improved ?
- The algorithm should in fact be robust if we modify the **shape** of the region, however we do require that there is "some nice" shape a priori.
- Finally, the decision-making and movement aspects of rendezvous are not decoupled. Agents are required to be mobile in Step 1 (choosing the leader).

Then there are fundamental issues of “practicality”:

Then there are fundamental issues of “practicality”:

- Is it “practical” to assume RP-5* can be implemented and in a way which doesn't cause significant time delay ? We feel this issue is beyond our domain of competence.

Then there are fundamental issues of "practicality":

- Is it "practical" to assume RP-5* can be implemented and in a way which doesn't cause significant time delay ? We feel this issue is beyond our domain of competence.
- Our whole analysis is
  (i) "asymptotic" (the diameter is imagined tending to infinity)
  (ii) "probabilistic" - one might only be interested in failsafe, deterministic procedures.