3
Introduction to Synchronization in Nature and Multi-agent Cooperative Control on Communication Graphs

This book applies techniques from distributed cooperative control of multi-agent dynamical systems to synchronization, power sharing, and load balancing problems arising in electric power microgrids. This chapter presents the background and fundamental ideas needed for the book in cooperative control of multi-agent systems. For more information see [1].

Distributed networks of coupled dynamical systems have received much attention over the years because they occur in many different fields including biological and social systems, physics and chemistry, and computer science. Various terms are used in literature for phenomena related to collective behavior on networks of systems, such as flocking, consensus, synchronization, formation, rendezvous, and so on.

In the past few decades, an increasing number of industrial, commercial, and consumer applications have called for the coordination of multiple interconnected dynamical agents. Research on synchronized behaviors of networked cooperative systems, or multi-agent systems (MAS), and on distributed decision algorithms has received extensive attention due to their widespread applications in spacecraft, unmanned air vehicles (UAVs) [4], multi-vehicle systems [6], mobile robots, multipoint surveillance [3], sensor networks [5], networked autonomous teams, and so on. See [2], [3] for surveys of engineering applications of cooperative multi-agent control systems.

This chapter outlines the fundamental ideas used in the rest of the book to develop analysis and design methods for cooperative control of multi-agent dynamical systems on graphs. These ideas are detailed in [1]. The chapter starts by presenting an overview of synchronization behavior in nature and social systems. It is seen that distributed decisions made by each agent in a group based only on the information locally available to it from its neighbors can result in collective synchronized motion of an overall group. Mechanisms are given by which decisions can be made locally by each agent and informed leaders can guide collective behaviors by interacting directly with only a few agents. These ideas are extended to achieve synchronization mechanisms in physical oscillators and rotating synchronous generators in interconnected electric power systems.

The idea of a communication graph that models the information flows in a multi-agent group is introduced. The dependence of collective behaviors of a group on the type of information flow allowed between its agents is emphasized. Simple dynamical systems on communication graphs are presented and simple distributed control protocols given to direct their collective behavior. Such protocols are used

in the rest of the book for synchronization, voltage sharing, and load balancing in electric power microgrids.

3.1 Synchronization in Nature, Social Systems, and Coupled Oscillators

This section presents an overview of synchronization behavior in nature and social systems. It is seen that distributed decisions made by each agent in a group based on the information locally available to it by observing only its nearest neighbors can result in collective synchronized motion of an overall group. Distributed control Mechanisms are given by which decisions can be made locally by each agent and informed leaders can guide collective behaviors by interacting directly with only a few agents. These mechanisms occurring in nature can be tailored to provide synchronization and collective performance objectives in engineered systems and electric power networks. More information is given in [1].

The ability to coordinate multiple intercommunicating agents is important in many real-world decision tasks where it is necessary for agents to exchange information with each other. Distributed local decision and control algorithms are desirable and appear in many situations due to their computational simplicity, flexibility, and robustness to the loss of single agents. Applications of synchronization and distributed coordination of multi-agent systems have been observed in the evolution of distributed cooperation in social groups, synchronization in coupled dynamical oscillators, biological synchronization, and social networks. It has been observed that distributed decision using local neighbor responses can accurately model the panic behavior of crowds during emergency building evacuation. Synchronization and local distributed control mechanisms have been observed in the spread of infectious diseases in structured populations and in metabolic stability in random genetic nets. More details are given in [1].

3.1.1 Synchronization in Animal Motion in Collective Groups

The collective synchronized motions of animal social groups are among the most beautiful sights in nature [9]. Collective motions allow the group to achieve what the individual cannot. Benefits of aggregate motion include defense from predators, social and mating advantages, and group foraging for food [11]. Each individual has its own inclinations and motions, yet the aggregate motion makes the group appear to be a single entity with its own laws of motion, psychology, and responses to external events. Flocks of birds, schools of fish, and herds of animals are aggregate entities that take on an existence of their own due to the collective motion instincts of their individual members [12]. In collective motion situations, the important entity becomes the group, not the individual. Such synchronized and

responsive motion makes one think of choreographed motions in a dance, yet they are a product not of planned scripts, but of simple instantaneous decisions and responses by individual members [9],[10].

Distributed Local Neighborhood Protocols for Synchronization

To reproduce the collective motion of an animal group in computer animation has been a challenge. It would be impossible to script the motion of each individual using planned motions or trajectories. Analysis of groups based on social behaviors is complex, yet the individuals in collectives appear to follow simple rules that make their motion efficient, responsive, and practically instantaneous. The cumulative motion of animal groups can be programmed in computer animation by endowing each individual with the same few rules that allow it to respond to the situations it encounters. The responses of the individuals accumulate to produce the combined motion of the group.

In large social groups, each individual is aware only of the motions of its immediate neighbors. The field of perceptual awareness of the individual changes for different types of animal groups and in different motion scenarios. The collective synchronized motion of large groups can be captured by using a few simple rules governing the behavior of the individuals based on the observed behaviors of their neighbors. Individual motions in a group are the result of the balance of two opposing behaviors: a desire to stay close to the group, and a desire to avoid collisions with other individuals. Reynolds [12] has captured the tendencies governing the motions of individuals through his three rules. These rules depend on the motions of the neighbors of each individual in the group.

Reynolds' Rules [12]:

1. Collision avoidance: avoid collisions with neighbors
2. Velocity matching: match speed and direction of motion with neighbors
3. Flock centering: stay close to neighbors

There are many mechanisms for implementing Reynolds' rules in dynamical systems control. Of importance is the definition of an individual's 'neighborhood'. Consider motion in R^2 and define $(p_i(t), q_i(t))$ as the position of node *i* in the (x, y)-plane. Define the state of agent *i* as $x_i = [\,p_i \quad q_i\,]^T \in R^2$.

Define the distance between nodes *i* and *j* is

$$r_{ij} = \left| x_j - x_i \right| = \sqrt{(p_j - p_i)^2 + (q_j - q_i)^2} \tag{3.1}$$

Agents seek to attract to their neighbors according to the third rule. To see how this can be implemented mathematically Define an interaction radius ρ and the interaction neighborhood by $N_i = \{j : r_{ij} \le \rho\}$. If agent *i* is within a distance of ρ from agent *j*, it seeks to approach agent *j* to stay close to its neighbors. It is noted that radius ρ is different for different animal groups and different vehicles. Moreover, for some groups, such as flocks of birds in migration, the collision and interaction neighborhoods are not circular.

The dynamics used to simulate the individual group members can be very simple, yet realistic results are obtained. Consider agent motion in 2-D described by the dynamics

$$\dot{x}_i = u_i \tag{3.2}$$

with states $x_i = [\,p_i \;\; q_i\,]^T \in R^2$. This is a simple point-mass dynamics with velocity control inputs $u_i = [u_{pi} \;\; u_{qi}]^T \in R^2$.

A suitable law for flock centering is given by

$$u_i = \sum_{j \in N_i} a_{ij}(x_j - x_i) \tag{3.3}$$

which causes agent *i* to turn towards other agents inside the interaction neighborhood N_i. The flock centering gain $a_{ij} \ge 0$ is nonzero only if agent *j* is in the interaction neighborhood of agent *i*. If a_{ij} is large, then agent *i* seeks more vigorously to approach agent *j*. The flock centering protocol can be written in terms of the components of velocity as

$$\dot{p}_i = u_{pi} = \sum_{j \in N_i \setminus N_i^c} a_{ij}(p_j - p_i) \tag{3.4}$$

$$\dot{q}_i = u_{qi} = \sum_{j \in N_i \setminus N_i^c} a_{ij}(q_j - q_i) \tag{3.5}$$

The control protocols (3.3)-(3.5) are known as *cooperative local voting protocols* because each agent seeks to make the difference between his state and those of his neighbors equal to zero. That is, each agent seeks to achieve consensus with its neighbors. These protocols are *distributed* in the sense that they depend only on the local neighbor information as allowed by the communication graph topology.

An alternative protocols for flock centering is given by

$$u = \sum_{j \in N_i} a_{ij} \frac{(x_j - x_i)}{r_{ij}} = \sum_{j \in N_i} a_{ij} \frac{(x_j - x_i)}{\left| x_j - x_i \right|} \tag{3.6}$$

which is normalized by dividing by the distance between agents. Note that the sum is over components of a unit vector. Therefore, this law prescribes a desired direction of motion and result in motion of uniform velocity. By contrast, the law (3.3) gives velocities that are smaller if one is closer to one's neighbors.

Alternative methods to local cooperative protocols for implementing Reynolds' rules include potential field approaches [14] ,[15]. Potential field methods include approaches for obstacle avoidance and goal seeking, and are intimately related to the topics in this section.

3.1.2 Leadership in Animal Groups on the Move

We have seen that information can be transferred only locally between individual neighboring members of animal groups, yet this results in collective synchronized motions of the whole group. Local motion control protocols are based on a few simple rules that are followed by all individuals. However, in many situations, the whole group must move towards some goal, such as along migratory routes or towards food sources. In these cases, only a few informed individuals may have pertinent information about the required directions of motion.

Some species have evolved specialized mechanisms for conveying information about location and direction of goals. One example is the waggle dance of the honeybee that recruits hive members to visit food sources. Mechanisms of information transfer in groups involve questions such as how information about required motion directions, originally held by only a few informed individuals, can propagate through an entire group by simple mechanisms that are the same for every individual [11], [13]. It is shown by Couzin that only a small percentage of the individuals in a group need be directly aware of the location of the goal. If the group is connected in the sense that information can eventually propagate through the graph from one individual to any other, then a small percentage of informed individuals can influence the motions of the entire swarm to align to the goal state location.

These ideas are formalized mathematically in Section 3.3.2 about multi-agent synchronization to the trajectory of a leader node who can be directly observed by only a few agents.

3.1.3 Synchronization in Coupled Oscillators and Electric Power Systems

Coupled dynamical oscillators occur in many fields of research, including chemistry, physics, electrical engineering, and biology [8],[9],[10]. Kuramoto [8] analyzed local coupling in populations of chemical oscillators to study waves and turbulence.

He showed that there is a critical information coupling coefficient above which coupled oscillators synchronize. In 1995, Vicsek et al. [16] reported interesting results of collective behaviors of self-driven particles in phase transition. They showed that by using nearest neighbor interaction rules, all agents eventually move in the same direction.

Kuramoto [8] analyzed local coupling in populations of chemical oscillators to study waves and turbulence. He showed that there is a critical information coupling coefficient above which coupled oscillators synchronize. Interconnected Kuramoto oscillators have the dynamics

$$\dot{\theta}_i = \omega_i + \tfrac{K}{N}\sum_j \sin(\theta_j - \theta_i) \tag{3.7}$$

with oscillation frequency ω_i and coupling gain *K* [Strogatz 2000]. The oscillators are said to synchronize if $\dot{\theta}_i(t) - \dot{\theta}_j(t) \to 0 \; as \; t \to \infty \; \forall i, j$. It is seen that these systems are interconnected by terms not in $(\theta_j - \theta_i)$ as in (3.3), but by terms like $\sin(\theta_j - \theta_i)$.

Kuramoto took the oscillation frequencies as distributed about a mean value according to a probability density function. He showed that there is a coupling gain *K* below which the oscillators remain incoherent, i.e. do not synchronize. Above this gain the incoherent state becomes unstable, and the oscillators split into two groups- those with oscillation frequencies close to the mean synchronize to a mean frequency $\overline{\theta}$, while the others drift relative to this group.

In [17] synchronization is studied using the Lyapunov function $V = \tfrac{1}{2}\dot{\theta}^T\dot{\theta}$, with $\theta = [\theta_1 \cdots \theta_N]^T$. It is shown that $\dot{V} = -\tfrac{K}{N}\sum_{i,j}\cos(\theta_i - \theta_j)(\dot{\theta}_i - \dot{\theta}_j)^2$. It is shown that above some value for the coupling gain, and if the initial phase differences are in a certain compact set, the oscillators converge to the mean frequency $\overline{\omega} = \tfrac{1}{N}\sum_i \omega_i$.

The ideas of synchronization in Kuramoto oscillators can be applied to synchronization in electric power systems. The basic swing equations for rotating synchronous electric generators are given for the *i*-th generator as (assuming the lossless case) by

$$\dot{\omega}_i = -D_i\omega_i + \frac{\omega_{i0}}{M_i}P_{mi} - E_i^2 G_{ii} - P_{ei} = -D_i\omega_i + \frac{\omega_{i0}}{M_i}P_{mi} - E_i^2 G_{ii} - E_i\sum_{j\in N_i} E_j Y_{ij}\sin(\delta_i - \delta_j) \tag{3.8}$$

$$\dot{E}_i = -a_i E_i + u_i + b_i \sum_{j \in N_i} E_j \cos(\delta_i - \delta_j) \tag{3.9}$$

with frequency $\omega_i = \dot{\delta}_i$, mechanical power P_{mi}, electrical power P_{ei}, and admittance Y_{ij} capturing the networked interconnection structure between generators. A frequency equilibrium is characterized by frequency synchronization $\omega_i = \omega_j, \forall i, j$ and balanced power flow $Q_i(\delta_i) = P_{mi} - P_{ei} - E_i^2 G_{ii} = 0, \ \forall i$. In [18] it was shown that these interconnected systems are generalized Kuramoto oscillators, and conditions for synchronization to a common frequency are given.

Ortega [19] used a nonlinear passivity based approach for general lossy systems. Partial differential equations were solved for the controls, which have the form

$$u_i = k_1 E_i - k_2 \omega_i - b_i \sum_{j \in N_i} E_j \cos(\delta_i - \delta_j) + \sum_{j \in N_i} k_{ij} \omega_j \tag{3.10}$$

where k_{ij} are nonlinear functions. It is seen that these controls are given not by terms not in $(\delta_j - \delta_i)$ as in (3.3), but by terms like $\cos(\delta_j - \delta_i)$.

3.2 Communication Graphs for Interconnected Systems

Local synchronization protocols such as those just discussed capture very well the dynamics of intercoupled biological and animal groups, dynamical oscillators in chemistry and physics, and interconnected electric power systems. These protocols depend on the awareness of each individual of his neighbors. We have seen that the information flow between members of a social group is instrumental in determining the motion of the overall group. In this book we are concerned with the behaviors and interactions of dynamical systems that are interconnected by the links of a communication network. The fundamental control issues concern how the graph topology interacts with the local feedback control protocols of the agents to produce overall behaviors of the interconnected nodes.

The communication network interconnecting the dynamical systems can be modeled as a graph with directed edges corresponding to the allowed flow of information between the systems. See [1], [33], [34] for more information. The systems are modeled as the nodes in the graph and are sometimes called agents. We call this the study of multi-agent dynamical systems on graphs.

A *graph* is a pair $G = (V, E)$ with $V = \{v_1, \cdots, v_N\}$ a set of *N* nodes or vertices and *E* a set of edges or arcs. Elements of *E* are denoted as (v_i, v_j) which is

termed an edge or arc from v_i to v_j, and represented as an arrow with tail at v_i and head at v_j. The edges represent the allowed flow of information in the graph. We assume the graph is simple, i.e. $(v_i, v_i) \notin E, \forall i$ no self-loops, and no multiple edges between the same pairs of nodes. Edge (v_i, v_j) is said to be outgoing with respect to node v_i and incoming with respect to v_j; node v_i is termed the parent and v_j the child. The in-degree of v_i is the number of edges having v_i as a head. The out-degree of a node v_i is the number of edges having v_i as a tail. The set of (in-) neighbors of a node v_i is $N_i = \{v_j : (v_j, v_i) \in E\}$, i.e. the set of nodes with edges incoming to v_i. The number of neighbors $|N_i|$ of node v_i is equal to its in-degree.

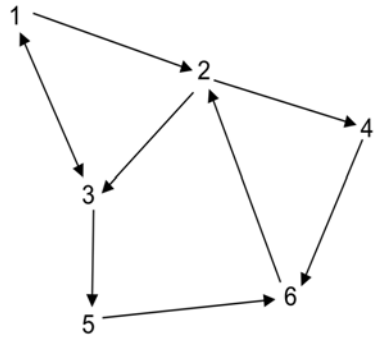


Fig. 3.1 A directed graph.

If the in-degree equals the out-degree for all nodes $v_i \in V$ the graph is said to be *balanced*. If $(v_i, v_j) \in E \Rightarrow (v_j, v_i) \in E, \forall i, j$ the graph is said to be *bi-directional*, otherwise it is termed a directed graph or digraph. Associate with each edge $(v_j, v_i) \in E$ a weight $a_{ij} \geq 0$. Note the order of the indices in this definition. One has $a_{ij} > 0$ only if there is an edge from node *j* to node *i*. A graph is said to be *undirected* if $a_{ij} = a_{ji}, \forall i, j$, that is, if it is bi-directional and the weights of edges (v_i, v_j) and (v_j, v_i) are the same.

A directed path is a sequence of nodes $v_0, v_1, \cdots, v_r$ such that $(v_i, v_{i+1}) \in E, i \in \{0, 1, \cdots, r-1\}$. Node v_i is said to be connected to node v_j if there is a directed path from v_i to v_j. The distance from v_i to v_j is the length of the shortest path from v_i to v_j. Graph G is said to be strongly connected if v_i, v_j are connected for all distinct nodes $v_i, v_j \in V$. For bidirectional and undirected graphs, if there is a directed path from v_i to v_j, then there is a directed path from v_j to v_i, and the graph is called simply 'connected'.

If a graph is not connected, it is disconnected. A *component* of an undirected graph is a connected subgraph that is not connected to the remainder of the graph.

Information in social networks only travels directly between immediate neighbors in a graph, as decreed by the nonzero edge weights a_{ij}. Nevertheless, if a graph is connected, then this locally transmitted information travels finally to every agent in the graph.

A (directed) tree is a connected digraph where every node except one, called the root, has in-degree equal to one. A spanning tree of a digraph is a directed tree formed by graph edges that connects all the nodes of the graph. A graph is said to have a spanning tree if a subset of the edges forms a directed tree. This is equivalent to saying that all nodes in the graph are reachable from a single (root) node by following the edge arrows. A graph may have multiple spanning trees. Define the root set or leader set of a graph as the set of nodes that are the roots of all spanning trees. If a graph is strongly connected it contains at least one spanning tree. In fact, if a graph is strongly connected then all nodes are root nodes.

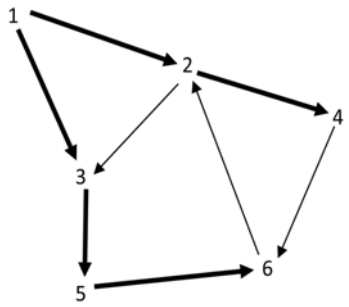


Fig. 3.2 A spanning tree for the graph in Fig. 3.1 with root node 1.

Graph Matrices- Algebraic Graph Theory

Graph structure and properties can be studied by examining the properties of certain matrices associated with the graph. This is known as algebraic graph theory [33],[34].

Given the edge weights a_{ij}, a graph can be represented by an adjacency or connectivity matrix $A = [a_{ij}]$ with weights $a_{ij} > 0 \; if \; (v_j, v_i) \in E$ and $a_{ij} = 0$ otherwise. Note that $a_{ii} = 0$. Define the weighted in-degree of node v_i as the i-th row sum of A

$$d_i = \sum_{j=1}^{N} a_{ij} \tag{3.11}$$

and the weighted out-degree of node v_i as the i-th column sum of A

$$d_i^o = \sum_{j=1}^{N} a_{ji} \tag{3.12}$$

The adjacency matrix A of an undirected graph is symmetric, $A = A^T$. A graph is said to be (*weight) balanced* if the weighted in-degree equals the weighted out-degree for all i. If all the nonzero edge weights are all equal to 1, this is the same as the definition of balanced graph. An undirected graph is weight balanced, since if $A = A^T$ and then the i-th row sum equals the i-th column sum. We may be loose at times and refer to node v_i simply as node i, and refer simply to in-degree, out-degree, and the balanced property, without the qualifier 'weight', even for graphs having non-unity weights on the edges.

Graph Laplacian Matrix. Define the diagonal in-degree matrix $D = diag\{d_i\}$ and the (weighted) graph Laplacian matrix $L = D - A$. Note that L has all row sums equal to zero. Many properties of a graph may be studied in terms of its graph Laplacian. In fact, we shall see that the Laplacian matrix is of extreme importance in the study of dynamical multi-agent systems on graphs. The eigenvalues of L specify many properties of the underlying graph topology.

3.3 Cooperative Control of Multi-Agent Systems on Communication Graphs

The main interest of this book is cooperative control of multi-agent dynamical electric power systems interconnected by a communication graph topology. The links of the graph represent the allowed information flow between the systems. In cooperative control systems on graphs, there are intriguing interactions between the individual agent dynamics and the topology of the communication graph. The graph topology may severely limit the possible performance of any control laws used by the agents. Moreover, in cooperative control on graphs, all the control protocols must be distributed in the sense that the control law of each agent is only allowed to depend on information from its immediate neighbors in the graph topology. If enough care is not shown in the design of the local agent control laws, the individual agent dynamics may be stable, but the networked systems on the graph may exhibit undesirable behaviors.

In cooperative control, each agent is endowed with its own state variable and dynamics. A fundamental problem in multi-agent dynamical systems on graph is the design of distributed protocols that guarantee consensus or synchronization in the sense that the states of all the agents reach the same value. The states could represent vehicle headings or positions, estimates of sensor readings in a sensor network, oscillation frequencies, frequencies and voltages in an electric power network, and so on.

The principles of cooperative multi-agent systems that we have discuss in this chapter have been developed by many researchers. Distributed decision and parallel computation algorithms were studied and developed in [20]. The field of cooperative control for multi-agent systems was started in earnest with the two seminal papers in 2003 [21] and 2004 [22]. Early work was furthered in [23],[24],[25],[26]. In the development of cooperative control theory, work was generally done initially for simple systems including first-order integrator dynamics in continuous time and discrete time. Then, results were established for second-order systems and higher-order dynamics. Applications were made to vehicle formation control and graphs with time-varying topologies and communication delays. See [27],[28][29][30][31][32].

Early work focused on reaching consensus or synchronization of networks of anonymous dynamical systems, where all agents have the same role. This is known as the leaderless consensus problem or the cooperative regulator problem. Later work focused on synchronization to the dynamics of a leader or root node which generates a command target trajectory. This is the controlled consensus problem or the cooperative tracker.

3.3.1 Consensus and the Cooperative Regulator Problem

Given dynamical systems at each node *i* with state $x_i(t)$, we wish to find controllers that make all the states the same, specified as follows.

Definition 3.1. Consensus- The Cooperative Regulator Problem. *Find a distributed control protocol for each agent i that drives all states to the same constant steady-state values* $x_i(t) \to x_j(\mathrm{t}) \to \mathrm{const}, \forall i, j$. *This value is known as a consensus value.*

The control protocols are required to be *distributed* in that the control for agent *i* is only allowed to depend on the state of agent *i* and its neighbors $j \in N_i$ in the graph topology.

Consider the case where all *N* nodes of the graph *G* have scalar first-order single-integrator dynamics

$$\dot{x}_i = u_i \tag{3.13}$$

with $x_i, u_i \in R$. This corresponds to endowing each node or agent with a memory.

Consider the local control protocols for each agent *i*

$$u_i = \sum_{j \in N_i} a_{ij}(x_j - x_i) \tag{3.14}$$

with a_{ij} the graph edge weights. This control is distributed in that it only depends on the immediate neighbors $j \in N_i$ of node *i* in the graph topology. This is known as a *local voting protocol* since the control input of each node depends on the difference between its state and all its neighbors. Note that if these states are all the same, then $\dot{x}_i = u_i = 0$. In fact, it will be seen that, under certain graph connectivity conditions, this protocol indeed drives all states to the same consensus value.

In the control protocol (3.14) there are only appearances of the states of node *i* and its neighbors. There is no external reference input. Therefore, this distributed controller is known as a *cooperative regulator*.

We wish to show that protocol (3.14) solves the consensus problem and to determine the consensus value reached. Write the closed-loop dynamics as

$$\dot{x}_i = \sum_{j \in N_i} a_{ij}(x_j - x_i) \tag{3.15}$$

$$\dot{x}_i = -x_i \sum_{j \in N_i} a_{ij} + \sum_{j \in N_i} a_{ij} x_j = -d_i x_i + \begin{bmatrix} a_{i1} & \cdots & a_{iN} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} \tag{3.16}$$

with d_i the in-degree. Define the global state vector $x = [x_1 \cdots x_N]^T \in R^N$ and the diagonal matrix of in-degrees $D = diag\{d_i\}$. Then the global dynamics are given by

$$\dot{x} = -Dx + Ax = -(D - A)x \tag{3.17}$$

$$\dot{x} = -Lx \tag{3.18}$$

With $L = D - A$ the Laplacian matrix. Note that the global control input vector $u = [u_1 \cdots u_N]^T \in R^N$ is given by

$$u = -Lx \tag{3.19}$$

It is seen that, using the local voting protocol (3.14), the closed-loop dynamics (3.18) depends on the graph Laplacian matrix *L*. We shall now see how the evolution of first-order integrator dynamical systems on graphs depends on the graph properties through the Laplacian matrix. The eigenvalues λ_i of *L* are instrumental in this analysis [1]. Order the eigenvalues of *L* in increasing magnitude as $\{\lambda_1, \lambda_2, \cdots, \lambda_N\}$, with *N* the number of nodes in the graph.

Note that the Laplacian matrix is $L = D - A$ and as such has row sum equal to zero. Therefore $L\underline{1} = 0$, with $\underline{1} = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T$ the vector of ones. Therefore the first eigenvalue of *L* is $\lambda_1 = 0$ with a right eigenvalue of $v_1 = \underline{1}$. It is known that the eigenvalues of *L* all have nonnegative real parts [1]. It may have several eigenvalues at zero, and the remainder of the eigenvalues have positive real parts. If the graph has a spanning tree, the only eigenvalue at zero is λ_1 and all other eigenvalues of *L* have positive real parts [1].

The dynamics given by (3.18) has a system matrix of $-L$, and hence has eigenvalues in the open left-half plane. At steady-state, according to (3.18) one has

$$0 = -Lx_{ss} \tag{3.20}$$

Therefore, the steady-state global state is in the nullspace of *L*. If the graph has a spanning tree, then $\lambda_1 = 0$ and all other eigenvalues of *L* have positive real parts. Then, (3.18) is a Type-1 system and all states reach constant steady-state values. Since $L\underline{1} = 0$, (3.20) becomes $0 = -cL\underline{1}$ and all agents reach the same consensus value of *c*.

The next result formalizes the consensus properties of the distributed control protocols (3.14).

Theorem 3.1. Cooperative Regulation for First-order Systems. *The cooperative regulator protocol (3.14) guarantees consensus of the single-integrator dynamics (3.13) if and only if the graph has a spanning tree. Then, all node states come to the same steady-state values* $x_i = x_j = c, \forall i, j$. *The consensus value is given by*

$$c = \sum_{i=1}^{N} p_i x_i(0) \tag{3.21}$$

where $w_1 = \begin{bmatrix} p_1 & \cdots & p_N \end{bmatrix}^T$ *is the normalized left eigenvector of the Laplacian L for* $\lambda_1 = 0$. *That is* $w_1^T L = 0, \|w_1\| = 1$. *Finally, consensus is reached with a time constant given by*

$$\tau = 1/\operatorname{Re}\{\lambda_2\} \tag{3.22}$$

with λ_2 *the second eigenvalue of L, known as the Fiedler eigenvalue.*
Proof: *[1].*

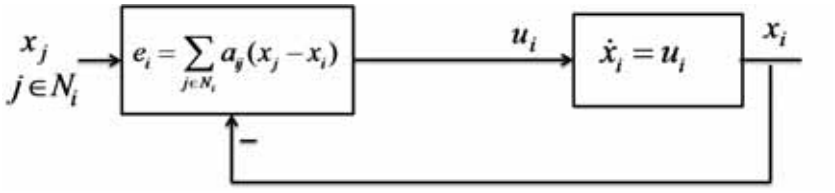


Fig. 3.3 Multi-agent Cooperative Regulator

Fig. 3.3 shows the cooperative regulator controller. It is the same as the single-agent regulator, except that now each agent must take into account the states $x_j(t)$ of its neighbors in determining its error. This is formalized by defining the *local neighborhood consensus error* as

$$e_i = \sum_{j \in N_i} a_{ij}(x_j - x_i) \tag{3.23}$$

According to the theorem, if the graph has a spanning tree, then $e_i(t) \to 0, \forall i$ if and only if the consensus problem is solved, that is, $x_i(t) \to x_j(t) \to \text{const}, \forall i, j$.

It is important to note that in the single-agent regulator problem, the state goes to zero. By contrast, in the multi-agent cooperative regulator problem, the states go to constant values that depend on the initial conditions of all agents in the graph, as per (3.21). This is a consequence of the interrelation in interconnected multi-agent systems between the control protocols (3.14) and the communication graph topology, as captured in the eigenvalues of the Laplacian matrix *L*.

3.3.2 Synchronization and the Cooperative Tracker Problem

Consider now the situation in Fig. 3.4 which depicts a leader or target node with state $x_0(t)$. Now, one desires to find controllers that make all the states synchronize to the leader's state, specified as follows.

Definition 3.2. Synchronization- The Cooperative Tracker Problem. *Find a distributed control protocol for each agent i that drives all states to the state of the leader node,* $x_i(t) \to x_0(t), \forall i$.

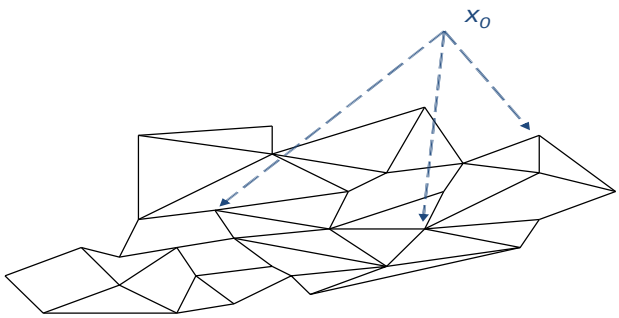


Fig. 3.4 Communication Graph with Leader node x_0

This is known as the synchronization or *cooperative tracker problem*. There has been much discussion of the terms 'consensus' and 'synchronization' in the literature, with ancillary terms appearing such as 'leaderless consensus', 'synchronization with or without leader', etc. This is cleared up by using simply the terms 'cooperative regulator', and 'cooperative tracker.

To solve this problem for the first-order integrator dynamics (3.13), consider the distributed cooperative tracker protocols

$$u_i = \sum_{j \in N_i} a_{ij}(x_j - x_i) \quad + g_i(x_0 - x_i) \tag{3.24}$$

with a_{ij} the graph edge weights. In this protocol, the gains $g_i \geq 0$ are known as pinning gains. One has $g_i > 0$ only if node *i* has a direct edge from the leader node. As depicted in Fig. 3.4, the intent is that only a small percentage of nodes has $g_i > 0$, and hence has direct access to the leader's state. Nevertheless, we shall

now see that under certain graph connectivity conditions, this protocol indeed solves the cooperative tracker problem.

Write the closed-loop dynamics as

$$\dot{x}_i = \sum_{j \in N_i} a_{ij}(x_j - x_i) \quad + g_i(x_0 - x_i) \tag{3.25}$$

$$\begin{aligned} \dot{x}_i &= -x_i \sum_{j \in N_i} a_{ij} + \sum_{j \in N_i} a_{ij} x_j + g_i x_0 - g_i x_i \\ &= -(d_i + g_i)x_i + \begin{bmatrix} a_{i1} & \cdots & a_{iN} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} + g_i x_0 \end{aligned} \tag{3.26}$$

with d_i the in-degree. Define the global state vector $x = [x_1 \cdots x_N]^T \in R^N$, the diagonal matrix of in-degrees $D = diag\{d_i\}$, and the diagonal pinning gain matrix $G = diag\{g_i\}$. Then the global dynamics are given by

$$\dot{x} = -(D+G)x + Ax = -(D+G-A)x + G\underline{x}_0 \tag{3.27}$$

$$\dot{x} = -(L+G)x + G\underline{x}_0 \tag{3.28}$$

where $\underline{x}_0 = \underline{1}x_0 = \begin{bmatrix} x_0 & \cdots & x_0 \end{bmatrix}^T$.

The properties of the cooperative tracker protocol (3.24) depend on the graph properties through the pinned Laplacian matrix *(L+G)*. The eigenvalues λ_i of *(L+G)* are instrumental in this analysis [1].

The analyze the behavior of the cooperative tracker, define the tracker synchronization errors

$$\delta_i(t) = x_0(t) - x_i(t) \tag{3.29}$$

and the global tracking error vector $\delta = \begin{bmatrix} \delta_1 & \cdots & \delta_N \end{bmatrix}$, with N the number of agents. Assume the leader node dynamics are

$$\dot{x}_0 = 0 \tag{3.30}$$

Then the global synchronization error dynamics is

$$\begin{aligned}\dot{\delta}(t) &= \dot{\underline{x}}_0(t) - \dot{x}(t) \\ &= (L+G)x - G\underline{x}_0\end{aligned} \tag{3.31}$$

However, since $L\underline{1} = 0$, one has $L\underline{x}_0 = 0$ and

$$\dot{\delta} = (L+G)x - (L+G)\underline{x}_0 = -(L+G)\delta \tag{3.32}$$

Hence, the global synchronization error goes to zero if *(L+G)* is stable. The next theorem gives the conditions.

Theorem 3.2. Cooperative Tracking for First-order Systems. *The cooperative tracking protocol* (3.24) *guarantees tracking of the single-integrator dynamics (3.13) if and only if the graph has a spanning tree. Then,* x $_0$.
Proof: *[1].*

The proof uses the fact that, if the graph has a spanning tree, all eigenvalues of *(L+G)* have positive real parts, so that (3.32) is a stable system.

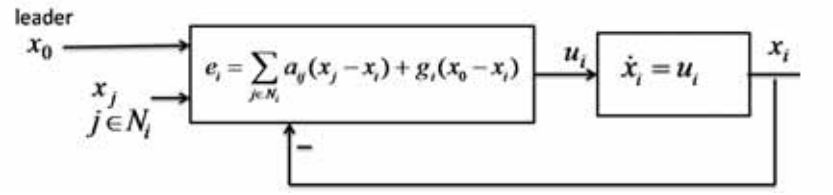


Fig. 3.5 Multi-agent Cooperative Tracker

Fig. 3.5 shows the cooperative tracker. It is the same as the single-agent tracker, except that now each agent must take into account the states $x_j(t)$ of its neighbors in determining its error, and the state of the leader node if $g_i > 0$. This is formalized by defining the *local neighborhood tracking error* as

$$e_i = \sum_{j \in N_i} a_{ij}(x_j - x_i) \quad + g_i(x_0 - x_i) \tag{3.33}$$

According to the theorem, if the graph has a spanning tree, then $e_i(t) \to 0, \forall i$ if and only if the consensus problem is solved, that is, $\delta_i(t) = x_0(t) - x_i(t) \to 0 \ \forall \mathrm{i}$.

3.3.3 More General Agent Dynamics and Vector States

The preceding results can be extended to more general agent dynamics, including vector states and general state variable dynamics.

Vector States.

We have presented cooperative regulator results for scalar first-order integrator dynamics (3.13), that is

$$\dot{x}_i = u_i \tag{3.34}$$

with scalar $x_i, u_i \in R$. Suppose now that the states and controls are vectors so that $x_i, u_i \in R^n$. Then the preceding developments can be extended using the idea of Kronecker product [35].

Consider two matrices $A \in R^{p\times q}, B \in R^{s\times t}$, denote $A = [a_{ij}]$. The (left) Kronecker product is given as

$$A \otimes B = [a_{ij}B] \in R^{ps\times qt} \tag{3.35}$$

For vector states, the integrator dynamics are treated as follows. If the state and input are vectors, then the step from (3.16) to (3.17) cannot be taken, for note that the states in (3.16) are vectors. Therefore, every element in *D* and *A* in (3.17) and (3.18) must be multiplied by the $n \times n$ identity matrix *I*. This yields

$$\dot{x} = -\big((D - A) \otimes \mathrm{I}\big) x = -(L \otimes I)x \tag{3.36}$$

Theorem 3.1 remains valid.

For the cooperative tracker, equations (3.28) and (3.32) must be replaced respectively by

$$\dot{x} = -\big((L + G) \otimes \mathrm{I}\big) x + (G \otimes I)\underline{x}_0 \tag{3.37}$$

$$\dot{\delta} = -\big((L + G) \otimes \mathrm{I}\big)\delta \tag{3.38}$$

Theorem 3.2 remains valid.

General State Variable Agent Dynamics.

Consider *N* agents interconnected on a communication graph having general linear time-invariant state variable dynamics

$$\dot{x}_i = Ax_i + Bu_i \tag{3.39}$$

with vector states and controls $x_i \in R^n, u_i \in R^m$. Consider the local neighborhood tracking errors (3.33) and the distributed cooperative tracker protocols

$$u_i = cKe_i = cK\left(\sum_{j \in N_i} a_{ij}(x_j - x_i) \quad + g_i(x_0 - x_i) \right) \tag{3.40}$$

with *K* a state variable feedback gain matrix and $c > 0$ a coupling gain. The closed-loop dynamics are

$$\dot{x}_i = Ax_i + cBK\left(\sum_{j \in N_i} a_{ij}(x_j - x_i) \quad + g_i(x_0 - x_i) \right) \tag{3.41}$$

Define the global state vector $x = vec\{x_i\} \equiv \left[x_1^T \quad x_2^T \quad \cdots \quad x_N^T \right]^T \in R^{nN}$ and the leader's global vector as $\underline{x}_0 = vec\{x_0\} \equiv (\underline{1} \otimes x_0) = \left[x_0^T \quad x_0^T \quad \cdots \quad x_0^T \right]^T \in R^{nN}$. Then using a development like (3.26), the global closed-loop dynamics are sen to be

$$\dot{x} = \left[(I_N \otimes A) - c\left((L+G) \otimes BK \right) \right] x + c\left((L+G) \otimes BK \right) \underline{x}_0 \tag{3.42}$$

Assume the leader's dynamics are

$$\dot{x}_0 = Ax_0 \tag{3.43}$$

Then the global error dynamics are

$$\dot{\delta} = \left[(I_N \otimes A) - c\left((L+G) \otimes BK \right) \right] \delta \tag{3.44}$$

There are many methods for selecting *c* and *K* to guarantee synchronization to the leader's state. One result is the following from [36].

Theorem 3.3. Cooperative Tracking for State Variable Multi-agent Systems. *Consider multi-agent dynamics (3.39) with distributed controls (3.40). Suppose that* (A, B) *is stabilizable. Select positive definite design matrices* $Q = Q^T \in \mathbb{R}^{n\times n}$ *and* $R = R^T \in \mathbb{R}^{m\times m}$, *Design the feedback control gain* K *as*

$$K = R^{-1}B^T P, \tag{3.45}$$

where P *is the unique positive definite solution of the control algebraic Riccati equation (ARE)*

$$0 = A^T P + PA + Q - PBR^{-1}B^T P. \tag{3.46}$$

Then under Assumption 1, the synchronization error dynamics (9) is asymptotically stable if the coupling gain

$$c \geq \frac{1}{2\underline{\lambda}_R} \tag{3.47}$$

with $\underline{\lambda}_R = min_{i\in\mathcal{N}} \mathrm{Re}(\lambda_i)$.
Proof: *[36].*

Note that this design has two parts, a local design procedure based on (3.45),(3.46) to select the gain matrix *K*, and a global design condition (3.47) based on the graph topology to select the coupling gains *c*. The graph topology is captured here in terms of $\underline{\lambda}_R = min_{i\in\mathcal{N}} Re(\lambda_i)$, the eigenvalue with smallest real part. Note that (3.47) and (3.22) employ the same object $\underline{\lambda}_R = min_{i\in\mathcal{N}} \mathrm{Re}(\lambda_i) = \mathrm{Re}(\lambda_2)$.

This theorem highlights an important factor about controls design for multi-agent systems on communication graphs. Namely, the communication graph topology must be taken into account in the design of local controllers. To wit, consider the local agent dynamics (3.39) with the local controls $u_i = -Kx_i$. Then, the closed-loop local dynamics are $\dot{x}_i = (A - BK)u_i$. These dynamics are stabilized by the gain computed using the ARE design (3.45), (3.46). However, if these dynamics are now linked together using a communication graph, the interconnected closed-loop dynamics are (3.41), which are influenced by the neighbors states $x_j,\ j \in N_i$. These interconnected dynamics can be unstable if the condition (3.47) does not hold.

Therefore, in multi-agent systems, the local controls design must take into account the graph topology up front, or stability cannot be guaranteed. The graph topology is captured here in terms of $\underline{\lambda}_R = min_{i \in \mathcal{N}} Re(\lambda_i)$, the eigenvalue with smallest real part.

3.4 Time-varying Edge Weights and Switched Graphs

The results in this chapter so far are for the case of graphs with fixed constant edge weights a_{ij}. In many applications, however, the edge weights are time varying, such as when communication links fade and the channel gains a_{ij} change, or when links are temporarily lost so that $a_{ij} = 0$. Dynamically changing interaction topologies have been studied in [21],[22],[24],[37]. The following results allow the extension of Theorems 3.1 and 3.2 to the case of time-varying edge weights and switched graphs.

Examine Figure 3.6, which shows three directed communication graphs, none of which is strongly connected or has a spanning tree. Their union, however, is strongly connected. If the nodes represent dynamical agents, the graphs could represent the communication topology at three subsequent times. Though at no time are all the agents strongly connected, over the complete time period all nodes will eventually communicate with each other.

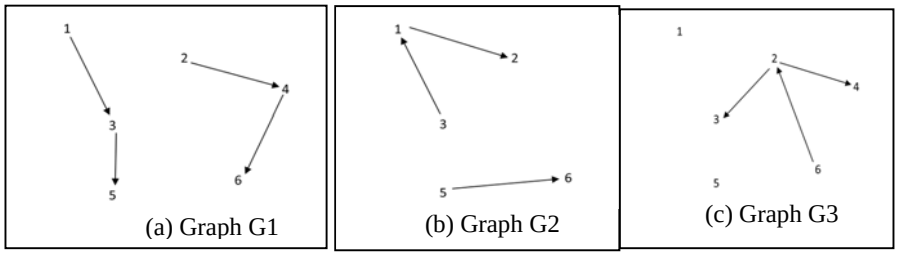


Figure 3.6. Three disconnected graphs whose union is strongly connected.

Consider a finite set of digraphs $\{G_p,\ p = 1, P\}$. The set is said to be jointly strongly connected if the union of its members is strongly connected. The set is said to jointly have a spanning tree if the union of its members has a spanning tree.

Consider the dynamics (3.13) with distributed protocols (3.14). We wish to allow the graph interactions to vary with time. Consider a finite set of possible

graph topologies $\{G_p : p = 1, \cdots, P\}$ with G_p having adjacency matrix A_p, in-degree matrix D_p, and Laplacian L_p. Let

$$\dot{x} = -L_\sigma x \tag{3.48}$$

with $\sigma(t) \in \{1, 2, \cdots, P\}$ a switching signal whose value at time *t* indicates which graph describes the current topology at time *t*. Let $t_0, t_1, \cdots$ be the times at which $\sigma(t)$ switches and let $t_{i+1} - t_i > \tau$ for some fixed dwell time $\tau > 0$.

The next result from [37] shows when the distributed local protocols (3.14) deliver consensus on switched graph topologies. In connection with this proof, the importance of the setup in here is that the Laplacians are constant between switchings, and $t_j - t_{j-1} > \tau$ for some fixed dwell time $\tau > 0$ so that the number of graphs in any bounded time interval is finite.

Theorem 3.4. **Continuous-Time Consensus on Changing Graph Topologies.** *Consider the dynamics (3.13) and protocols (3.14) on switched graphs. Let the graphs switch at times* $t_0, t_1, \cdots$. *Assume there exists an infinite sequence of contiguous, nonempty, bounded time intervals* $[t_{i_j}, t_{i_{j+1}}),\ j = 1, 2, \cdots$ *starting at* $t_{i_1} = t_0 = 0$, *such that the union of graphs over each interval has a spanning tree. Then consensus is reached and*

$$x_{ss} = \underline{1}c \tag{3.49}$$

for some consensus value c.

This theorem states that if there exists an infinite sequence of contiguous, nonempty, bounded time intervals over each of which the set of graphs jointly has a spanning tree, then consensus is reached. Note however that the consensus value c in Theorem 3.2 is not the same as (3.21) and depends on the switching times and individual graph topologies at each time instant.

Results similar to this hold for cooperative tracking Theorem 3.2, and for the case when the graph weights are time-varying between switching times [37].

References

[1] F.L. Lewis, H. Zhang, K. Hengster-Movric, A. Das, *Cooperative Control of Multi-Agent Systems: Optimal and Adaptive Design Approaches*, Springer-Verlag, Berlin, 2014.

[2] M. Defoort, T. Floquet, A. Kokosy, and W. Perruquetti, "Sliding-mode formation control for cooperative autonomous mobile robots," *IEEE Trans. Ind. Electron.*, vol. 55, no. 11, pp. 3944–3953, Nov. 2008.

[3] D. Cruz, J. McClintock, B. Perteet, O. Orqueda, Y. Cao, and R. Fierro, "Decentralized cooperative control - a multivehicle platform for research in networked embedded systems," *IEEE Control Syst. Mag*., vol. 27, no. 3, pp. 58–78, Jun. 2007.

[4] X. Wang, V. Yadav, and S. Balakrishnan, "Cooperative UAV formation flying with obstacle/collision avoidance," *IEEE Trans. Contr. Syst. Technol*., vol. 15, no. 4, pp. 672–679, Jul. 2007.

[5] J. Chen, X. Cao, P. Cheng, Y. Xiao, and Y. Sun, "Distributed collaborative control for industrial automation with wireless sensor and actuator networks," *IEEE Trans. Ind. Electron*., vol. 57, no. 12, pp. 4219–4230, Dec. 2010.

[6] R. Murray, "Recent research in cooperative control of multivehicle systems," *J. Dyn. Syst., Meas., Control*, vol. 129, no. 5, pp. 571–583, Sep. 2007.

[7] Kauffman, S. A., "Metabolic stability and epigenesis in randomly constructed genetic nets," *J. Theor. Biol.* **22,** 437–467 (1969).

[8] Y. Kuramoto, *Chemical Oscillations, Waves, and Turbulence*, Springer, Berlin, 1984.

[9] Strogatz, S. H. & Stewart, I. "Coupled oscillators and biological synchronization,"*Sci. Am.* **269**(6), 102–109 (1993).

[10] S.H. Strogatz, "From Kuramoto to Crawford: exploring the onset of synchronization in populations of coupled oscillators," Physica D, vol. 143, pp. 1-20, 2000.

[11] I.D. Couzin, J. Krause, N.R. Franks, S.A. Levin, "Effective leadership and decision-making in animal groups on the move," Nature, vol. 533, pp. 513-516, Feb. 2005.

[12] C.W. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model," Proc. SIGGRAPH 87, Computer Graphics 21(4), pages 25-34. edited by Maureen C. Stone, 1987.

[13] W. Wang and J. Slotine, "A theoretical study of different leader roles in networks," *IEEE Trans. Autom. Control*, vol. 51, no. 7, pp. 1156–1161, Jul. 2006.

[14] N.E. Leonard and E. Fiorelli, "Virtual leaders, artificial potentials, and coordinated control of groups," Proc. IEEE Conf. Decision and Control, pp. 2968-2973, Dec. 2001.

[15] Gazi V., Passino K.M., "Stability Analysis of Social Foraging Swarms," IEEE Trans. on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 34, No. 1. Feb. pp. 539-557, 2004.

[16] T. Vicsek, A. Czir´ok, E. Ben-Jacob, I. Cohen, and O. Shochet, “Novel type of phase transition in a system of self-driven particles,” *Phys. Rev. Lett.*, vol. 75, no. 6, pp. 1226–1229, 1995.

[17] N. Chopra and M.W. Spong, “On synchronization of Kuramoto Oscillators,” Proc. IEEE CDC, 2005.

[18] F. Dorfler and F. Bullo, “Transient stability analysis in power networks and synchronization of non-uniform Kuramoto oscillators,” Proc. *American Control Conference*, Baltimore, MD, pp. 930-937, June 2010.

[19] R. Ortega, M. Galaz, A. Astolfi, Y. Sun, “Transient stabilization of multimachine power systems with nontrivial transfer conductances,” IEEE Transactions on Automatic control, vol. 50, no. 1, pp. 60-75, Jan. 2005.

[20] D. Bertsekas and J.N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Athena Scientific, Belmont, MA, 1997.

[21] A. Jadbabaie, J. Lin, and A. Morse, “Coordination of groups of mobile autonomous agents using nearest neighbor rules,*” IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 988–1001, Jun. 2003.

[22] R. Olfati-Saber and R. Murray, “Consensus problems in networks of agents with switching topology and time-delays,” *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

[23] J. Fax and R. Murray, “Information flow and cooperative control of vehicle formations,*” IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1465–1476, Sep. 2004.

[24] W. Ren and R. Beard, “Consensus seeking in multiagent systems under dynamically changing interaction topologies,” *IEEE Trans. Autom. Control*, vol. 50, no. 5, pp. 655–661, May 2005.

[25] L. Moreau, “Stability of multiagent systems with time-dependent communication links,*” IEEE Trans. Autom. Control*, vol. 50, no. 2, pp. 169– 182, Feb. 2005.

[26] W. Ren, R. Beard, and E. Atkins,, “A survey of consensus problems in multiagent coordination,” in *Proc. Amer. Control Conf.*, Portland, OR, pp. 1859–1864, 2005.

[27] J. Han, M. Li, and L. Guo, “Soft control on collective behavior of a group of autonomous agents by a shill agent,*” J. System Science and Complexity*, vol. 19, pp.54-62, 2006.

[28] W. Ren, R. Beard, and E. Atkins, “Information consensus in multivehicle cooperative control,*” IEEE Control Syst. Mag.*, vol. 27, no. 2, pp. 71–82, Apr. 2007.

[29] W. Ren, K. Moore, and Y. Chen, “High-order and model reference consensus algorithms in cooperative control of multivehicle systems,*” J. Dyn. Syst., Meas., Control*, vol. 129, no. 5, pp. 678–688, 2007.

[30] R. Olfati-Saber, J. Fax, and R. Murray, “Consensus and cooperation in networked multi-agent systems,*” Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[31] W. Zhu and D. Cheng, “Leader-following consensus of second-order agents with multiple time-varying delays,” *Automatica*, vol. 46, no. 12, pp. 1994–1999, 2010.

[32] W. Ren and C. Cao, *Distributed Coordination of Multi-agent Networks: Emergent Problems*, Models, and Issues . London: Springer-Verlag, 2011.
[33] R. Diestel, *Graph Theory*, Springer-Verlag, New York,2000.
[34] C. Godsil and G.F. Royle, *Algebraic Graph Theory*, Springter-Verlag, New York, 2001.
[35] J. Brewer, "Kronecker products and matrix calculus in system theory," IEEE Trans. Circuits and Systems, vol. 25, no. 9, pp. 772-781.
[36] H. Zhang, F.L. Lewis, and Abhijit Das, "Optimal design for synchronization of cooperative systems: state feedback, observer and output feedback," IEEE Trans. Automatic Control, vol. 56, no. 8, pp. 1948-1952, August 2011.
[37] W. Ren and R.W. Beard, *Distributed Consensus in Multi-Vehicle Cooperative Control*, Springer-Verlag, London, 2008.