コンセンサスな制御はいかが？

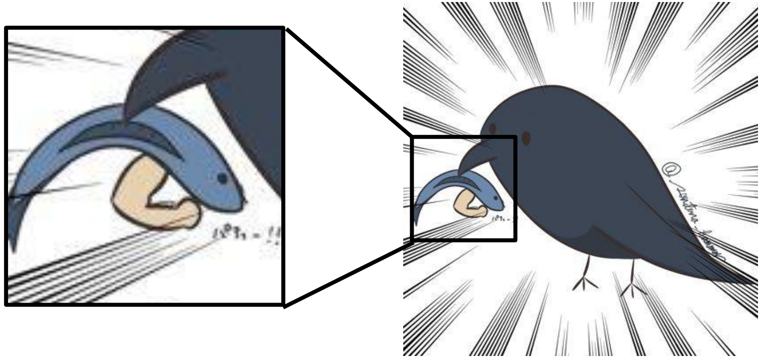
～合意制御についてちょっと触れてみよう～

ざきまつ

April 6, 2024

はじめに

- 初めまして, 「ざきまつ」です
- カラスに咥えられている魚が本体です
- ツイ廃です
- 大学院新入生です



X @santana_hammer

合意制御ってなんなのさ？

皆様,「合意制御」という言葉を聞いたことありますか？

聞いたことある方：ここから先は寝ていただいても大丈夫です．おやｽﾔｧ..

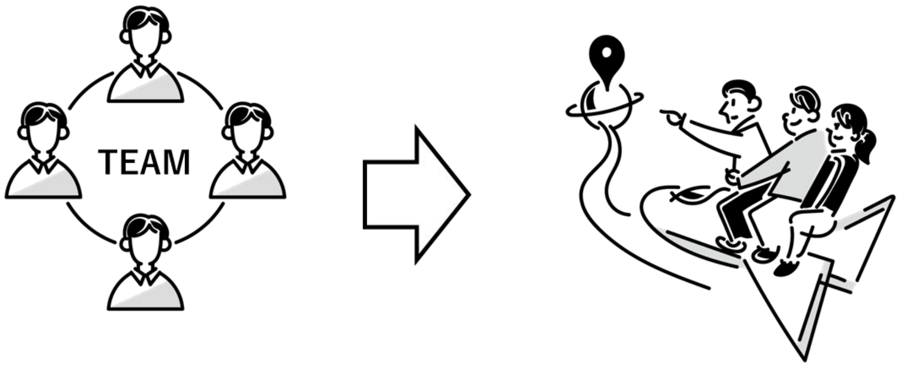
聞いたことない方：僕も研究室に配属されるまで聞いたことありませんでした.

仲間です，仲良くしてください.

マルチエージェントシステム (MAS)

複数の自律的なエージェントが相互作用し，目標を達成するために協力するシステム

- エージェント: 自律的に動作する個体（人間、ロボット、ソフトウェアなど）
- 自律: 外部からの直接的な介入なく，自分自身の判断で行動を選択
- 相互作用: エージェント同士が情報を共有したり，行動を調整したりするプロセス



マルチエージェントシステム (MAS)

どんな特徴があるのか？

- 分散性: システム全体の制御が集中型ではなく，各エージェントに分散している
- スケーラビリティ: エージェントを追加することで，システムの規模を柔軟に調整可能
- ロバスト性: 一部のエージェントが失敗しても、システム全体としては機能を維持可能
- 適応性: 外部環境の変化に対して，エージェントが自律的に行動を調整可能

合意制御ってなんなのさ？

合意制御(Consensus Control)

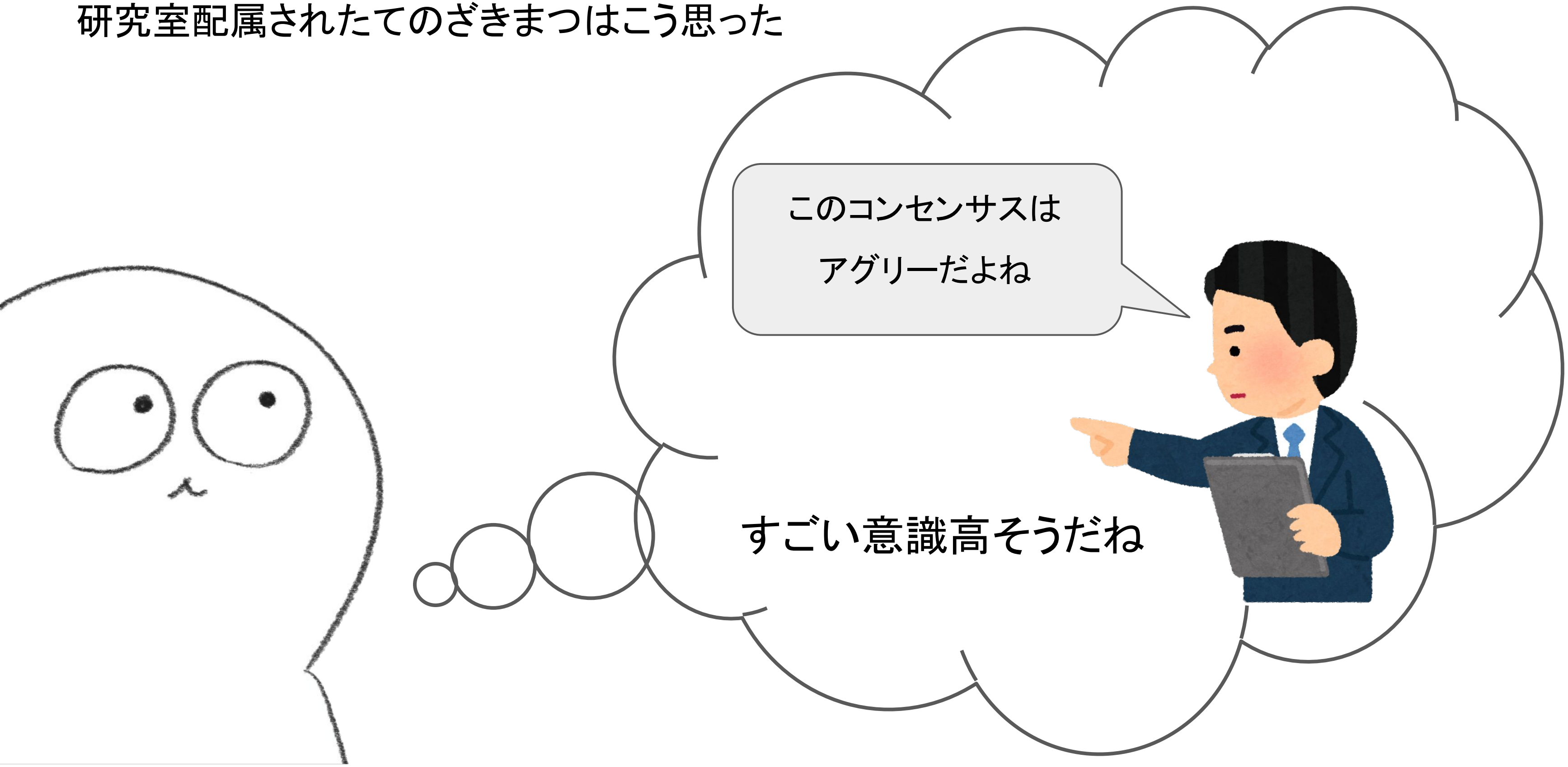
複数のエージェント(ロボット、センサー、ネットワーク上のノードなど)が情報を共有し合い、共通の目標に到達するための手法

目的

- 複数のエージェントが，互いの状態(位置、速度、方向など)について合意を達成
- エージェント間で共通の目標値や動作を実現

合意制御ってなんなのさ？

研究室配属されたてのざきまつはこう思った



合意制御の数式

合意アルゴリズム

$$u_i(t) = -\sum_{j \in \mathcal{N}_i} (x_i(t) - x_j(t))$$

where,

- $u_i(t)$ is the control input for agent i at time t.
- $\mathcal{N}_i$ denotes the set of neighbors of agent i.
- $x_i(t)$ and $x_j(t)$ represent the states of agent i and its neighbor j at time t, respectively.

- 通信可能(情報交換・参照が可能)なエージェントとの状態の差を計算
- 計算した差分を元に制御入力を決定
- いい塩梅を探しながら一つの状態に収束していく,

合意制御の数式

少し式変形

$$
\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \vdots \\ \dot{x}_n(t) \end{bmatrix}
= \begin{bmatrix} -\sum_{j=1}^{n} a_{1j}\left(x_1(t) - x_j(t)\right) \\ -\sum_{j=1}^{n} a_{2j}\left(x_2(t) - x_j(t)\right) \\ \vdots \\ -\sum_{j=1}^{n} a_{nj}\left(x_n(t) - x_j(t)\right) \end{bmatrix}
$$

$$
= \begin{bmatrix} \sum_{j=1}^{n} a_{1j} & -a_{12} & \cdots & -a_{1n} \\ -a_{21} & \sum_{j=1}^{n} a_{2j} & \ddots & -a_{2n} \\ \vdots & \ddots & \ddots & -a_{(n-1)n} \\ -a_{na} & \cdots & -a_{n(n-1)} & \sum_{j=1}^{n} a_{nj} \end{bmatrix}
\begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix}
$$

$$
\Rightarrow \qquad \dot{x}(t) = -Lx(t)
$$

ちょっとグラフ理論

グラフラプラシアン (Graph Laplacian)

- グラフの構造を表す行列で，次数行列と隣接行列の差によって定義される
- エージェント間の相対的な位置関係や，全体としての接続性の特性を分析するのに使用される

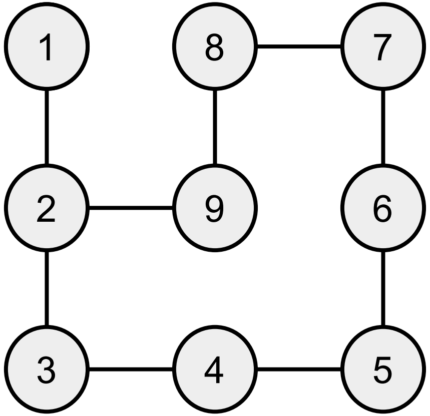
隣接行列 (Adjacency Matrix)

- エージェント（またはノード）間の接続を示す行列
- 行と列はシステム内のエージェントを表し、各要素の値はそのエージェント間の接続を表す
（通常，接続がある場合は 1，ない場合は0）

次数行列 (Degree Matrix)

- 各エージェント（ノード）がどれだけ多くのエージェントと接続しているかを示す対角行列
- 対角要素はそのノードの「次数」（接続の数）を示し，非対角要素は全て 0

ちょっとした例



$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \\ \dot{x}_5(t) \\ \dot{x}_6(t) \\ \dot{x}_7(t) \\ \dot{x}_8(t) \\ \dot{x}_9(t) \end{bmatrix} = - \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \\ x_6(t) \\ x_7(t) \\ x_8(t) \\ x_9(t) \end{bmatrix}$$

各時間系

連続時間バージョンと離散時間バージョンがある（当たり前）

連続時関系

$$u_i(t) = -\sum_{j\in\mathcal{N}_i}(x_i(t) - x_j(t))$$

離散時関系

$$u_i[k] = -\varepsilon\sum_{j\in\mathcal{N}_i}(x_i[k] - x_j[k])$$

離散時関系については，ペロン行列と呼ばれる行列を用いて，簡潔に表現できる.

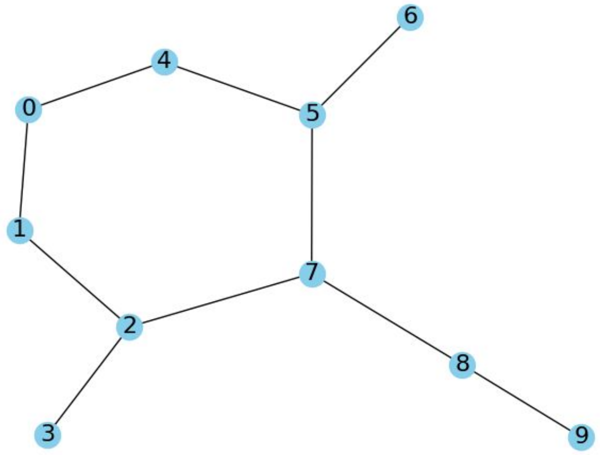
$$\begin{aligned} x[k+1] &= x[k] + u[k] \\ &= Px[k] \end{aligned}$$

where $P = I - \varepsilon L$ and $\varepsilon < 1/\Delta$.

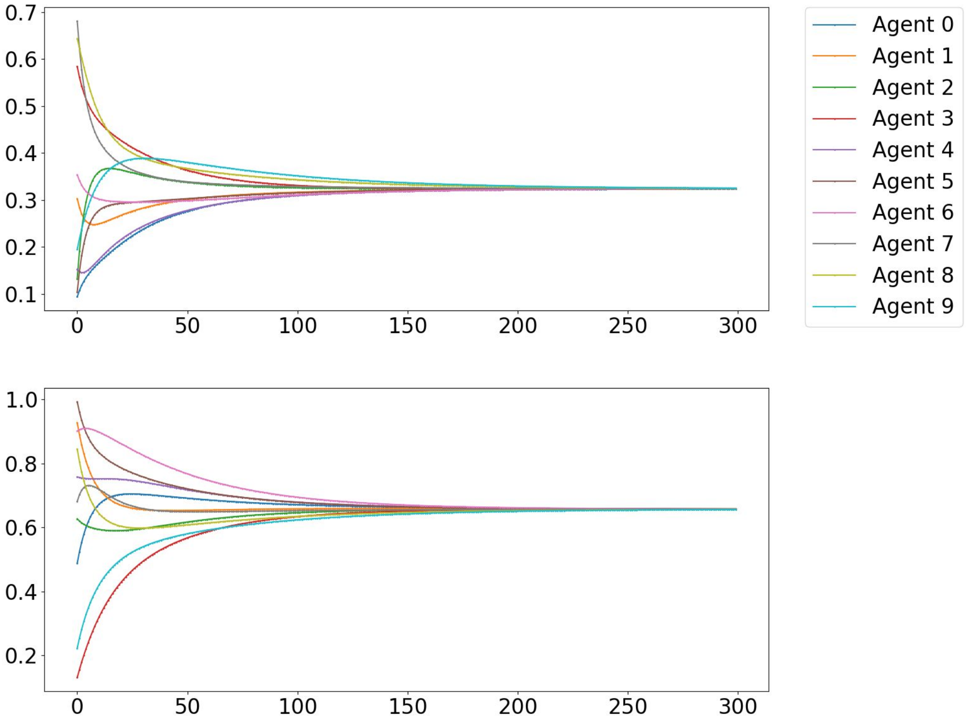
シミュレーション例

$$n = 10,\ d = 2,\ \varepsilon = 0.05,\ k \in [0, 300]$$

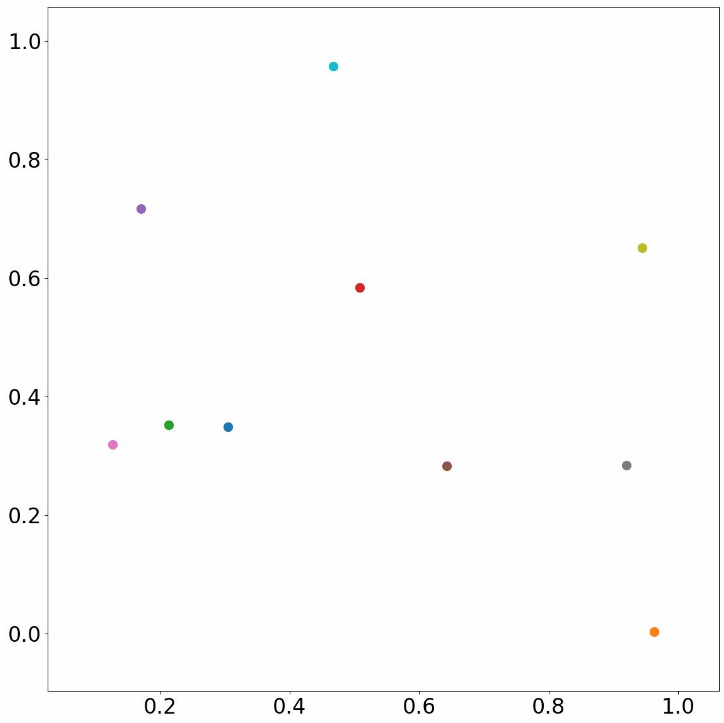
$$L = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$



シミュレーション例



0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
50
100
150
200
250
300
Agent 0
Agent 1
Agent 2
Agent 3
Agent 4
Agent 5
Agent 6
Agent 7
Agent 8
Agent 9
1.0
0.8
0.6
0.4
0.2

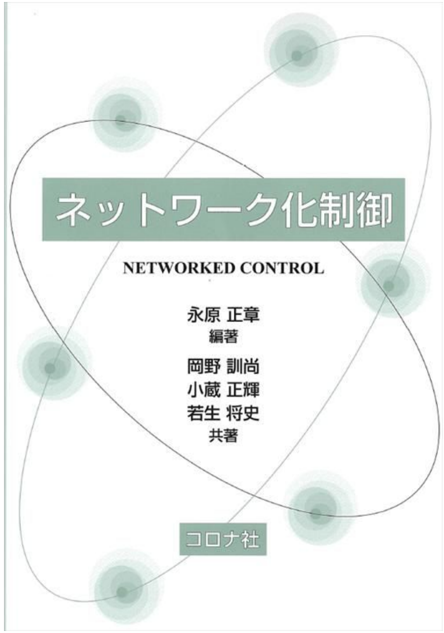


1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0

書籍等

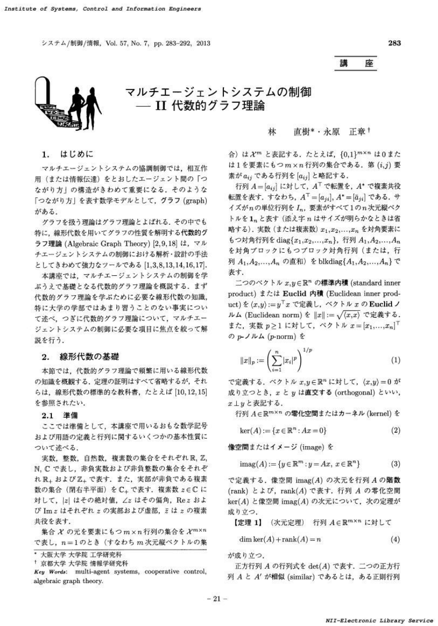
コロナ社

マルチエージェントシステムの制御



コロナ社

ネットワーク化制御



システム/制御/情報

マルチエージェントシステムの制御