Distributed robotic networks:
models, tasks, and complexity

Jorge Cortés and Sonia Martínez

Mechanical and Aerospace Engineering
University of California, San Diego
{cortes,soniamd}@ucsd.edu

28th Benelux Meeting on Systems and Control
Spa, Belgium, March 17, 2009

Acknowledgments: Francesco Bullo
Emilio Frazzoli
Cooperative multi-agent systems

What kind of systems?
Groups of agents with control, sensing, communication and computing

Each individual
- **senses** its immediate environment
- **communicates** with others
- **processes** information gathered
- **takes local action** in response
Self-organized behaviors in biological groups
Decision making in animals

Able to
- deploy over a given region
- assume specified pattern
- rendezvous at a common point
- jointly initiate motion/change direction in a synchronized way

Species achieve synchronized behavior
- with limited sensing/communication between individuals
- without apparently following group leader

(Couzin et al, Nature 05; Conradt et al, Nature 03)
Engineered multi-agent systems

Embedded robotic systems and sensor networks for

- high-stress, rapid deployment — e.g., disaster recovery networks
- distributed environmental monitoring — e.g., portable chemical and biological sensor arrays detecting toxic pollutants
- autonomous sampling for biological applications — e.g., monitoring of species in risk, validation of climate and oceanographic models
- science imaging — e.g., multисpacecraft distributed interferometers flying in formation to enable imaging at microarcsecond resolution
Research challenges

What useful engineering tasks can be performed with limited-sensing/communication agents?

Feedback rather than open-loop computation for known/static setup

Information flow who knows what, when, why, how, dynamically changing

Reliability/performance robust, efficient, predictable behavior

How to coordinate individual agents into coherent whole?

Objective: systematic methodologies to design and analyze cooperative strategies to control multi-agent systems

Integration of control, communication, sensing, computing
Research program: what are we after?

**Design** of provably correct coordination algorithms for basic tasks

**Formal model** to rigorously formalize, analyze, and compare coordination algorithms

**Mathematical tools** to study convergence, stability, and robustness of coordination algorithms

**Coordination tasks**

- exploration, map building, search and rescue,
- surveillance, odor localization, monitoring, distributed sensing
## Technical approach

### Optimization Methods
- resource allocation
- geometric optimization
- load balancing

### Geometry & Analysis
- computational structures
- differential geometry
- nonsmooth analysis

### Control & Robotics
- algorithm design
- cooperative control
- stability theory

### Distributed Algorithms
- adhoc networks
- decentralized vs centralized
- emerging behaviors

---

![Robot Network Images](image-url)
Outline

1. Synchronous networks
   - A motivating problem: leader election
   - Distributed algorithms
   - Complexity notions

2. Robotic networks
   - A motivating problem: direction agreement and equidistance
   - Proximity graphs
   - Control and communication laws
   - Coordination tasks and complexity notions

3. Complexity analysis of agree and pursue law

4. Conclusions
Leader election on a ring of processors

Network size is unknown to agents

Problem (Leader election)

Assume all processors have a state variable, say leader, initially set to unknown

A leader is elected when one and only one processor has the state variable set to true and all others have it set to false

Objective: elect a leader
The Le Lann-Chang-Roberts (LCR) algorithm

To solve the leader election problem, each agent

- sets max UID received so far to its own UID
- initially transmits its UID to neighbors
- at each communication round: listens to messages from other agents, and compares the received UIDs with its UID
  - if max UID received is larger than own UID, declares itself a non-leader, resets max UID, and transmits it in the next communication round
  - if max UID received is smaller than own UID, does nothing
  - if max UID is equal to own UID, declares itself a leader

![Diagram of the LCR algorithm](image)
The LCR algorithm solves the leader election problem on a ring digraph
- only agent with largest UID declares itself a leader
- all other agents declare themselves as non-leaders

Ways to improve the algorithm?
- so that remaining agents know leader has been elected?
- so that leader election problem is solved differently/faster?
- so that fewer messages are transmitted?

If somebody comes up with a different algorithm, how can we tell which algorithm is better?
Synchronous networks

Distributed algorithms and parallel computing study algorithms that can be implemented in static networks of parallel processors.

**Synchronous network** is a group of processors with the ability to exchange messages and perform local computations. Mathematically, a digraph \((I, E)\),

1. \(I = \{1, \ldots, n\}\) is the set of unique identifiers (UIDs), and
2. \(E\) is a set of directed edges over the vertices \(\{1, \ldots, n\}\), called the communication links.
Outline

1. Synchronous networks
   - A motivating problem: leader election
   - Distributed algorithms
   - Complexity notions

2. Robotic networks
   - A motivating problem: direction agreement and equidistance
   - Proximity graphs
   - Control and communication laws
   - Coordination tasks and complexity notions

3. Complexity analysis of agree and pursue law

4. Conclusions
Distributed algorithm $\mathcal{DA}$ for a network $\mathcal{S}$ consists of the sets

1. $L$, a set containing the null element, called the communication alphabet; elements of $L$ are called messages;
2. $W[i]$, $i \in I$, called the processor state sets;
3. $W_0[i] \subseteq W[i]$, $i \in I$, sets of allowable initial values;

and of the maps

1. $\text{msg}^{[i]} : W[i] \times I \to L$, $i \in I$, called message-generation functions;
2. $\text{stf}^{[i]} : W[i] \times L^n \to W[i]$, $i \in I$, called state-transition functions.

If $W[i] = W$, $\text{msg}^{[i]} = \text{msg}$, and $\text{stf}^{[i]} = \text{stf}$ for all $i \in I$, then $\mathcal{DA}$ is said to be uniform and is described by a tuple $(L, W, \{W_0[i]\}_{i \in I}, \text{msg}, \text{stf})$.
The LCR algorithm – formally

Network: Ring network
Alphabet: $L = I \cup \{\text{null}\}$
Processor State: $w = (u, \text{max-uid}, \text{leader}, \text{transmit})$, where

$u \in I$, \hspace{1cm} \text{init: } u[i] = i \text{ for all } i$

$\text{max-uid} \in I$, \hspace{1cm} \text{init: } \text{max-uid}[i] = i \text{ for all } i$

$\text{leader} \in \{\text{true}, \text{false}, \text{unknown}\}$, \hspace{1cm} \text{init: } \text{leader}[i] = \text{unknown} \text{ for all } i$

$\text{transmit} \in \{\text{true}, \text{false}\}$, \hspace{1cm} \text{init: } \text{transmit}[i] = \text{true} \text{ for all } i$

\begin{verbatim}
function msg(w, i)
  1: if transmit = true then
  2:     return max-uid
  3: else
  4:     return null
\end{verbatim}
The LCR algorithm – formally

function stf(w, y)
1: if (y contains only null messages) OR (largest identifier in y < u) then
2: new-uid := max-uid
3: new-leader := leader
4: new-transmit := false
5: if (largest identifier in y = u) then
6: new-uid := max-uid
7: new-leader := true
8: new-transmit := false
9: if (largest identifier in y > u) then
10: new-uid := largest identifier in y
11: new-leader := false
12: new-transmit := true
13: return (u, new-uid, new-leader, new-transmit)
Network evolution

Execution: discrete-time communication and computation

Formally, evolution of \((\mathcal{S}, \mathcal{DA})\) from initial conditions \(w^{[i]}_0 \in W^{[i]}_0, i \in I\), is the collection of trajectories \(w^{[i]}: \mathbb{T} \rightarrow W^{[i]}, i \in I\), satisfying

\[
w^{[i]}(\ell) = \text{stf}^{[i]}(w^{[i]}(\ell - 1), y^{[i]}(\ell))
\]

where \(w^{[i]}(-1) = w^{[i]}_0, i \in I\), and where the trajectory \(y^{[i]}: \mathbb{T} \rightarrow L^n\) (describing the messages received by processor \(i\)) has components \(y^{[i]}_j(\ell), j \in I\),

\[
y^{[i]}_j(\ell) = \begin{cases} 
\text{msg}^{[j]}(w^{[j]}(\ell - 1), i), & \text{if } (j, i) \in \mathcal{E}, \\
\text{null}, & \text{otherwise}.
\end{cases}
\]
Outline

1. Synchronous networks
   - A motivating problem: leader election
   - Distributed algorithms
   - Complexity notions

2. Robotic networks
   - A motivating problem: direction agreement and equidistance
   - Proximity graphs
   - Control and communication laws
   - Coordination tasks and complexity notions

3. Complexity analysis of agree and pursue law

4. Conclusions
Characterizing performance: complexity notions

How good is a distributed algorithm? How costly to execute?
Complexity notions characterize performance of distributed algorithms

Algorithm completion: an algorithm *terminates* when only null messages are transmitted and all processors states become constants.

Time complexity: $\text{TC}(\mathcal{DA}, S)$ is maximum number of rounds required by execution of $\mathcal{DA}$ on $S$ among all allowable initial states.

Space complexity: $\text{SC}(\mathcal{DA}, S)$ is maximum number of basic memory units required by a processor executing $\mathcal{DA}$ on $S$ among all processors and all allowable initial states.

Communication complexity: $\text{CC}(\mathcal{DA}, S)$ is maximum number of basic messages transmitted over the entire network during execution of $\mathcal{DA}$ among all allowable initial states until termination (basic memory unit, message contains $\log(n)$ bits).
Quantifying complexity

Asymptotic “order of magnitude” measures. E.g., algorithm has time complexity of order

1. $O(f(n))$ if, for all $n$, for all networks of order $n$ and for all initial processor values, $TC$ is lower than a constant factor times $f(n)$

2. $\Omega(f(n))$ if, for all $n$, $\exists$ network of order $n$ and initial processor values such that $TC$ is greater than a constant factor times $f(n)$

3. $\Theta(f(n))$ if $TC$ is of order $\Omega(f(n))$ and $O(f(n))$ at the same time

Similar conventions for space and communication complexity

Numerous variations of complexity definitions are possible

1. “Global” rather than “existential” lower bounds

2. Expected or average complexity notions

3. Complexity notions for problems, rather than for algorithms
Leader election by comparison

Le Lann-Chang-Roberts (LCR) algorithm solves leader election on a ring with complexities

1. time complexity is $n$
   
   *it takes $n$ communication rounds for UID “n” to travel back to agent n*

2. space complexity is 4
   
   *u, max-uid, leader, transmit*

3. communication complexity is $\Theta(n^2)$
   
   *Upper bound is straightforward. Initial condition that gives rise to lower bound?*
Beyond leader election

Plenty of problems within distributed algorithms and parallel processing

- Distributed breadth-first and depth-first tree construction, flooding, consensus, linear algebra, optimization
- Asynchronism, processor failures, communication failures

For **robotic networks**, spatial dimension introduces new problems

- mobility means changing interaction topology
- physical variables live in continuous spaces
- in addition to processing and communication, need to take control and sensing into account
- task and complexity notions have different meanings
Outline

1. Synchronous networks
   - A motivating problem: leader election
   - Distributed algorithms
   - Complexity notions

2. Robotic networks
   - A motivating problem: direction agreement and equidistance
   - Proximity graphs
   - Control and communication laws
   - Coordination tasks and complexity notions

3. Complexity analysis of agree and pursue law

4. Conclusions
Direction agreement and equidistance

Network size is unknown to agents

Problem (Direction agreement & equidistance)

Assume agents move in circle according to first-order integrator dynamics. Some move clockwise, others counterclockwise

Agents talk to other agents within distance $r$

**Objective:** agree on a common direction of motion and uniformly deploy over circle
The agree-and-pursue algorithm

To solve the direction agreement and equidistance problem, each agent

- sets max UID received so far to its own UID
- initially transmits its direction of motion and UID to neighbors

**at each communication round:** listens to messages from other agents and compares the received UIDs from agents moving toward its position with its own UID. If max UID is larger than own UID, resets UID and direction of motion

**between communication rounds:** moves $k_{\text{prop}} \in (0, 1/2)$ times the distance to the immediately next neighbor in chosen direction, or, if no neighbors, $k_{\text{prop}}$ times communication range $r$
The agree-and-pursue algorithm solves the direction agreement and equidistance problem on a circle

- all agents agree on a common direction of motion – either clockwise or counterclockwise
- network asymptotically achieves uniform, equally-spaced rotating configuration

New issues arise when considering robotic networks

- As agents move, interconnection topology changes (e.g., network might be disconnected, and then leader election would not work)
- Tasks might not be achieved exactly, but asymptotically (e.g., equidistance)
- Need to rethink model and notions of complexity to account for spatial component
Outline

1 Synchronous networks
   - A motivating problem: leader election
   - Distributed algorithms
   - Complexity notions

2 Robotic networks
   - A motivating problem: direction agreement and equidistance
   - Proximity graphs
   - Control and communication laws
   - Coordination tasks and complexity notions

3 Complexity analysis of agree and pursue law

4 Conclusions
Proximity graphs model interconnection topology

Proximity graph

A graph whose vertex set is a set of distinct points and whose edge set is a function of the relative locations of the point set

Appear in computational geometry and topology control of wireless networks

**Definition (Proximity graph)**

Let $X$ be a $d$-dimensional space chosen among $\mathbb{R}^d$, $S^d$, and $\mathbb{R}^{d_1} \times S^{d_2}$, with $d_1 + d_2 = d$. Let $\mathcal{G}(X)$ be the set of all undirected graphs whose vertex set is an element of $\mathcal{F}(X)$ (finite subsets of $X$)

A proximity graph $\mathcal{G} : \mathcal{F}(X) \to \mathcal{G}(X)$ associates to $\mathcal{P} = \{p_1, \ldots, p_n\} \subset X$ an undirected graph with vertex set $\mathcal{P}$ and edge set $\mathcal{E}_\mathcal{G}(\mathcal{P}) \subseteq \{(p, q) \in \mathcal{P} \times \mathcal{P} \mid p \neq q\}$. 
Examples of proximity graphs

On $(\mathbb{R}^d, \text{dist}_2)$, $(S^d, \text{dist}_g)$, or $(\mathbb{R}^{d_1} \times S^{d_2}, (\text{dist}_2, \text{dist}_g))$

1. the $r$-disk graph $G_{\text{disk}}(r)$, for $r \in \mathbb{R}_{>0}$, with $(p_i, p_j) \in E_{G_{\text{disk}}(r)}(\mathcal{P})$ if $\text{dist}(p_i, p_j) \leq r$

2. the Delaunay graph $G_{\text{D}}$, with $(p_i, p_j) \in E_{G_{\text{D}}}(\mathcal{P})$ if $V_i(\mathcal{P}) \cap V_j(\mathcal{P}) \neq \emptyset$

3. the $r$-limited Delaunay graph $G_{\text{LD}}(r)$, for $r \in \mathbb{R}_{>0}$, with $(p_i, p_j) \in E_{G_{\text{LD}}(r)}(\mathcal{P})$ if $V_{i, \frac{r}{2}}(\mathcal{P}) \cap V_{j, \frac{r}{2}}(\mathcal{P}) \neq \emptyset$

4. given a simple polygon $Q$ in $\mathbb{R}^2$, the visibility graph $G_{\text{vis}, Q}$, with $(p_i, p_j) \in E_{G_{\text{vis}, Q}}(\mathcal{P})$ if the closed segment $[p_i, p_j]$ from $p_i$ to $p_j$ is contained in $Q$
Set of neighbors map

For proximity graph $\mathcal{G}$, $p \in X$, and $\mathcal{P} = \{p_1, \ldots, p_n\} \in \mathbb{F}(X)$ associate set of neighbors map $\mathcal{N}_{\mathcal{G},p}: \mathbb{F}(X) \rightarrow \mathbb{F}(X)$

$$\mathcal{N}_{\mathcal{G},p}(\mathcal{P}) = \{q \in \mathcal{P} \mid (p, q) \in \mathcal{E}_\mathcal{G}(\mathcal{P} \cup \{p\})\}$$

Typically, $p$ is a point in $\mathcal{P}$, but this works for any $p \in X$

When does a proximity graph provide sufficient information to compute another proximity graph?
Spatially distributed graphs

E.g., if a node knows position of its neighbors in the complete graph, then it can compute its neighbors with respect to any proximity graph.

Formally, given $G_1$ and $G_2$,

$G_1$ is spatially distributed over $G_2$ if, for all $p \in \mathcal{P}$,

$$\mathcal{N}_{G_1,p}(\mathcal{P}) = \mathcal{N}_{G_1,p}(\mathcal{N}_{G_2,p}(\mathcal{P})),$$

that is, any node equipped with the location of its neighbors with respect to $G_2$ can compute its set of neighbors with respect to $G_1$.

$G_1$ spatially distributed over $G_2 \implies G_1 \subset G_2$

Converse not true: $G_D \cap G_{\text{disk}}(r) \subset G_{\text{disk}}$, but $G_D \cap G_{\text{disk}}(r)$ not spatially distributed over $G_{\text{disk}}(r)$. 
Spatially distributed maps

Given a set $Y$ and a proximity graph $\mathcal{G}$, a map $T : X^n \rightarrow Y^n$ is spatially distributed over $\mathcal{G}$ if $\exists$ a map $\tilde{T} : X \times \mathbb{F}(X) \rightarrow Y$ such that for all $(p_1, \ldots, p_n) \in X^n$ and for all $j \in \{1, \ldots, n\}$,

$$T_j(p_1, \ldots, p_n) = \tilde{T}(p_j, N_{\mathcal{G}, p_j}(p_1, \ldots, p_n)),$$

where $T_j$ denotes the $j$th-component of $T$.

Equivalently,

the $j$th component of a spatially distributed map at $(p_1, \ldots, p_n)$ can be computed with the knowledge of the vertex $p_j$ and the neighboring vertices in the undirected graph $\mathcal{G}(P)$. 
Physical components of a robotic network

Group of robots with the ability to exchange messages, perform local computations, and control motion

**Mobile robot:** continuous-time continuous-space dynamical system,

1. $X$ is $d$-dimensional space chosen among $\mathbb{R}^d$, $\mathbb{S}^d$, and the Cartesian products $\mathbb{R}^{d_1} \times \mathbb{S}^{d_2}$, for some $d_1 + d_2 = d$, called the state space;

2. $U$ is a compact subset of $\mathbb{R}^m$ containing $0_n$, called the input space;

3. $X_0$ is a subset of $X$, called the set of allowable initial states;

4. $f: X \times U \to \mathbb{R}^d$ is a smooth control vector field on $X$
Definition (Robotic network)

The physical components of a uniform robotic network $\mathcal{S}$ consist of a tuple $(I, \mathcal{R}, \mathcal{E})$, where

1. $I = \{1, \ldots, n\}$; $I$ is called the set of unique identifiers (UIDs);
2. $\mathcal{R} = \{R[i]\}_{i \in I} = \{(X, U, X_0, f)\}_{i \in I}$ is a set of mobile robots;
3. $\mathcal{E}$ is a map from $X^n$ to the subsets of $I \times I$; this map is called the communication edge map.

Map $x \mapsto (I, \mathcal{E}(x))$ models topology of the communication service among robots – proximity graph induced by network capabilities.
A couple of examples

**Locally-connected first-order robots in** $\mathbb{R}^d$: $S_{\text{disk}}$

$n$ points $x^{[1]}, \ldots, x^{[n]}$ in $\mathbb{R}^d$, $d \geq 1$, obeying $\dot{x}^{[i]}(t) = u^{[i]}(t)$, with $u^{[i]} \in [-u_{\text{max}}, u_{\text{max}}]$. These are identical robots of the form

$$(\mathbb{R}^d, [-u_{\text{max}}, u_{\text{max}}]^d, \mathbb{R}^d, (0, e_1, \ldots, e_d))$$

Each robot can communicate to other robots within $r$, $G_{\text{disk}}(r)$ on $\mathbb{R}^d$

**Locally-connected first-order robots in** $S^1$: $S_{\text{circle,disk}}$

$n$ robots $\theta^{[1]}, \ldots, \theta^{[n]}$ in $S^1$, moving along on the unit circle with angular velocity equal to the control input. Each robot is described by

$$(S^1, [-u_{\text{max}}, u_{\text{max}}], S^1, (0, e))$$

($e$ describes unit-speed counterclockwise rotation). Each robot can communicate to other robots within $r$ along the circle, $G_{\text{disk}}(r)$ on $S^1$
Model assumes ability of each robot to know its own absolute position

**Alternative setting:** robots do not communicate amongst themselves, but instead

- detect and measure each other’s relative position through appropriate sensors
- perform measurements of the environment without having a priori knowledge

Robots do not have the ability to perform measurements expressed in a common reference frame
Outline

1 Synchronous networks
   - A motivating problem: leader election
   - Distributed algorithms
   - Complexity notions

2 Robotic networks
   - A motivating problem: direction agreement and equidistance
   - Proximity graphs
   - Control and communication laws
   - Coordination tasks and complexity notions

3 Complexity analysis of agree and pursue law

4 Conclusions
Uniform control and communication law

1. **communication schedule**
   \[ \mathbb{T} = \{ t_\ell \}_{\ell \in \mathbb{N}_0} \subset \mathbb{R}_{\geq 0} \]

2. **communication alphabet**
   \[ L \text{ including the null message} \]

3. **processor state space**
   \[ W, \text{ with initial allowable } W_0^{[i]} \]

4. **message-generation function**
   \[ \text{msg}: \mathbb{T} \times X \times W \times I \to L \]

5. **state-transition functions**
   \[ \text{stf}: X \times W \times L^n \to W \]

6. **control function**
   \[ \text{ctrl}: \mathbb{T} \times X \times W \times L^n \to U \]

**Execution:**
- discrete-time communication
- discrete-time computation
- continuous-time motion

Cortés & Martínez (UCSD)  Distributed robotic networks  March 17, 2009  39 / 64
The agree-and-pursue algorithm – formally

Alphabet: \( L = S^1 \times \{c, cc\} \times I \cup \{\text{null}\} \)

Processor State: \( w = (\text{dir}, \text{max-uid}) \), where
\[
\begin{align*}
\text{dir} &\in \{c, cc\}, \quad \text{initially: } \text{dir}^{[i]} \text{ unspecified} \\
\text{max-uid} &\in I, \quad \text{initially: } \text{max-uid}^{[i]} = i \text{ for all } i
\end{align*}
\]

function \( \text{msg}(\theta, w, i) \)
\[
1: \text{return } (\theta, w)
\]

function \( \text{stf}(w, y) \)
\[
1: \text{for each non-null message } (\theta_{\text{rcvd}}, (\text{dir}_{\text{rcvd}}, \text{max-uid}_{\text{rcvd}})) \text{ do} \\
2: \quad \text{if } (\text{max-uid}_{\text{rcvd}} > \text{max-uid}) \text{ AND } (\text{dist}_{cc}(\theta, \theta_{\text{rcvd}}) \leq r \text{ AND } \text{dir}_{\text{rcvd}} = c) \text{ OR } \\
\hspace{1cm} (\text{dist}_{c}(\theta, \theta_{\text{rcvd}}) \leq r \text{ AND } \text{dir}_{\text{rcvd}} = cc) \text{ then} \\
3: \quad \text{new-dir} := \text{dir}_{\text{rcvd}} \\
4: \quad \text{new-uid} := \text{max-uid}_{\text{rcvd}} \\
5: \text{return } (\text{new-dir}, \text{new-uid})
\]

function \( \text{ctrl}(\theta_{\text{smpld}}, w, y) \)
\[
1: \quad \text{d}_{\text{tmp}} := r \\
2: \text{for each non-null message } (\theta_{\text{rcvd}}, (\text{dir}_{\text{rcvd}}, \text{max-uid}_{\text{rcvd}})) \text{ do} \\
3: \quad \text{if } (\text{dir} = cc) \text{ AND } (\text{dist}_{cc}(\theta_{\text{smpld}}, \theta_{\text{rcvd}}) < d_{\text{tmp}}) \text{ then} \\
4: \quad \text{d}_{\text{tmp}} := \text{dist}_{cc}(\theta_{\text{smpld}}, \theta_{\text{rcvd}}) \text{ and } \text{u}_{\text{tmp}} := k_{\text{prop}} d_{\text{tmp}} \text{ (} k_{\text{prop}} \in (0, \frac{1}{2}) \text{)} \\
5: \quad \text{if } (\text{dir} = c) \text{ AND } (\text{dist}_{c}(\theta_{\text{smpld}}, \theta_{\text{rcvd}}) < d_{\text{tmp}}) \text{ then} \\
6: \quad \text{d}_{\text{tmp}} := \text{dist}_{c}(\theta_{\text{smpld}}, \theta_{\text{rcvd}}) \text{ and } \text{u}_{\text{tmp}} := -k_{\text{prop}} d_{\text{tmp}} \\
7: \text{return } \text{u}_{\text{tmp}}
Outline

1. Synchronous networks
   - A motivating problem: leader election
   - Distributed algorithms
   - Complexity notions

2. Robotic networks
   - A motivating problem: direction agreement and equidistance
   - Proximity graphs
   - Control and communication laws
   - Coordination tasks and complexity notions

3. Complexity analysis of agree and pursue law

4. Conclusions
Coordination tasks

What is a coordination task for a robotic network? When does a control and communication law achieve a task? And with what time, space, and communication complexity?

A coordination task for a robotic network $S$ is a map $T: X^n \times W^n \rightarrow \{true, false\}$

- Logic-based: agree, synchronize, form a team, elect a leader
- Motion: deploy, gather, flock, reach pattern
- Sensor-based: search, estimate, identify, track, map

A control and communication law $CC$ achieves the task $T$ if, for all initial conditions $x_0^{[i]} \in X_0^{[i]}$ and $w_0^{[i]} \in W_0^{[i]}$, $i \in I$, the network evolution $t \mapsto (x(t), w(t))$ has the property

\[
\text{there exists } T \in \mathbb{R}_{>0} \text{ such that } T(x(t), w(t)) = true \text{ for } t \geq T
\]
Loosely speaking, achieving a task means obtaining and maintaining a specified pattern in the robot physical or processor state. In other words, the task is achieved if at some time and for all subsequent times the predicate evaluates to true along system trajectories.

More general tasks based on more expressive predicates on trajectories can be defined through temporal and propositional logic, e.g.,

periodically visiting a desired set of configurations.
Direction agreement task $T_{\text{dir}}: (S^1)^n \times W^n \rightarrow \{\text{true, false}\}$

$$T_{\text{dir}}(\theta, w) = \begin{cases} 
\text{true}, & \text{if } \text{dir}[1] = \cdots = \text{dir}[n] \\
\text{false}, & \text{otherwise}
\end{cases}$$

For $\epsilon > 0$, equidistance task $T_{\epsilon-\text{eqdstnc}}: (S^1)^n \rightarrow \{\text{true, false}\}$ is true iff

$$\left| \min_{j \neq i} \text{dist}_c(\theta[i], \theta[j]) \right| - \left| \min_{j \neq i} \text{dist}_{cc}(\theta[i], \theta[j]) \right| < \epsilon, \quad \text{for all } i \in I$$
For network $S$, task $T$, and algorithm $CC$, define costs/complexity
control effort, communication packets, computational cost

**Time complexity:** maximum number of communication rounds required to
achieve $T$

**Space complexity:** maximum number of basic memory units required by a
robot processor among all robots

**Communication complexity:** maximum number of basic messages transmitted
over entire network

(among all allowable initial physical and
processor states until termination)

basic memory unit/message contain $\log(n)$ bits
The **time complexity to achieve** $\mathcal{T}$ **with** $CC$ **from**

$$(x_0, w_0) \in \prod_{i \in I} X^i_0 \times \prod_{i \in I} W^i_0$$

**is**

$$TC(\mathcal{T}, CC, x_0, w_0) = \inf \{ \ell \mid \mathcal{T}(x(t_k), w(t_k)) = \text{true}, \text{ for all } k \geq \ell \},$$

where $t \mapsto (x(t), w(t))$ is the evolution of $(S, CC)$ from the initial condition $(x_0, w_0)$

The **time complexity to achieve** $\mathcal{T}$ **with** $CC$ **is**

$$TC(\mathcal{T}, CC) = \sup \left\{ TC(\mathcal{T}, CC, x_0, w_0) \mid (x_0, w_0) \in \prod_{i \in I} X^i_0 \times \prod_{i \in I} W^i_0 \right\}.$$

The **time complexity of** $\mathcal{T}$ **is**

$$TC(\mathcal{T}) = \inf \{ TC(\mathcal{T}, CC) \mid CC \text{ compatible with } \mathcal{T} \}$$
The set of all non-null messages generated during one communication round from network state \((x, w)\)

\[
\mathcal{M}(x, w) = \left\{ (i, j) \in \mathcal{E}(x) \mid \text{msg}^i(x[i], w[i], j) \neq \text{null} \right\}.
\]

The mean communication complexity and the total communication complexity to achieve \(T\) with \(CC\) from \((x_0, w_0) \in \prod_{i \in I} X^{[i]}_0 \times \prod_{i \in I} W^{[i]}_0\) are,

\[
\begin{align*}
\text{MCC}(T, CC, x_0, w_0) &= \frac{|L|_{\text{basic}}}{\lambda} \sum_{\ell=0}^{\lambda-1} |\mathcal{M}(x(\ell), w(\ell))|, \\
\text{TCC}(T, CC, x_0, w_0) &= |L|_{\text{basic}} \sum_{\ell=0}^{\lambda-1} |\mathcal{M}(x(\ell), w(\ell))|,
\end{align*}
\]

where \(|L|_{\text{basic}}\) is number of basic messages required to represent elements of \(L\) and \(\lambda = \text{TC}(CC, T, x_0, w_0)\).
Asymptotic results

Complexities in $O(f(n))$, $\Omega(f(n))$, or $\Theta(f(n))$ as $n \to \infty$.

- **Infinite-horizon mean communication complexity**: mean communication complexity to maintain true the task for all times

\[
cc(\mathcal{CC}, x_0, w_0) = \lim_{\lambda \to +\infty} \left\lfloor \frac{|L|_{\text{basic}}}{\lambda} \sum_{\ell=0}^{\lambda} |M(x(\ell), w(\ell))| \right\rfloor
\]

- **Communication complexity in omnidirectional networks**: All neighbors of agent receive the signal it transmits. Makes sense to count the number of transmissions, i.e., a unit cost per node, rather than a unit cost per edge of the network.

- **Energy complexity**

- **Expected**, rather than worst-case notions
1 Synchronous networks
   • A motivating problem: leader election
   • Distributed algorithms
   • Complexity notions

2 Robotic networks
   • A motivating problem: direction agreement and equidistance
   • Proximity graphs
   • Control and communication laws
   • Coordination tasks and complexity notions

3 Complexity analysis of agree and pursue law

4 Conclusions
Let \( r : \mathbb{N} \rightarrow ]0, 2\pi[ \) be a monotone non-increasing function of number of agents \( n \) – modeling wireless communication congestion.

**Theorem**

*In the limit as \( n \to +\infty \) and \( \epsilon \to 0^+ \), the network \( S_{\text{circle,disk}} \), the law \( CC_{\text{agree \\& pursue}} \), and the tasks \( T_{\text{dir}} \) and \( T_{\epsilon\text{-eqdstnc}} \) together satisfy:*

1. \( TC(T_{\text{dir}}, CC_{\text{agree \\& pursue}}) \in \Theta(r(n)^{-1}) \);
2. If \( \delta(n) = nr(n) - 2\pi \) is lower bounded by a positive constant as \( n \to +\infty \),

\[
TC(T_{\epsilon\text{-eqdstnc}}, CC_{\text{agree \\& pursue}}) \in \Omega(n^2 \log(n\epsilon)^{-1}), \\
TC(T_{\epsilon\text{-eqdstnc}}, CC_{\text{agree \\& pursue}}) \in O(n^2 \log(n\epsilon^{-1})).
\]

*If \( \delta(n) \) is lower bounded by a negative constant, then \( CC_{\text{agree \\& pursue}} \) does not achieve \( T_{\epsilon\text{-eqdstnc}} \) in general.*
Proof sketch - $O$ bound for $\mathcal{T}_{\text{dir}}$

**Claim:** $TC(\mathcal{T}_{\text{dir}}, \mathcal{CC}_{\text{AGREE} \ & \ \text{Pursue}}) \leq \frac{2\pi}{(k_{\text{prop}} r(n))}$

By contradiction, assume there exists initial condition such that execution has
time complexity larger than $\frac{2\pi}{(k_{\text{prop}} r(n))}$
Without loss of generality, $\mathbf{dir}^{[n]}(0) = c$. For $\ell \leq \frac{2\pi}{(k_{\text{prop}} r(n))}$, let

$$k(\ell) = \arg\min \{ \text{dist}_{\text{cc}}(\theta^{[n]}(0), \theta^{[i]}(\ell)) \mid \mathbf{dir}^{[i]}(\ell) = \text{cc}, i \in I \}$$

Agent $k(\ell)$ is agent moving counterclockwise that has smallest
counterclockwise distance from the initial position of agent $n$

Recall that according to $\mathcal{CC}_{\text{AGREE} \ & \ \text{Pursue}}$

- messages with $\mathbf{dir} = \text{cc}$ can only travel counterclockwise
- messages with $\mathbf{dir} = c$ can only travel clockwise

Therefore, position of agent $k(\ell)$ at time $\ell$ can only belong to the
counterclockwise interval from the position of agent $k(0)$ at time 0 to the
position of agent $n$ at time 0
Proof sketch - O bound for $\tau_{\text{dir}}$

How fast the message from agent $n$ travels clockwise?

For $\ell \leq 2\pi/(k_{\text{prop}}r(n))$, define

$$j(\ell) = \arg\max\{\text{dist}_c(\theta[n](0), \theta[i](\ell)) \mid \text{prior}^{[i]}(\ell) = n, i \in I\}$$

Agent $j(\ell)$

- has prior equal to $n$
- is moving clockwise

and is the agent furthest from the initial position of agent $n$ in the clockwise direction with these two properties

Initially, $j(0) = n$. Additionally, for $\ell \leq 2\pi/(k_{\text{prop}}r(n))$, we claim

$$\text{dist}_c(\theta[j(\ell)](\ell), \theta[j(\ell+1)](\ell + 1)) \geq k_{\text{prop}}r(n)$$  

(1)
Proof sketch - O bound for $T_{\text{dir}}$

$TC(T_{\text{dir}}, CC_{\text{AGREE & PURSUE}}) \leq 2\pi/(k_{\text{prop}}r(n))$

Claim (1) happens because either (1) there is no agent clockwise-ahead of $\theta[j(\ell)](\ell)$ within clockwise distance $r$ and, therefore, the claim is obvious, or (2) there are such agents. In case (2), let $m$ denote the agent whose clockwise distance to agent $j(\ell)$ is maximal within the set of agents with clockwise distance $r$ from $\theta[j(\ell)](\ell)$. Then,

$$\text{dist}_c(\theta[j(\ell)](\ell), \theta[j(\ell+1)](\ell+1))$$

$$= \text{dist}_c(\theta[j(\ell)](\ell), \theta[m](\ell+1))$$

$$= \text{dist}_c(\theta[j(\ell)](\ell), \theta[m](\ell)) + \text{dist}_c(\theta[m](\ell), \theta[m](\ell+1))$$

$$\geq \text{dist}_c(\theta[j(\ell)](\ell), \theta[m](\ell)) + k_{\text{prop}}(r - \text{dist}_c(\theta[j(\ell)](\ell), \theta[m](\ell)))$$

$$= k_{\text{prop}}r + (1 - k_{\text{prop}})\text{dist}_c(\theta[j(\ell)](\ell), \theta[m](\ell)) \geq k_{\text{prop}}r$$

Therefore, after $2\pi/(k_{\text{prop}}r(n))$ communication rounds, the message with prior = $n$ has traveled the whole circle in the clockwise direction, and must therefore have reached agent $k(\ell)$

Contradiction
Proof sketch - O bound for $T_{\varepsilon}$-equidstnc

Assume $T_{\text{dir}}$ has been achieved and all agents are moving clockwise.

At time $\ell \in \mathbb{N}_0$, let $H(\ell)$ be the union of all the empty “circular segments” of length at least $r$,

$$H(\ell) = \{ x \in S^1 | \min_{i \in I} \text{dist}_c(x, \theta[i](\ell)) + \min_{j \in I} \text{dist}_{cc}(x, \theta[j](\ell)) > r \}.$$ 

$H(\ell)$ does not contain any point between two agents separated by a distance less than $r$, and each connected component has length at least $r$.

Let $n_H(\ell)$ be number of connected components of $H(\ell)$,

- if $H(\ell)$ is empty, then $n_H(\ell) = 0$
- $n_H(\ell) \leq n$
- if $n_H(\ell) > 0$, then $t \mapsto n_H(\ell + t)$ is non-increasing
Proof sketch - O bound for $T_{\epsilon}$-eqdstnc
Number of connected components is strictly decreasing

Claim: if $n_H(\ell) > 0$, then $\exists t > \ell$ such that $n_H(t) < n_H(\ell)$

By contradiction, assume $n_H(\ell) = n_H(t)$ for all $t > \ell$. Without loss of generality, let $\{1, \ldots, m\}$ be a set of agents with the properties

- $\text{dist}_{cc}(\theta[i](\ell), \theta[i+1](\ell)) \leq r$, for $i \in \{1, \ldots, m\}$
- $\theta[1](\ell)$ and $\theta[m](\ell)$ belong to the boundary of $H(\ell)$

One can show that, for $\tau \geq \ell$ and $i \in \{2, \ldots, m\}$

$$\theta[1](\tau + 1) = \theta[1](\tau) - k_{prop} r$$
$$\theta[i](\tau + 1) = \theta[i](\tau) - k_{prop} \text{dist}_c(\theta[i](\tau), \theta[i-1](\tau))$$
Tridiagonal and circulant linear dynamical systems

\[ \text{Trid}_n(a, b, c) = \begin{bmatrix} b & c & 0 & \cdots & 0 \\ a & b & c & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & a & b & c \\ 0 & \cdots & 0 & a & b \end{bmatrix}, \quad \text{Circ}_n(a, b, c) = \begin{bmatrix} b & c & 0 & \cdots & a \\ a & b & c & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & a & b & c \\ c & \cdots & 0 & a & b \end{bmatrix} \]

Linear dynamical systems

\[ y(\ell + 1) = Ay(\ell), \quad \ell \in \mathbb{N}_0 \]

Rates of convergence to set of equilibria can be characterized – carefully look at eigenvalues. Statements of the form

if \( a \geq 0, c \geq 0, b > 0, \text{ and } a + b + c = 1 \), then

\[ \lim_{\ell \to +\infty} y(\ell) = y_{\text{ave}}^1, \text{ where } \]

\[ y_{\text{ave}} = \frac{1}{n} \mathbf{1}^T y_0, \text{ and maximum time required (over all initial conditions } y_0 \in \mathbb{R}^n \text{) for } \|y(\ell) - y_{\text{ave}}^1\|_2 \leq \epsilon \|y_0 - y_{\text{ave}}^1\|_2 \text{ is } \Theta(n^2 \log \epsilon^{-1}) \]
Proof sketch- O bound for $T_{\varepsilon\text{-eqdstnc}}$

Contradiction argument

For $d(\tau) = (\text{dist}_{cc}(\theta[1](\tau), \theta[2](\tau)), \ldots, \text{dist}_{cc}(\theta[m-1](\tau), \theta[m](\tau)))$,

$$d(\tau + 1) = \text{Trid}_{m-1}(k_{\text{prop}}, 1 - k_{\text{prop}}, 0) d(\tau) + r[k_{\text{prop}}, 0, \ldots, 0]^T$$

Unique equilibrium point is $r(1, \ldots, 1)$. For $\eta_1 \in ]0, 1[$, $\tau \mapsto d(\tau)$ reaches ball of radius $\eta_1$ centered at equilibrium in $O(m \log m + \log \eta_1^{-1})$

This implies that $\tau \mapsto \sum_{i=1}^{m} d_i(\tau)$ is larger than $(m - 1)(r - \eta_1)$ in time $O(m \log m + \log \eta_1^{-1}) = O(n \log n + \log \eta_1^{-1})$. After this time,

$$2\pi \geq n_H(\ell)r + \sum_{j=1}^{n_H(\ell)} (r - \eta_1)(m_j - 1)$$

$$= n_H(\ell)r + (n - n_H(\ell))(r - \eta_1) = n_H(\ell)\eta_1 + n(r - \eta_1)$$
Proof sketch- O bound for $T_{\epsilon$-eqdstnc}

Take $\eta_1 = (nr - 2\pi)n^{-1} = \delta(n)n^{-1}$, and the contradiction follows from

$$2\pi \geq n_H(\ell)\eta_1 + nr - n\eta_1$$

$$= n_H(\ell)\eta_1 + nr + 2\pi - nr = n_H(\ell)\eta_1 + 2\pi$$

Therefore $n_H(\ell)$ decreases by one in time $O(n \log n)$

Iterating this argument $n$ times, in time $O(n^2 \log n)$ the set $H$ becomes empty. At that time, resulting network obeys

$$d(\tau + 1) = \text{Circ}_n(k_{\text{prop}}, 1 - k_{\text{prop}}, 0) d(\tau)$$

In time $O(n^2 \log \epsilon^{-1})$, the error 2-norm satisfies the contraction inequality

$$\|d(\tau) - d_*\|_2 \leq \epsilon\|d(0) - d_*\|_2,$$ for $d_* = \frac{2\pi}{n} 1$

The conversion of this inequality into an appropriate inequality on $\infty$-norms yields the result
Theorem

In the limit as \( n \to +\infty \) and \( \epsilon \to 0^+ \), the network \( S_{\text{circle,disk}} \), the law \( CC_{\text{AGREE \& PURSUE}} \), and the tasks \( T_{\text{dir}} \) and \( T_{\epsilon-\text{eqdstnc}} \) together satisfy:

1. If \( \delta(n) \geq \pi\left(1/k_{\text{prop}} - 2\right) \) as \( n \to +\infty \), then
   
   \[
   \text{TCC}_{\text{unidir}}(T_{\text{dir}}, CC_{\text{AGREE \& PURSUE}}) \in \Theta(n^2 r(n)^{-1}),
   \]

   otherwise if \( \delta(n) \leq \pi\left(1/k_{\text{prop}} - 2\right) \) as \( n \to +\infty \), then

   \[
   \text{TCC}_{\text{unidir}}(T_{\text{dir}}, CC_{\text{AGREE \& PURSUE}}) \in \Omega(n^3 + nr(n)^{-1}),
   \]

   \[
   \text{TCC}_{\text{unidir}}(T_{\text{dir}}, CC_{\text{AGREE \& PURSUE}}) \in O(n^2 r(n)^{-1});
   \]

2. If \( \delta(n) \) is lower bounded by a positive constant as \( n \to +\infty \), then

   \[
   \text{TCC}_{\text{unidir}}(T_{\epsilon-\text{eqdstnc}}, CC_{\text{AGREE \& PURSUE}}) \in \Omega(n^3 \delta(n) \log(n\epsilon)^{-1}),
   \]

   \[
   \text{TCC}_{\text{unidir}}(T_{\epsilon-\text{eqdstnc}}, CC_{\text{AGREE \& PURSUE}}) \in O(n^4 \log(n\epsilon^{-1})).
   \]
Comparison with leader election

- **Leader election task** is different from, but closely related to, $T_{\text{dir}}$
- **LCR algorithm** operates on a static ring network, and achieves leader election with time and total communication complexity, respectively, $\Theta(n)$ and $\Theta(n^2)$
- **Agree-and-pursue** law operates on robotic network with $r(n)$-disk communication topology, and achieves $T_{\text{dir}}$ with time and total communication complexity, respectively, $\Theta(r(n)^{-1})$ and $O(n^2r(n)^{-1})$

If wireless communication congestion is modeled by $r(n)$ of order $1/n$, then identical time complexity and the LCR algorithm has better communication complexity

Computations on a possibly disconnected, dynamic network are more complex than on a static ring topology
Conclusions

Robotic network model

- proximity graphs
- control and communication law, task, execution
- time, space, and communication complexity
- agree and pursue

Complexity analysis is **challenging** even in 1 dimension! Blend of mathematical tools required

Plenty of open issues and problems

- Asynchronism, quantization, delays
- What is best algorithm to achieve a task?
- What tools are useful to characterize complexity?
- How does combination of algorithms affect complexities?
Voronoi partitions

Let \((p_1, \ldots, p_n) \in \mathbb{Q}^n\) denote the positions of \(n\) points.

The **Voronoi partition** \(\mathcal{V}(P) = \{V_1, \ldots, V_n\}\) generated by \((p_1, \ldots, p_n)\)

\[
V_i = \{q \in \mathbb{Q} | \|q - p_i\| \leq \|q - p_j\|, \forall j \neq i\} = \mathbb{Q} \cap \bigcap_j \mathcal{H}\mathcal{P}(p_i, p_j)
\]

where \(\mathcal{H}\mathcal{P}(p_i, p_j)\) is half plane \((p_i, p_j)\)

![Diagram](3 generators, 5 generators, 50 generators)
Let \((p_1, \ldots, p_n) \in Q^n\) denote the positions of \(n\) points

The \(r\)-limited Voronoi partition \(V_r(P) = \{V_1,r, \ldots, V_n,r\}\) generated by \((p_1, \ldots, p_n)\)

\[V_i,r(P) = V_i(P) \cap B(p_i, r)\]

\(G_{\text{LD}}(r)\) is spatially distributed over \(G_{\text{disk}}(r)\)
$\mathcal{G}_D$ and $\mathcal{G}_D \cap \mathcal{G}_{\text{disk}}(r)$ computation

$\mathcal{G}_D$ and $\mathcal{G}_D \cap \mathcal{G}_{\text{disk}}(r)$ are not spatially distributed over $\mathcal{G}_{\text{disk}}(r)$