Semi-Autonomous Networks: Effective Control of Networked Systems through Design, Modeling, and Protocols

Airlie Chapman

Distributed Space Systems Lab (DSSL)
Robotics, Aerospace and Information Networks (RAIN)

University of Washington

Advisor: Mehran Mesbahi
Semi-Autonomous Networked Systems
Effective control...

Semi-Autonomous Networks

- Design
- Modeling
- Protocols

Network measures?
Favorable topological features?
Local features?

Network scalability?
Network decomposition?
Invariant system properties?

Extensions to agreement?
Extensions to nonlinear?
Families of network protocols?
Effective control...

- Design
- Modeling
- Protocols

Semi-Autonomous Networks

Network measures?
Favorable topological features?
Local features?

Network scalability?
Network decomposition?
Invariant system properties?

Extensions to agreement?
Extensions to nonlinear?
Families of network protocols?
Approach

Effective control...

Semi-Autonomous Networks

Design

Modeling

Protocols

Network measures? Favorable topological features? Local features?

Network scalability? Network decomposition? Invariant system properties?

Extensions to agreement? Extensions to nonlinear? Families of network protocols?
Effective control...

Network measures?
Favorable topological features?
Local features?

Network scalability?
Network decomposition?
Invariant system properties?

Extensions to agreement?
Extensions to nonlinear?
Families of network protocols?
The Network in the Dynamics

- Node $i$ dynamics

$$
\dot{x}_i(t) = -w_{ii}x_i(t) + \sum_{i \sim j} w_{ij}x_j(t) + u_i(t)
$$

$$
y_i(t) = x_i(t)
$$

Dynamics

$$
\dot{x}(t) = A(G)x(t) + B(S)u(t)
$$

$$
y(t) = C(R)x(t)
$$

- $A(G)$: Encodes the graph structure, e.g. Consensus: undirected unweighted simple graph $G$ and $A(G) = A(G) - \text{diag}(A(G)1)$

- Input node set $S = \{v_i, v_j, \ldots\}$, $B(S) = [e_i, e_j, \ldots]$

- Output node set $R = \{v_p, v_q, \ldots\}$, $C(R) = [e_p, e_q, \ldots]^T$
Beyond Linear Consensus

General Dynamics

\[ \dot{x}(t) = f(G, x(t), u(t)) \]
\[ y(t) = g(G, x(t), u(t)) \]

Example:
Human-Swarm Interaction

**Cartesian Product Networks**

Networks within Networks

![Networks diagram](image)

Approximate Product Networks

![Approximate networks diagram](image)

- Invariant features over the factor networks:
  - Controllability, stability, trajectory subspaces

---


Strong Structural Controllability

What can one say about controllability based on the $\mathcal{G}$ alone?

... Structural Controllability


### Metrics for Semi-Autonomous Networks

- **Mean tracking error due to a constant input**

\[
J_\mu = \frac{1}{n} \sum_{i=1}^{n} E_{\text{eff}}(v_i)
\]

- **Energy at the output due to a unit impulse input**

\[
J_\sigma = \frac{1}{n} \sum_{v_i \in \mathcal{N}(S)} E_{\text{eff}}(v_i)
\]

where \(v_i \in \mathcal{N}(S)\) if it neighbors a leader

**Effective Resistance:** \(E_{\text{eff}}(v_i)\) is the voltage drop between \(v_i\) and \(s^*\), when a 1 Amp current source is connected across them.

---


Design changes through Rewiring

- Circuit theory provides a way to predictably alter structure

Example: Wind Gust Alleviation

Network Measures
Semi-Autonomous Dynamics

Consensus Model

\[ \dot{x}_i(t) = \sum_{\{i,j\} \in E} \left( x_j(t) - x_i(t) \right) \]

\[ \dot{x}(t) = -L(G)x(t) \]

where \( L(G) = -A(G) + \text{diag}(A(G)1) \)

Influence Model

\[ \dot{x}_i(t) = \sum_{\{i,j\} \in E} \left( x_j(t) - x_i(t) \right) + \sum_{\{i,k\} \in E_R} \left( u_k(t) - x_i(t) \right) \]

\[ \dot{x}(t) = Ax(t) + Bu(t) \]

- What metrics capture the efficiency of semi-autonomous systems and how does this relate to the network structure?
Semi-Autonomous Model

- Influencing node set
  \( \mathcal{R} = (R, \mathcal{E}_R), |\mathcal{E}_R| = r \)
- Each agent in \( R \) is attached to exactly one agent in \( V \), composing the set \( \pi(\mathcal{E}_R) \)

**Dirichlet Matrix**

\[
A(G, \mathcal{R}) = -(L(G) + B(\mathcal{R})B(\mathcal{R})^T)
\]

**Influence Model**

\[
\dot{x}(t) = A(G, \mathcal{R})x(t) + B(\mathcal{R})u(t)
\]

Example:

\[
A(G, \mathcal{R}) = \begin{bmatrix}
-3 & 1 & 1 & 0 \\
1 & -2 & 1 & 0 \\
1 & 1 & -3 & 1 \\
0 & 0 & 1 & -2 \\
\end{bmatrix},
\]

\[
B(\mathcal{R}) = \begin{bmatrix}
1 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 1 \\
\end{bmatrix}
\]
Mean Tracking Measure

- Infinite time horizon convergence with $\tilde{x}(t) = x(t) - u_c 1$,

$$J(G, R, \tilde{x}(0)) = 2 \int_0^\infty \tilde{x}(t)^T \tilde{x}(t) dt = -\tilde{x}(0)^T A(G, R)^{-1} \tilde{x}(0)$$

Mean Tracking Measure

The expected quadratic performance cost:

$$J_\mu (G, R) = 2 \mathbb{E}_{\|\tilde{x}(0)\|=1} \left[ \int_0^\infty \tilde{x}(t)^T \tilde{x}(t) dt \right]$$

$$= -\frac{1}{n} \text{tr} \left( A(G, R)^{-1} \right)$$

Error signal $\tilde{x}(t) = x(t) - u_c 1$, where $u_c \in \mathbb{R}$ is the common mean of $u(t)$
Variance Damping Measure

- When $u(t)$ is a white noise vector, as $t \to \infty$, $\mathbb{E} [\dot{x}(t)\dot{x}(t)^T] = P(\mathcal{G}, \mathcal{R})$ (the controllability gramian)
- From the Lyapunov equation

$$A(\mathcal{G}, \mathcal{R})P(\mathcal{G}, \mathcal{R}) + P(\mathcal{G}, \mathcal{R})A(\mathcal{G}, \mathcal{R}) = -B(\mathcal{R})B(\mathcal{R})^T$$

Variance Damping Measure

The average steady state variance is:

$$J_\sigma (\mathcal{G}, \mathcal{R}) = \frac{2}{n} \text{tr} (P(\mathcal{G}, \mathcal{R}))$$

$$= -\frac{1}{n} \text{tr} \left( B(\mathcal{R})^T A(\mathcal{G}, \mathcal{R})^{-1} B(\mathcal{R}) \right)$$
Effective Resistance of a Graph

- Consider edges $\mathcal{E}$ and $\mathcal{E}_R$ in the graph model replaced with 1Ω resistors, and nodes $R$ shorted as a common node $r_0$
- $\left[ -A(G, R)^{-1} \right]_{ii}$ corresponds to node $v_i$’s effective resistance to $r_0$, denoted $E_{\text{eff}}(v_i)$. [Barooah and Hespanha 2006]

Average $\mathbf{E}(\tilde{x})$

$$J_\mu (G, R) = \frac{1}{n} \sum_{i=1}^{n} E_{\text{eff}}(v_i)$$

Average $\mathbf{Var}(\tilde{x})$

$$J_\sigma (G, R) = \frac{1}{n} \sum_{v_i \in \pi(\mathcal{E}_R)} E_{\text{eff}}(v_i)$$
Results over Trees $\mathcal{T}$

Finding $E_{\text{eff}}$ is relatively simple over tree graphs $\mathcal{T}$

- **Main path agents $\mathcal{M}$**: Set of agents that lies on any shortest paths between agents in $\mathcal{R}$
- **Subgraph $\mathcal{G}_\mathcal{M} = (\mathcal{M}, E_\mathcal{M})$**: Agents $\mathcal{M}$ and edges between them
- **Main path neighbor $\Gamma(v_i)$**: Closest agent to $v_i$ that is in $\mathcal{M}$

![Diagram showing a tree graph with nodes $v_1$ to $v_6$ and subgraph $\mathcal{M}$ highlighted]

**Average $E(\tilde{x})$ for trees**

$$J_\mu(\mathcal{T}, \mathcal{R}) = \frac{1}{n} \left( \sum_{v_i \in \mathcal{M}} E_{\text{eff}}(v_i) + \sum_{v_i \notin \mathcal{M}} \left[ E_{\text{eff}}(\Gamma(v_i)) + d(v_i, \Gamma(v_i)) \right] \right)$$

**Average $\text{Var}(\tilde{x})$ for trees**

$$J_\sigma(\mathcal{T}, \mathcal{R}) = \frac{|\mathcal{M}|}{n} \cdot J_\sigma(\mathcal{G}_\mathcal{M}, \mathcal{R})$$
Results over Trees and One Attached Agent \((\mathcal{T}, \mathcal{R}^i)\)

**Centrality Lemma**

\[ J_{\mu}(\mathcal{T}, \mathcal{R}^i) = \frac{1}{n} \sum_{j=1}^{n} d(v_i, v_j) + 1 \]

**Single Bounds**

\[ 2 - \frac{1}{n} \leq J_{\mu}(\mathcal{T}, \mathcal{R}^i) \leq \frac{1}{2}(n+1) \]

**Topology Independence**

\[ J_{\sigma}(G, \mathcal{R}^i) = \frac{1}{n} \]
Rayleigh’s Monotonicity Principle:
“If the edge resistance in a electrical network is decreased then the effective resistance between any two agents in the network can only decrease.”

Graphs and their underlying trees
For a graph $G$ any underlying tree $T$ has the property $J_\mu (G, R) \leq J_\mu (T, R)$ and $J_\sigma (G, R) \leq J_\sigma (T, R)$

Can one dynamically adapt the network to encourage/deter the effect of the influencing agents?
Network Rewiring and Reweighting
Network Design Problem

Edge Swaps:
- Maintain connected graph
- Decentralized, parallel, asynchronous
- Requires only local agent information of the graph structure

\[ \mathcal{I}(v_i) \] is the neighbors of \( v_i \) closer to some influence agent than \( v_i \)
Balancing Measures

Relationship between measures

\[ J_\mu (G, R) = J_\sigma (G, R) + \frac{1}{n} \sum_{v_i \notin \pi(E_R)} E_{\text{eff}}(v_i) \]

- For security, large \( J_\mu (G, R) \) (resistance to external control) and small \( J_\sigma (G, R) \) (external noise damping) is favorable
- Approach: Increase \( \sum_{v_i \notin \pi(E_R)} E_{\text{eff}}(v_i) \) while keeping \( J_\sigma (G, R) \) small
- Compacts the main path (Protocol 3)
  Elongates the remaining graph (Protocol 1)
Distributed Protocol for Trees

If $\exists v_j, v_k \in \mathcal{N}(v_i), v_j \neq v_k$, then:

- Protocol 1: $\{(v_j, v_k) \notin I(v_i)\}$ - Increases $J_\mu(T, R)$
- Protocol 3: $\{v_j, v_k \in I(v_i) \text{ and } v_i \notin \pi(E_R)\}$ - Decreases $J_\sigma(T, R)$
- Protocol 5: either condition of Protocol 1 and 3 - Guarantees?

Then remove edge $e_{ij}$ and add edge $e_{jk}$
Simulation of Protocol 5
Protocol 5 is a **finite signed potential game** \( \Rightarrow \) at least one Nash equilibrium

### Definition

With the objective to minimize some function value then

\[
\text{PoS} = \frac{\text{Best Equilibrium Value}}{\text{Optimal Value}} \quad \text{and} \quad \text{PoA} = \frac{\text{Worst Equilibrium Value}}{\text{Optimal Value}}
\]

For protocol 5:
- With cost \( \frac{1}{J_\mu (\mathcal{T}, \mathcal{R})} \) the PoS = 1 and PoA \( \leq r \)
- With cost \( J_\sigma (\mathcal{T}, \mathcal{R}) \) the PoS = 1 and PoA < \( \frac{11\sqrt{5}}{20} \approx 1.23 \)

Consequence: For \( r = 1 \), protocol 5 will always reach the optimal value for \( \frac{1}{J_\mu (\mathcal{T}, \mathcal{R})} \)
Swarms of Unmanned Aerial Vehicles (UAVs) cooperate using a sensor network (interaction network).

Wind gusts can be inadvertently amplified by the network.

Selectively choose sensors to turn on/off (network topology design) to improve gust disturbance rejection under constraints.
Network Reweighting

- Varying $R_t$ due to mis-identification of neighbors
- Varying $G_t$ due to comms limitations and edge failures
- Reweight the edges in the network to minimize $J_\mu$ or $J_\sigma$ with uncertain $G_t(w_t)$ and $R_t$

Equivalent to:

$$\min_{(w_1, w_2, \ldots, w_T)} \sum_{t=1}^{T} f(G_t, R_t, w_t) = \sum_{t=1}^{T} \left( J_\mu(G_t(w_t), R_t) + \frac{h}{2} w_t^T w_t \right)$$

s.t. $w_t \in \chi$

where $h > 0$, and $\chi$ is a “distributable” convex set. (sim. for $J_\sigma$)
Online Convex Optimization

- Game formulation to solve a uncertain convex optimization problem
- Setup: At each time step $t$
  - player takes an action in a convex set:
    - agents select edge weights $w_t$ in $\chi$
  - convex cost of the action is revealed:
    - agents calculate the new $f(G_t, R_t, \cdot)$
  - player pays a penalty:
    - agents weight selection costs $f(G_t, R_t, w_t)$
- Literature: Incremental gradient [e.g., Bertsekas ’10]; Online gradient descent [e.g., Hazan et al. ’07]; Distributed online optimization [e.g., Yan et al. ’10]
If costs are known ahead of time, i.e., \((G_1, R_1), (G_2, R_2), \ldots, (G_T, R_T)\), let the best \textit{static} weight selection be \(w^*\)

Regret:

\[
R_T = \sum_{t=1}^{T} (f(G_t, R_t, w_t) - f(G_t, R_t, w^*))
\]

over all possible \(\{(G_t, R_t)\}\)

Objective: Sublinear \(R_T\) or \(R_T / T \to 0\),

i.e., “on average” \((w_1, w_2, \ldots, w_T)\) performs as well as \((w^*, w^*, \ldots, w^*)\)

Example: Online gradient descent for strongly convex is \(O(\log(T))\), requiring only \(\frac{\partial f_t(w_t)}{\partial w_{ij}}\)
Main Steps

- Showed convexity of $f_t(w) := f(G_t, R_t, w_t)$

- A local agent form of the cost function gradient

$$\frac{\partial f_t(w)}{\partial w_{ij}} = -\frac{1}{2} \sum_{s \in R} (y^s_i - y^s_j)^2 + h w_{ij},$$

- For each $s \in R$, the vector $y^s = A_t^{-1} e_s$ is found via a distributed conjugate gradient algorithm

- Formulated as an distributed online gradient descent algorithm with

$$\sup_{f_T \in F} (R_T) = O(\log(T))$$
Movie for $J_\sigma$
Performance Run for $J_σ$

- Edge weights increase close to the disturbance inputs
- Edge memory of past disturbance location

**Variable disturbance locations**

**Fixed disturbance locations**

$t = 0$

$t = 1$

$t = 2$

$t = 3$

$t = 4$

$t = 5$

$t = 9$

$t = 10$

$t = 14$
Future Work
Distributed Online Design for Other Metrics

- e.g., Distributed online design to increase $\lambda_2(L(G))$ via edge reweighting


Task Allocation for Heterogenous Swarms

- Use online bipartite matching task, where tasks are presented and must be immediately matched, to cooperatively achieve a mission.
Use online bipartite matching task, where tasks are presented and must be immediately matched, to cooperatively achieve a mission.
Task Allocation for Heterogeneous Swarms

- Use online bipartite matching task, where tasks are presented and must be immediately matched, to cooperatively achieve a mission.
Task Allocation for Heterogenous Swarms

- Use online bipartite matching task, where tasks are presented and must be immediately matched, to cooperatively achieve a mission.
Task Allocation for Heterogenous Swarms

- Use online bipartite matching task, where tasks are presented and must be immediately matched, to cooperatively achieve a mission.
Task Allocation for Heterogenous Swarms

- Use online bipartite matching task, where tasks are presented and must be immediately matched, to cooperatively achieve a mission
Controllability for other Graph Products

- Use PBH test, and eigenvector properties of the Kronecker product representations to establish controllability for other graph products

\[ D_1 \]

\[ D_2 \]

\[ D = D_1 \otimes D_2 \]

- Direct Product: \( A(D_1 \times D_2) = A(D_1) \otimes A(D_2) \)
- Star Product: \( A(D_1 \star D_2) = I \otimes A(D_2) + A(D_1) \otimes A(D_2) \)
- Strong Product: \( A(D_1 \boxdot D_2) = A(D_1) \otimes I + I \otimes A(D_2) + A(D_1) \otimes A(D_2) \)
- Lexicographic Product: \( A(D_1 \bullet D_2) = A(D_1) \otimes 11^T + I \otimes A(D_2) \)
- Rooted Product with root \( i \): \( A(D_1 \circ_i D_2) = A(D_1) \otimes e_i e_i^T + I \otimes A(D_2) \)

Human-Swarm Interactions

- Design distributed protocols to interpret coarse inputs from a human operator

Conclusion

- **Graph Measures**
  - Presented two measures of semi-autonomous performance
  - Linked efficiency of semi-autonomous systems and the underlying network structure

- **Graph Design**
  - Proposed local protocols involving adjacent edge swaps that predictably alter these measures
  - Formed a distributed online optimization protocol to provide low regret in minimizing the measures

<table>
<thead>
<tr>
<th>Network</th>
<th>System Dynamics</th>
<th>Analysis Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph spectrum</td>
<td>Rate of convergence</td>
<td>Design</td>
</tr>
<tr>
<td>Random graphs</td>
<td>Random matrices</td>
<td>Interfaces/Modeling</td>
</tr>
<tr>
<td><strong>Graph factorization</strong></td>
<td><strong>Decomposition</strong></td>
<td>Modeling/Interfaces</td>
</tr>
<tr>
<td>Automorphisms</td>
<td>Homogeneity</td>
<td>Interfaces</td>
</tr>
<tr>
<td>Matching</td>
<td>Structural controllability</td>
<td>Interfaces</td>
</tr>
<tr>
<td>Effective resistance</td>
<td>Influence metrics</td>
<td>Design</td>
</tr>
</tbody>
</table>
Thanks to my committee, collaborators and lab mates...

- Professor Mesbahi
- Professor Burke
- Professor Klavins
- Professor Morgansen
- Professor Thomas
- Professor Shea-Brown
- Professor Mesbahi
- Professor Ran Dai
- Saghar Hosseini
- Dr Marzieh Nabi-Abdolyousefi
- Eric Schoof
Distributed Conjugate Gradient

For each node \( i \in V \)

Given \( y_i^s[0] \)

**Initialize**

- \( r_i[0] \leftarrow \sum_{j \sim i} a_{ij} x_j[0] - b_i^s \),
- \( p_i[0] \leftarrow -r_i[0] \)
- \( \tilde{r}_i[0] \leftarrow \text{Consensus on } r_i^2[0] \approx \frac{1}{n} r_i^T[0] r_i[0] \)

**While** \( |\tilde{r}_i[k]| > \varepsilon \)

1. \( \gamma_i[k] \leftarrow \sum_{j \sim i} a_{ij} p_j[k] \)
2. \( \tilde{p}_i[k] \leftarrow \text{Consensus on } p_i[k] \gamma_i[k] \approx \frac{1}{n} p_i^T A p_i \)
3. \( \alpha_i[k] \leftarrow \frac{\tilde{r}_i[k]}{\tilde{p}_i[k]} \)
4. \( y_i^s[k+1] \leftarrow y_i^s[k] + \alpha_i[k] p_i[k] \)
5. \( r_i[k+1] \leftarrow r_i[k] + \alpha_i[k] \gamma_i[k] \)
6. \( \tilde{r}_i[k+1] \leftarrow \text{Consensus on } r_i^2[k] \approx \frac{1}{n} r_i^T[k] r_i[k] \)
7. \( \beta_i[k] \leftarrow \frac{\tilde{r}_i[k+1]}{\tilde{r}_i[k]} \)
8. \( p_i[k+1] \leftarrow \beta_i[k] p_i[k] - r_i[k+1] \)
9. \( k \leftarrow k + 1 \)

**End**