Minimum Control Latency of Dynamic Networks

Weiguo Dai*, Zhaoquan Gu*, Xiao Lin*, Qiang-Sheng Hua†, and Francis C.M.Lau‡
*Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China
†School of Computer Science & Technology, Huazhong University of Science and Technology, China
‡Department of Computer Science, The University of Hong Kong, Hong Kong, China

***Abstract*—Controlling a dynamic network is interesting and important in practical applications, which is to drive the network from any initial state to any desired state. Much research has been conducted in revealing the controllability and seeking the underlying correlations of the network. However, no existing works have considered the time needed to control the network, which we refer to as *control latency*. In this paper, we initiate the study of control latency of dynamic networks. First of all, we formulate the minimum control latency (MCL) problem for designing the controlling pattern with minimum number of controllers. We show that the MCL problem is NP-hard by reducing the multiprocessor scheduling problem to it. Then, we propose a greedy algorithm for designing a controlling pattern that can control the network within two times the minimum control latency. Moreover, when the control latency is bounded by a given value, we propose another constant approximation algorithm to design a controlling pattern which uses at most three times the minimum number of controllers. We conduct extensive simulations on both synthetic and real networks to corroborate our theoretic analysis.**

***Index Terms*—Structural controllability, Minimum control latency, Controlling pattern design**

I. Introduction

Dynamic networks exist in a wide-range of application scenarios involving systems such as natural, social and industrial systems [13], [16], [24], where the network state may be updated through the system's own influence. For example, a social network can be modeled as a dynamic system [15], [16] where the state of each person (i.e. the sentiment towards a particular topic) is influenced by its neighbors and updated from time to time. The public traffic network [1], [10] is another example where the load of each road is affected by the other nearby roads and needs to be updated dynamically. In recent years, how to control dynamic networks has attracted researchers' attention due to its important academic and practical significance; essentially, the problem is how to drive the network from any initial state to any desired state by imposing some appropriate controllers [8], [11], [25].

Most real networks are driven by nonlinear processes, but it happens that the controllability of nonlinear networks is similar to that of linear networks in many aspects [11], [21]. Therefore, we adopt the linear time-invariant model [19], [20], [23] to describe dynamic networks in this paper:

$$\overrightarrow{x(t+1)} = A \cdot \overrightarrow{x(t)} + B \cdot \overrightarrow{u(t)} \tag{1}$$

where $\overrightarrow{x(t)} \in R^N$ are the state values of the N nodes in the network at time t, $\overrightarrow{u(t)} \in R^M$ are the input signals of the M controllers, $A \in R^{N \times N}$ is the transmission matrix that reveals the influence relation in the network, and $B \in R^{N \times M}$ is the controlling matrix which shows the controlling pattern. It is proved in [5] that the network is *controllable* if and only if the matrix

$$C = (B, AB, A^2B, \ldots, A^{N-1}B)$$

has full rank. In practical networks, it is difficult to obtain the exact value of the influence, and so we focus on *structural controllability* where we are only aware of the non-zero and zero entries of the transmission matrix [9], [11].

There are many outstanding results on the structural controllability of dynamic networks [9], [11], [14], [18]. The property of structural controllability is reduced to the property of a graph based on the matrices (A, B) [9]. A maximum matching based algorithm is proposed in [11] to compute the minimum number of controllers. The dynamic process defined on the edges of a network is studied in [14], and it is different from simple nodal dynamics. The nature of the underlying correlations that affect the minimum number of controllers is also studied in [18]. However, to our best knowledge, no existing works have considered the control latency (i.e. the minimum time needed to control the network), which is important in many practical applications. For example, a controversial topic is spreading in a social network and one would like to know how to incite all people to have a sentiment about it, positive or negative, as quickly as possible.

In this paper, we propose the minimum control latency (MCL) problem, which is to design a controlling pattern that can drive the network from an initial state to a desired state in a sufficiently short time. The main challenge of the problem is to minimize the control latency with the minimum number of controllers. Without this constraint, we can impose N controllers to control each node separately, but it is costly and hard to implement in practice. Moreover, some applications may need the control latency to be bounded by a given value, such as the public traffic network which should be driven to a less crowded state as quickly as possible. We also study the problem of designing a suitable controlling pattern under some bounded control latency.

The main contributions of this paper are:

1) We formulate the minimum control latency (MCL) problem and prove that it is NP-hard.
2) A greedy algorithm is proposed which controls the network within *two* times the latency of the optimal solution.
3) When the control latency is bounded, an approximation algorithm is introduced which uses at most *three* times the

978-1-4799-8381-0/15/$31.00 ©2015 IEEE

number of controllers of the optimal solution to control the network.

4) We have conducted extensive simulations to evaluate our algorithms on both synthetic networks and real networks, and the results clearly show the efficiency of our algorithms.

The remainder of the paper is organized as follows. We give the system model and problem formulation in the next section. Section III highlights the related works and introduces some important graph structures in structural controllability. The control properties are shown in IV and we derive the control latency of the graph structures. We show the NP-hardness of the MCL problem in Section V and give a constant approximation algorithm. Moreover, we propose another constant approximation algorithm to control the network under bounded latency in Section VI. Simulation results are presented in Section VII, and we conclude the paper in Section VIII.

II. Preliminaries

A. System Model

We consider a dynamic directed network $G(A)$ which consists of N nodes $V = \{v_1, v_2, \ldots, v_N\}$. Denote the column vector $\overrightarrow{x(t)} \in R^N$ as the values (states) of the nodes at time t and $A \in R^{N\times N}$ as the transmission matrix that reveals the influence values between the nodes. If there is a directed edge from node v_j to node v_i, the influence value $a_{ij} \neq 0$. It is clear that A is the transpose of the network's adjacent matrix if all influence values equal 1.

In this paper, we focus on controlling the dynamic network by imposing M input signals (the controllers) $C = \{c_1, c_2, \ldots, c_M\}$ on the network, and we call $G(A, B)$ the controlled network, where $B \in R^{N\times M}$ is the controlling matrix that reveals the connection pattern of the input signals going into the network. Suppose that time is divided into slots of equal length, Δt (time units), and the input signal can change at the beginning of each slot. We assume the network $G(A, B)$ exhibits a good fit to a linear time-invariant model of the form as in [11], [19], [23] in a discrete-time way, and it is formulated as:

$$\overrightarrow{x(t+1)} = A \cdot \overrightarrow{x(t)} + B \cdot \overrightarrow{u(t)}$$

where column vector $u(t) \in R^M$ is the time-variant input signal at time t.

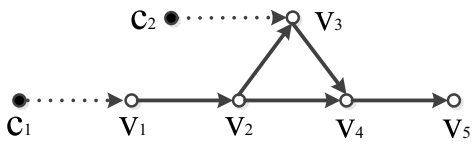


Fig. 1. An example of the controlled network

As depicted in Fig. 1, there are 5 nodes in the network $\{v_1, v_2, v_3, v_4, v_5\}$ and the matrix A can be expressed as:

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ a_{21} & 0 & 0 & 0 & 0 \\ 0 & a_{32} & 0 & 0 & 0 \\ 0 & a_{42} & a_{43} & 0 & 0 \\ 0 & 0 & 0 & a_{54} & 0 \end{pmatrix}$$

where $a_{21}, a_{32}, a_{42}, a_{43}, a_{54} \neq 0$ and they represent the influence strength between two nodes. Two controllers c_1, c_2 are imposed to control the network with two directed edges (dotted edges in the figure) and the controlling matrix B can be expressed as:

$$B = \begin{pmatrix} b_{11} & 0 \\ 0 & 0 \\ 0 & b_{32} \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

where $b_{11}, b_{32} \neq 0$ represent the controlling strength.

B. Structural Controllability

We call a pair of transmission matrix A and controlling matrix B *controllable* if the controlled network can be driven from any initial state $\overrightarrow{x_0}$ to a desired state $\overrightarrow{x_d}$ in finite time by imposing time-dependent input signals. It is proved in [5] that (A, B) is controllable if and only if the $N \times NM$ matrix

$$C = (B, AB, A^2B, \ldots, A^{N-1}B)$$

has full rank, i.e. $rank(C) = N$. (This is called Kalman's controllability rank condition [8].)

However, it is difficult to obtain the exact influence values in practical dynamic networks and the rank condition cannot be computed correctly when the matrices A, B are not fixed. Thus, *structural controllability* is more interesting which does not need the specific values of A and B.

Definition 2.1: Two pairs of matrices $(A, B), (A', B')$ have the same *structure* if for every zero entry of matrices (A, B), the corresponding entry of (A', B') is zero and for every zero entry of matrices (A', B'), the corresponding entry of (A, B) is also zero.

For example, the following matrices (A, B) and (A', B') have the same structure.

$$A = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \quad B = \begin{pmatrix} 2 & 0 \\ 1 & 3 \\ 0 & 1 \end{pmatrix}$$

$$A' = \begin{pmatrix} 0 & 0 & 3 \\ 0 & 0 & 0 \\ 0 & 2 & 0 \end{pmatrix} \quad B' = \begin{pmatrix} 1 & 0 \\ 1 & 2 \\ 0 & 3 \end{pmatrix}$$

Structural controllability is proposed in [9] and it is useful in exploring the controllability of practical dynamic networks since the structure of the networks can be easily analyzed and the matrices are composed of zero and non-zero elements.

Definition 2.2: A pair of matrices (A, B) is *structurally controllable* if and only if $\forall \epsilon > 0$, there exists a controllable

pair (A', B') of the same structure as (A, B) such that $||A' - A|| < \epsilon$ and $||B' - B|| < \epsilon$.

Structural controllability helps overcome the incomplete knowledge of the influence values between nodes. Obviously, a controllable pair of matrices (A, B) is also structurally controllable, but the reverse may not be true. It is shown in [9], [22] that a structurally controllable system can be controllable for almost all combinations of elements, except for the pathological situations where the matrices A, B satisfy certain accidental constraints. In this paper, we focus on the structural controllability of dynamic networks.

C. Problem Formulation

Since no existing works considered the time needed to control the network, we define control latency as follows:

Definition 2.3: Considering a controlled network with matrices (A, B) satisfying Eqn. (1), for any initial state $\overrightarrow{x_0} \in R^N$ and any desired state $\overrightarrow{x_d} \in R^N$, control latency T is the number of time slots needed to drive the network state from $\overrightarrow{x_0}$ to $\overrightarrow{x_d}$ with time-dependent input signals $\overrightarrow{u} \in R^M$.

In this paper, we minimize the control latency with the constraint that only the minimum number of controllers could be used, for two reasons. First, without the constraint, we can impose N controllers on each node separately. However, it is costly to control all nodes in real networks. Second, a maximum matching based method is proposed in [11] to find the minimum number of controllers, but different patterns exist in controlling the network (i.e. B has many combinations). Thus, we formulate the Minimum Control Latency (**MCL**) problem as:

Problem 1: For any dynamic network $G(A)$, find the controlling matrix B that minimizes the control latency when the number of controllers is also minimized.

III. Related Works

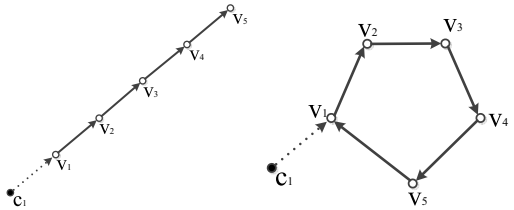
According to control theory, a linear dynamic network is controllable if the network can be driven from any initial state to any desired state with some appropriate input signals [5], [8], [12]. Structural controllability is more significant since the exact influence values cannot be obtained in some real networks. Several graph structures are proposed in [9] to transform structural controllability into graph based properties.

To begin with, there are two important structures in describing structural controllability. The first one is *elementary path*, which is a sequence of directed edges $\{(v_1 \rightarrow v_2), (v_2 \rightarrow v_3), \ldots, (v_{k-1} \rightarrow v_k)\}$ where nodes $\{v_1, v_2, \ldots, v_k\}$ are distinct. The other one is *elementary cycle* when a directed edge $(v_k \rightarrow v_1)$ exists in the above elementary path.

A special graph structure called *cactus* is introduced in [9] to analyze the structural controllability, which is constructed iteratively based on *stem* and *bud*. We introduce these three notions briefly.

Definition 3.1: (Stem) A stem is an elementary path where the first vertex is a controller.

Definition 3.2: (Bud) A bud is an elementary cycle in V with a directed edge that ends in a vertex of the cycle.



(a) stem (b) bud

Fig. 2. Examples of a stem and a bud

For example, Fig. 2(a) is a stem with 5 nodes in V and one controller c_1. Fig. 2(b) shows a bud which consists of an elementary cycle and one directed link $(c_1 \rightarrow v_1)$. Note that the first vertex of the bud does not need to be a controller and it could be any vertex in the node set V.

Definition 3.3: (Cactus) A cactus is constructed recursively as a sequence $\{S_0, S_1, \ldots, S_k\}$. A stem is a cactus and denote it as S_0. Suppose there are k buds $B_1, B_2, \ldots, B_k$; then for any $1 \leq i \leq k$, $S_i = S_{i-1} \cup B_i$ and the first vertex of B_i is also the only vertex that belongs to S_{i-1}.

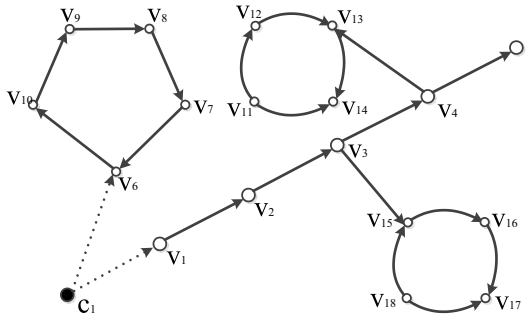


Fig. 3. An example of a cactus

Fig. 3 depicts an example of cactus which is composed of a stem $S_0 = \{c_1 \rightarrow v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_4 \rightarrow v_5\}$ and three buds.

Theorem 1: A linear dynamic network is structurally controllable if and only if there exists a vertex disjoint union of cactus that spans the graph $G(A, B)$.

The theorem is given in [2] and it reveals the equivalence of structural controllability and the constructed cactus. However, we are aware of only the transmission matrix A in real networks, with no knowledge of what the controlling matrix B looks like, and what is the minimum number of controllers (i.e. M).

A breakthrough result was achieved in [11] where a polynomial time algorithm was proposed to find the minimum number of controllers based on the maximum matching method. The basic idea is to find the maximum matching in the dynamic network and construct the cactus to span the network. Stimulated by the work, a great deal of controllability related research has since been conducted. For example, the dynamical process on the edges of a network was proposed in [14] and it revealed different controllability properties from simple

nodal dynamics. The nodes were classified in [7] according to the roles they took part in the control configuration. A new control profile to classify the nodes according to the underlying correlations of the networks was proposed in [20]. However, to the best of our knowledge, no existing works considered the control latency issue, which is what we try to initiate through this paper.

IV. Control Latency of Graph Structures

In this section, we first introduce two important lemmas about controlling the elementary paths and cycles. Then, we derive the control latency of the related graph structures.

A. Control Properties

Lemma 4.1: One controller can control only one elementary path.

Proof: To begin with, we prove that one controller can control one elementary path, which shows the importance of the stem structure. Suppose there is an elementary path $\{v_1 \to v_2 \to \ldots \to v_n\}$ and one controller c_1 is connected to the first vertex v_1. The matrices A, B can be expressed as:

$$A = \begin{pmatrix} 0 & 0 & \ldots & 0 \\ a_{21} & 0 & \ldots & 0 \\ 0 & a_{32} & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & a_{n-1,n} & 0 \end{pmatrix} \quad B = \begin{pmatrix} b_{11} \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2}$$

It is easy to compute $AB = (0, b_{11} \cdot a_{21}, 0, \cdots, 0\}^T$ and $A^k B = (0, \cdots, 0, b_{11} \cdot \prod_{i=1}^{k} a_{k+1,k}, 0, \cdots, 0)^T$ where the $(k+1)$-th element is non-zero. Thus the matrix $C = (A, AB, \ldots, A^{n-1}B)$ is a diagonal matrix with full rank and it is controllable.

For any two elementary paths $\{v_1 \to v_2 \to \ldots \to v_{n_1}\}$ and $\{v'_1 \to v'_2 \to \ldots \to v'_{n_2}\}$, we show that one controller cannot control them simultaneously. For simplicity, we assume the controller is connected to both v_1, v'_1 (the first vertex of the two paths). Without loss of generality, suppose $n_1 > n_2$ and the matrices can be expressed as:

$$A = \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} \quad B = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}$$

where A_1, A_2 have a similar structure as Eqn. (2) and $B_1 = \{b_1, 0, \ldots, 0\}$, $B_2 = \{b_2, 0, \ldots, 0\}$ where $b_1, b_2 \neq 0$. Then

$$C = \begin{pmatrix} B_1 & A_1B_1 & \ldots & \ldots & A_1^{n_1-1}B_1 & 0 \ldots 0 \\ B_2 & A_2\ B_2 & \ldots & A_2^{n_2-1}B_2 & 0 \ldots & 0 \ldots 0 \end{pmatrix}$$

and it does not have full rank. Actually, no matter what the controlling matrix $B \in R^{n \times 1}$ looks like, $A^k B = \overrightarrow{0}$ when $k \geq n$. Thus the last n_2 columns of C are all zeros and it is not structurally controllable. Hence, the lemma holds. ■

Corollary 1: k elementary paths can be controlled by k controllers, where each controller is connected to the first vertex of each path.

The corollary can be verified similarly by checking the matrix C's rank and we omit the details.

Lemma 4.2: One controller can control many elementary cycles (along with one elementary path).

Proof: First, we show that one controller can control one elementary cycle, which is the constructed bud. Supposing there is an elementary cycle $\{v_1 \to v_2 \to \ldots \to v_n \to v_1\}$ and one controller c_1 is connected to v_k, then

$$A = \begin{pmatrix} 0 & 0 & \ldots & a_{1n} \\ a_{21} & 0 & \ldots & 0 \\ 0 & a_{32} & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & a_{n-1,n} & 0 \end{pmatrix} \quad B = \begin{pmatrix} 0 \\ \vdots \\ b_{k1} \\ \vdots \\ 0 \end{pmatrix} \tag{3}$$

It is easy to compute $AB = (0, 0, \cdots, b_{k1} \cdot a_{k+1,k}, \cdots 0)^T$ where the $(k+1)$-th column is non-zero. Similarly, $A^2B = \{(0, 0, \cdots, b_{k1} \cdot a_{k+1,k} \cdot a_{k+2,k+1}, \cdots 0)^T$ and the $(k+2)$-th column is non-zero. Through such recursive computing, A^iB has only one non-zero entry at the $((k+i-1)\%n+1)$-th column and the matrix $C = (B, AB, A^2B, \ldots, A^{n-1}B)$ has full rank, which implies that it is structurally controllable.

Supposing a stem and several elementary cycles are controlled by a single controller (denote the matrices as (A_1, B_1)), we show that it is also structurally controllable when a directed edge is added from the controller to a new elementary cycle. Denote the matrices as:

$$A = \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} \quad B = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix} \tag{4}$$

where A_2 has the same structure as in Eqn. (3) and $B_2 = (b_2, 0, \ldots, 0)^T$ means the controller is connected to the first vertex of the cycle. As shown above, (A_2, B_2) is structurally controllable. We use another principle to prove the lemma. As shown in [17], a pair of matrices (A, B) is controllable if and only if $\forall \overrightarrow{x}, \alpha,\ \overrightarrow{x}A = \alpha\overrightarrow{x}$ implies $\overrightarrow{x}B \neq 0$, where $\overrightarrow{x} \neq \overrightarrow{0}$ is a vector of complex entries and α is a complex number.

Since (A_1, B_1) is structurally controllable, there exists a controllable pair (A'_1, B'_1) of the same structure as (A_1, B_1) and we can modify the entries of A_2 such that A'_1 and A_2 have no common eigenvalue. Supposing (A, B) in Eqn. (4) is not controllable, there exist $\overrightarrow{x_1}, \overrightarrow{x_2}$ that are not both equal to $\overrightarrow{0}$ and a complex number α such that:

$$(\overrightarrow{x_1} \quad \overrightarrow{x_2}) \begin{pmatrix} A'_1 & 0 \\ 0 & A_2 \end{pmatrix} = \alpha(\overrightarrow{x_1} \quad \overrightarrow{x_2})$$

and

$$(\overrightarrow{x_1} \quad \overrightarrow{x_2}) \begin{pmatrix} B'_1 \\ B_2 \end{pmatrix} = \overrightarrow{0}$$

That is, $\overrightarrow{x_1}A'_1 = \alpha\overrightarrow{x_1}$, $\overrightarrow{x_2}A_2 = \alpha\overrightarrow{x_2}$, and $\overrightarrow{x_1}B'_1 + \overrightarrow{x_2}B_2 = \overrightarrow{0}$. Since A'_1 and A_2 have no common eigenvalue, $\overrightarrow{x_1} = \overrightarrow{0}$ or $\overrightarrow{x_2} = \overrightarrow{0}$ (otherwise, α is their common eigenvalue, a contradiction).

If $\overrightarrow{x_1} = \overrightarrow{0}$ and $\overrightarrow{x_2} \neq \overrightarrow{0}$, $\overrightarrow{x_2}A_2 = \alpha\overrightarrow{x_2}$ but $\overrightarrow{x_2}B_2 = \overrightarrow{0}$, which implies (A_2, B_2) is not controllable and it is a contradiction. If $\overrightarrow{x_1} \neq \overrightarrow{0}$ and $\overrightarrow{x_2} = \overrightarrow{0}$, (A'_1, B'_1) is not controllable, which is also a contradiction. Thus the matrices (A, B) in Eqn. (4) are structurally controllable.

Combining these two aspects, one controller can control many elementary cycles along with one elementary path. ■

From Lemma 4.1 and Lemma 4.2, the cactus can be controlled by a single controller and the minimum number of controllers is thus the number of vertex disjoint cactus.

B. Control Latency

Since no existing works have considered the needed number of time slots in driving any initial state to a desired state, we pioneer to analyze the control latency of the constructed structures.

Lemma 4.3: The control latency of a stem $\{c_1 \rightarrow v_1 \rightarrow v_2 \rightarrow \ldots \rightarrow v_n\}$ is n time slots.

Proof: Suppose the controller c_1 drives the initial state $\overrightarrow{x_0} \in R^n$ to a desired state $\overrightarrow{x_d} \in R^n$ in T time slots; thus:

$$\begin{aligned}\overrightarrow{x(1)} &= A \cdot \overrightarrow{x(0)} + B \cdot u(1)\\ \overrightarrow{x(2)} &= A \cdot \overrightarrow{x(1)} + B \cdot u(2)\\ &\vdots\\ \overrightarrow{x_d} = \overrightarrow{x(T)} &= A \cdot \overrightarrow{x(T-1)} + B \cdot u(T)\end{aligned}$$

Combine these equations to get

$$\overrightarrow{x_d} - A^T\overrightarrow{x_0} = Bu(1) + ABu(2) + \ldots + A^{T-1}Bu(T)$$

From the analysis of Lemma 4.1, A^kB has only one non-zero entry at the $(k+1)$-th row and when $k \geq n$, $A^kB = \overrightarrow{0}$. Since $\overrightarrow{x_d} \in R^n$, the needed time $T \geq n$ and the input signal $u(k)$ can be constructed correspondingly (u_k is constructed as dividing the k-th row of $\overrightarrow{x_d} - A^T\overrightarrow{x_0}$ by the non-zero entry of $A^{k-1}B$). Therefore, the control latency is equal to the number of nodes in the elementary path n. ■

Lemma 4.3 has an intuitive explanation: as the controller is connected to the first vertex of the elementary path, the input signal cannot reach the last vertex of the path in less than n time slots. Thus at least n time slots are needed to control the path. This approach is also applicable to the elementary cycle.

Corollary 2: The control latency of an elementary cycle is equal to the number of nodes in the cycle.

Since a cactus consists of a stem and many elementary cycles, we derive the control latency of a cactus when only one input signal is imposed.

Theorem 2: The control latency of a cactus is equal to the number of nodes controlled by the single controller.

Proof: Supposing the cactus has a controller and n nodes, it is obvious that the pair of matrices (A, B) is structurally controllable from Theorem 1 where $A \in R^{n\times n}$ and $B \in R^{n\times 1}$. Suppose that the controller can drive the network from initial state $\overrightarrow{x_0} \in R^n$ to a desired state $\overrightarrow{x_d} \in R^n$ in T time slots. Similar to Lemma 4.3, let $\overrightarrow{\Delta x} = \overrightarrow{x_d} - A^T\overrightarrow{x_0}$ and

$$\begin{aligned}\overrightarrow{\Delta x} &= Bu(1) + ABu(2) + \ldots + A^{T-1}Bu(T)\\ &= (B, AB, \ldots, A^{T-1}B)\begin{pmatrix} u(1)\\ u(2)\\ \vdots\\ u(T)\end{pmatrix}\end{aligned}$$

When $T < n$, the rank of matrix $C_T = (B, AB, \ldots, A^{T-1}B)$ is less than n, and there exists $\overrightarrow{\Delta x} \in R^N$ such that $u(1), u(2), \ldots, u(T)$ do not have a feasible solution. When $T = n$, $C_n = (B, AB, \ldots, A^{n-1}B)$ has full rank n since (A, B) is structurally controllable, and thus:

$$(u(1), u(2), \ldots, u(T))^T = C_n^{-1} \cdot \overrightarrow{\Delta x}$$

where C_n^{-1} is the inverse matrix of C_n. Therefore, the control latency is n, which concludes the theorem. ■

V. Minimum Control Latency Design

In this section, we prove that the MCL problem is NP-hard, and then we propose a constant approximation algorithm to control the network.

A. NP-Hardness

We show that the multiprocessor scheduling problem is reducible to the MCL problem in polynomial time. The multiprocessor scheduling problem: given a set of jobs $J = \{j_1, j_2, \ldots, j_n\}$ where each job j_i has length l_i (a positive integer) and a number of processors m, what is the minimum possible time required to schedule all jobs on these processors? This problem has been proved to be NP-hard [4].

Theorem 3: The MCL problem is NP-Hard.

Proof: For an instance of the multiprocessor scheduling problem, where there are m processors and the job set is $J = \{j_1, j_2, \ldots, j_n\}$ of which each job j_i has length l_i, we construct a corresponding directed graph in polynomial time. As illustrated in Fig. 4, construct m edges as $(v_{11} \rightarrow v_{12}), (v_{21} \rightarrow v_{22}), \ldots, (v_{m1} \rightarrow v_{m2})$, and construct n directed cycles $C_1, C_2, \ldots, C_n$, where the length of each cycle C_i is l_i.

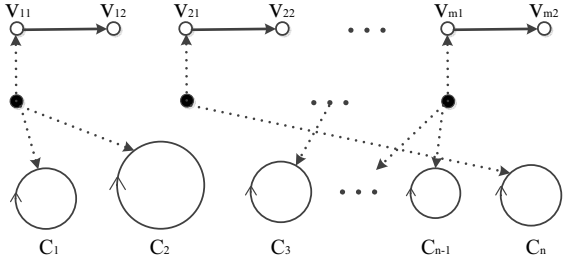


Fig. 4. An example of constructing a controlled network for a given instance of the multiprocessor scheduling problem

From Lemma 4.1, m controllers are needed to control m elementary paths and each controller is directly connected to the first vertex of the edge $(v_{i1} \rightarrow v_{i2})$. From Lemma 4.2, each controller can be connected to more than one cycle for the controllability, and thus they comprise a cactus. As shown in Theorem 2, the control latency for such a cactus is the number of controlled nodes and we need a suitable way to connect these cycles. We show that a controlling pattern minimizing the control latency exists if and only if a feasible solution of the multiprocessor scheduling problem exists.

Suppose an optimal solution of the multiprocessor scheduling problem exists and the maximum time to schedule these

jobs is T. If job j_i is scheduled to processor k, connect cycle C_i to the k-th controller. Then it is easy to check that the control latency of the network is $T+2$. If there is another controlling pattern with smaller control latency, a better schedule can be constructed: if cycle C_i is connected to the k-th controller, schedule job j_i to processor k and this should have a shorter scheduling time, which is a contradiction. Thus, the constructed controlling pattern has the smallest control latency. On the other side, an optimal solution of the MCL problem corresponds to an optimal schedule of these jobs. Therefore, the multiprocessor scheduling problem is reducible to the MCL problem, which concludes the theorem. ■

B. Constant Approximation Algorithm

We first present the maximum matching based algorithm that finds the elementary paths and elementary cycles.

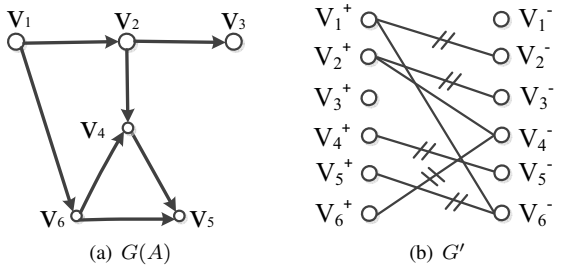
Algorithm 1 Elementary Paths and Cycles Construction

1: Invoke undirected graph construction to get graph G';
2: Apply the Hopcroft-Karp algorithm [6] on G' to compute the maximum matching M^* of G';
3: Mark all nodes in $G(A)$ as black;
4: **while** $\exists$ black node $v_i \in G(A)$ such that v_i^+ is in the maximum matching M^* **do**
5: Find edges $(v_i^+, v_{i+1}^-), (v_{i+1}^+, v_{i+2}^-), \ldots, (v_j^+, v_{j+1}^-) \in M^*$ until v_{j+1}^+ is not in M^* or $v_{j+1}^* = v_i^+$;
6: Choose edges in $G(A)$ as $v_i \to v_{i+1} \to \ldots \to v_{j+1}$;
7: Mark nodes $\{v_i, v_{i+1}, \ldots, v_j, v_{j+1}\}$ white;
8: **end while**
9: **for** each black node $v_i \in G(A)$ **do**
10: Regard v_i as an elementary path;
11: **end for**

Undirected Graph Construction

1: Construct an undirected graph G' based on $G(A)$
2: For every vertex $v \in G(A)$, construct v^+, v^- in G';
3: For every directed edge $(v_i \to v_j) \in G(A)$, construct an undirected edge (v_i^+, v_j^-) in G';

To apply the maximum matching algorithm for a bipartite graph [6], we construct an undirected graph $G' = (V^+ \cup V^-, E')$ based on the directed network $G(A) = (V, E)$. For each node $v_i \in V$, construct two nodes v_i^+, v_i^- where v_i^+ means an edge starting at node v_i while v_i^- represents an edge ending at it. Denote $V^+ = \{v_1^+, v_2^+, \ldots, v_N^+\}$ and $V^- = \{v_1^-, v_2^-, \ldots, v_N^-\}$. For each edge $v_i \to v_j \in E$, add edge (v_i^+, v_j^-) to E'. After the undirected graph construction, we apply the Hopcroft-Karp algorithm [6] to G' to compute the maximum matching M^* since G' is a bipartite graph. Then we find the elementary paths and cycles based on the matching M^* as in Lines 3-11. Fig. 5 shows an example where 6 nodes exist in the graph $G(A)$ as Fig. 5(a) and the undirected bipartite graph G' is constructed as Fig. 5(b). The Hopcroft-Karp algorithm finds the maximum matching as $M^* = \{(v_1^+, v_2^-), (v_2^+, v_3^-), (v_4^+, v_5^-), (v_5^+, v_6^-), (v_6^+, v_4^-)\}$; then we find an elementary path in $G(A)$ as $\{v_1 \to v_2 \to v_3\}$ and an elementary cycle $\{v_4 \to v_5 \to v_6 \to v_4\}$ as in Lines 3-8.



(a) $G(A)$ (b) G'

Fig. 5. An example of Alg. 1

Lemma 5.1: Alg. 1 finds the elementary paths and cycles in Line 6 and Line 10.

Proof: First of all, all nodes in $G(A)$ are marked black and Alg. 1 finds a black node $v_i \in G(A)$ such that v_i^+ is in the matching M^*. Then it finds picked edges in M^* in the order $(v_i^+, v_{i+1}^-), (v_{i+1}^+, v_{i+2}^-), \ldots, (v_j^+, v_{j+1}^-) \in M^*$ until v_{j+1}^+ is not in M^* or $v_{j+1}^* = v_i^+$. It is obvious that when $v_{j+1}^- \in M^*$, the constructed edges in $G(A)$ as in Line 6 form an elementary path, while it is an elementary cycle if v_{j+1} coincides with v_i. After the construction, the picked edges are marked white and this process proceeds until all nodes $v_i \in G(A)$ are marked white or no such node is in M^*. Then, the left black nodes that are not chosen in the paths or cycles compose an elementary path by itself in Line 10. Thus, Alg. 1 finds all elementary paths and cycles in Line 6 and Line 10. ■

After constructing the elementary paths and cycles, we present the approximation algorithm to control the network with minimum number of controllers. Denote these elementary paths as $\{P_1, P_2, \ldots, P_m\}$, the elementary cycles as $\{C_1, C_2, \cdots, C_n\}$, and let $|P_i|$ (or $|C_i|$) be the length of path P_i (or cycle C_i).

Algorithm 2 Greedy Controlling Pattern Design

1: Impose m controllers $\{c_1, c_2, \ldots, c_m\}$ and each controller c_i is connected to the first vertex of path P_i;
2: Order the elementary cycles by decreasing order of length as $\{C_1', C_2', \ldots, C_n'\}$;
3: Let $CL(i) := |P_i| + 1$, $\forall i \in [1, m]$;
4: **for** Each cycle C_i' **do**
5: Find $k \in [1, M]$ such that $CL(k) = \min_{j=1}^{m} CL(j)$;
6: Connect controller c_k to any vertex of cycle C_i';
7: Update $CL(k) = CL(k) + |C_i'|$;
8: **end for**

In Alg. 2, we impose m controllers on the networks and each controller is connected to an elementary path. Then $CL(i)$ is used to denote the control latency of controlling the i-th set (each set has only one elementary path initially). After ordering the elementary cycles by decreasing length, add the elementary cycle to the set with smallest control latency by connecting the controller to any vertex of the cycle. We show that our controlling pattern uses the minimum number

of controllers and the control latency is only two times the optimal value.

Theorem 4: Alg. 2 imposes the minimum number of controllers on the network and the control latency is two times the optimal value.

Proof: From Lemma 4.1, at least m controllers are needed to control the network. From the construction of elementary paths and cycles, the number of such paths is equal to the number of unmatched nodes in set V^+, and thus the number of controllers we use is minimum.

Let $p_{max} = \max_{i=1}^{m} |P_i|$ and $p_{min} = \min_{i=1}^{m} |P_i|$ be the maximum and minimum length of the elementary paths. Let $c_{max} = \max_{i=1}^{n} |C_i|$ be the maximum length of the elementary cycles and $CL(\delta) = \frac{1}{m}(\sum_{i=1}^{m}(|P_i|+1) + \sum_{i=1}^{n}|C_i|)$ be the average control latency. The lower bound of the control latency (optimal value) suits:

$$OPT \geq \max\{p_{max}+1, p_{min}+1+c_{max}, CL(\delta)\} \quad (5)$$

Denote the maximum latency of Alg. 2 as T. Supposing the maximum latency happens in the k-th set with $CL(k)$ and the last cycle added to the set is C_i', we claim that:

$$T \leq \max\{\frac{1}{m}[\sum_{j=1}^{m}(|P_j|+1) + \sum_{j=1}^{i-1}|C_j'|], p_{max}+1\} + |C_i'|$$

The first part is because $CL(k)$ is the minimum one among the m sets after cycles $C_1', C_2', \ldots, C_{i-1}'$ are added (before cycle C_i' is added to the k-th set). Therefore, $CL(k) \leq \max\{\frac{1}{m}[\sum_{j=1}^{m}(|P_j|+1) + \sum_{j=1}^{i-1}|C_j'|], p_{max}+1\}$. Then,

$$\begin{aligned} T &\leq \max\{\frac{1}{m}[\sum_{j=1}^{m}(|P_j|+1) + \sum_{j=1}^{i-1}|C_j'|], p_{max}+1\} + |C_i'| \\ &\leq \max\{CL(\delta), p_{max}+1\} + c_{max} \\ &\leq 2OPT \end{aligned}$$

thus the theorem is completed. ■

Supposing there are N nodes and L edges in the graph $G(A)$, Alg. 1 uses $O(N+L)$ time to construct the undirected graph and the time complexity of Alg. 2 is $O(L \log L + L^2)$. Due to the page limits, we omit the details. Therefore, the time complexity of our algorithm is $O(L(N+L))$.

VI. Controlling Pattern Design with Bounded Latency

In many practical situations, such as information propagation in a social network and transportation in a traffic network, the control latency needs to be capped under a fixed value; we show the method to design controlling pattern such that the network can be controlled with bounded latency.

Suppose the control latency of the network should be bounded by T^*. We invoke Alg. 1 to find the elementary paths and cycles. Similarly, suppose the elementary paths are $\{P_1, P_2, \ldots, P_m\}$ and the elementary cycles are $\{C_1, C_2, \cdots, C_n\}$, and let $|P_i|$ (or $|C_i|$) be the length of path P_i (or cycle C_i). In Alg. 3, we first tackle long cycles and paths that cannot be controlled in T^* time slots by splitting them into shorter paths. After that, we order all the shorter cycles by decreasing order as $\{C_1', C_2', \ldots, C_{n'}'\}$. For each cycle C_i', we first check if an elementary path exists such that they can be controlled with C_i' in T^* time slots. Otherwise, we find the uncontrolled cycles in increasing order of their length such that the control latency is bounded in T^* time slots.

Algorithm 3 Controlling Pattern with Bounded Latency

1: Invoke Alg. 1 to find the elementary paths and cycles;
2: **for** Cycle C_i with length $|C_i| > T^*$ **do**
3: Split the cycles into $\delta_i = \lceil \frac{|C_i'|}{T^*} \rceil$ paths where the first $\delta_i - 1$ paths have length $T^* - 1$;
4: **end for**
5: **for** Path P_i with length $|P_i| + 1 > T^*$ **do**
6: Split the cycles into $\delta_i = \lceil \frac{|P_i'|+1}{T^*} \rceil$ paths where the first $\delta_i - 1$ paths have length $T^* - 1$;
7: **end for**
8: Use controllers to control the paths with length $T^* - 1$;
9: Order the cycles with length no more than T^* by decreasing order of length as $\{C_1', C_2', \ldots, C_{n'}'\}$;
10: **for** Each cycle C_i' **do**
11: Find the minimum length of uncontrolled path as P_j;
12: **if** $|C_i'| + |P_j| + 1 \leq T^*$ **then**
13: Use a controller to connect P_j and C_i';
14: **else**
15: Find uncontrolled cycles of increasing order such that $|C_i'| + |C_1'| + \ldots + |C_k'| \leq T^*$;
16: Use a controller to connect these cycles;
17: **end if**
18: **end for**
19: For each uncontrolled path, connect it to a new controller;

Theorem 5: Alg. 3 can control the network in T^* time slots and the number of controllers used is at most three times the optimal solution.

Proof: It is obvious that the network can be controlled in T^* time slots based on the designed controlling pattern. Suppose there are m_1 elementary paths with length no greater than T^* (denoted as $\{P_{11}, P_{12}, \ldots, P_{1m_1}\}$) and m_2 paths longer than T^* (denoted as $\{P_{21}, P_{22}, \ldots, P_{2m_2}\}$). Similarly, suppose there are n_1 elementary cycles with length no greater than T^* (denote as $\{C_{11}, C_{12}, \ldots, C_{1n_1}\}$) and n_2 cycles longer than T^* (denoted as $\{C_{21}, C_{22}, \ldots, C_{2n_2}\}$).

The minimum number of controllers needed to control the m_1 paths and n_1 cycles suits:

$$OPT_1 \geq \max\{\frac{\sum_{i=1}^{m_1}(|P_{1i}|+1) + \sum_{i=1}^{n_1}|C_{1i}|}{T^*}, m_1\}$$

and the minimum number of controllers to control the m_2 paths and n_2 cycles suits:

$$OPT_2 \geq \frac{\sum_{i=1}^{m_2}(|P_{2i}|+1) + \sum_{i=1}^{n_2}|C_{2i}|}{T^*}$$

thus, the optimal value $OPT = OPT_1 + OPT_2$. Denote the number of controllers in Alg. 3 as N. From Lines 2-8, the

number of controllers is

$$N_2 = \sum_{i=1}^{m_2} \lceil \frac{|P_{2i}|+1}{T^*} \rceil + \sum_{i=1}^{n_2} \lceil \frac{|C_{2i}|}{T^*} \rceil \leq OPT_2$$

We bound the number of controllers to control m_1 paths and n_1 cycles, as follows. $N_{12} = m_1$ controllers are needed to control the elementary paths and each controller may also control one cycle satisfying $|C_i'|+|P_j|+1 \leq T^*$ (Line 12). Supposing there are N_{11} controllers to control the other cycles and the control latency under each controller suits: $\frac{T^*}{2} < CL(i) \leq T^*$ (at most one controller suits $CL(i) \leq \frac{T^*}{2}$, or otherwise, they can be controlled by one controller with bounded latency not greater than T^*). So $N_{11} \leq \frac{\sum_{i=1}^{n_1}|C_{1i}|}{T^*/2} \leq 2OPT_1$. Therefore,

$$N_1 \leq N_{11} + N_{12} \leq 3OPT_1$$

Combining all these, the controllers used in Alg. 3 have $N = N_1 + N_2 \leq OPT_2 + 3OPT_1 \leq 3OPT$. ■

Supposing there are N nodes and L edges in the graph $G(A)$, the time complexity of our algorithm is $O(L(N+L))$. Due to the page limits, we omit the details.

VII. Simulation

To evaluate our algorithms in controlling the dynamic networks, we conduct simulations on both synthetic and real networks. We choose the Erdös-Rényi (ER) model [3] to generate random networks as examples of synthetic network, and for real networks, we collect data from three real networks: transportation data of USA top 500 [1], UClonline [15], and a Facebook-like social network [16].

In the ER model, an edge between each pair of nodes exists with an equal probability p and obviously there are $\binom{N}{2}p$ expected edges, where N is the number of nodes in the graph. In our simulations, we fix $p = 0.01$ and $p = 0.005$ respectively and when N increases from 200 to 1000, Table I lists the results as the means of 1000 separate runs. From the table, our greedy algorithm has good performance compared to the lower bound in minimizing the control latency with minimum number of controllers. When N increases, fewer controllers are needed and the control latency increases. It is because when more edges are added to the graph, it is more likely to find long elementary paths or cycles. When $N = 1000, p = 0.01$, only one controller is needed, which implies only one elementary path or cycle exists after Alg. 1. When the probability that an edge exists is larger ($p = 0.01$), there are more edges in the network and the number of controllers is smaller (than $p = 0.005$), which incurs larger control latency.

In order to evaluate the controlling algorithm under bounded latency (Alg. 3), we fix $N = 500, p = 0.01$. Fig. 6 shows the comparison between our algorithm and the lower bound when the bound of the control latency increases from 10 to 200. As depicted, when the bound is smaller, more controllers are needed, which corroborates our analysis. Moreover, our algorithm is only slightly worse than the optimal solution.

We also evaluate our algorithms on real networks of which the transportation network [1] records the flight network in USA (500 nodes and 5960 edges), UClonline [15] is an online message network of students at UC, Irvine (1899 nodes and 20296 edges), and the Facebook-like [16] is an online social network similar to Facebook (899 nodes and 7089 edges). We compare the control latency of these three networks in Table II. It turns out that our algorithm can achieve the minimum control latency with minimum number of controllers for them.

In order to evaluate Alg. 3 under different control latency circumstances, we first bound the control latency from 2 to 16 for the transportation networks, and Fig. 7 shows that our algorithm is slightly worse than the lower bound. Moreover, the number of controllers decreases as the control latency increases, which also corroborates our analysis. When the control latency increases from 2 to 20 in the Facebook-like network, Fig. 8 shows the same phenomena. (In the UClonline network, we get similar results which we omit here.)

Through the simulation results, we can see that Alg. 2 has good performance on both synthetic and real networks, the control latency of which is comparable to the lower bound. When the control latency is bounded by a given value, the number of controllers used in Alg. 3 is also nearly optimal.

TABLE I
Control Latency Comparison for the ER Model

N		Control Latency (Alg. 2)	Control Latency (Lower Bound)	Number of Controllers
200	$p = 0.005$	9.56	9.56	91.48
	$p = 0.01$	18.36	18.36	42.03
400	$p = 0.005$	21.13	21.13	87.14
	$p = 0.01$	108.49	108.49	9.83
600	$p = 0.005$	56.09	56.09	45.67
	$p = 0.01$	387.22	383.33	2.26
800	$p = 0.005$	144.99	144.99	20.09
	$p = 0.01$	785.34	785.42	1.06
1000	$p = 0.005$	276.46	276.26	9.26
	$p = 0.01$	1000	1000	1

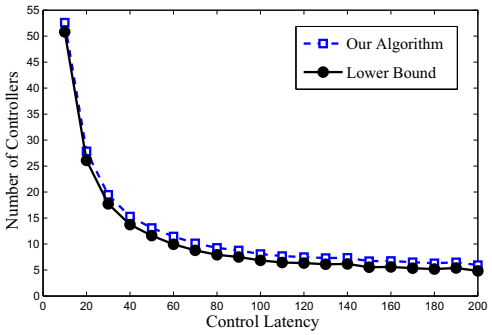


Fig. 6. The number of controllers when the control latency increases from 10 to 200.

TABLE II
Control Latency Comparison for Real Networks

Network	Control Latency (Alg. 2)	Control Latency (Lower Bound)	Number of Controllers
Transportation [1]	16	16	125
UClonline [15]	21	21	641
Facebook-like [16]	45	45	392

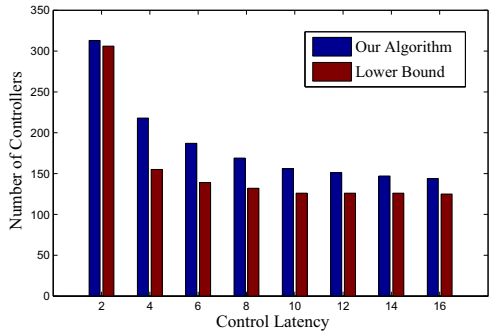


Fig. 7. The comparison between Alg. 3 and the lower bound for the transportation network [1].

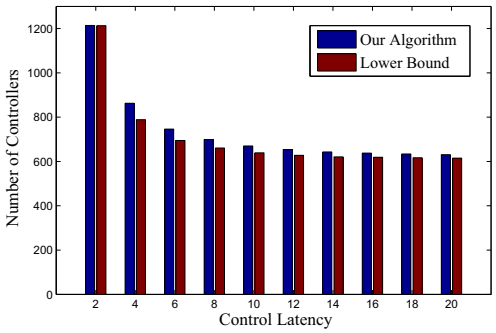


Fig. 8. The comparison between Alg. 3 and the lower bound for the Facebook-like network [16].

VIII. Conclusion

Controlling dynamic networks is important in practical applications. Many previous works have considered the problem of minmizing the number of controllers and the underlying correlation of controllability, but not much effort has been devoted to the issue of control latency. In this paper, we propose the minimum control latency (MCL) problem of designing a controlling pattern to control a dynamic network in a short time with the minimum number of controllers. We prove that the MCL problem is NP-hard. Then we propose a polynomial time approximation algorithm to design a controlling pattern on the basis of two special graph structures: elementary path and elementary cycle. This algorithm can control the network in a short time which is no longer than two times that of the optimal solution. In order to guarantee bounded control latency needed in many practical applications, we also present a constant approximation algorithm for designing the controlling pattern, which uses no more than three times the optimal number of controllers. Extensive simulations have been conducted on both synthetic and real networks, which clearly show that our algorithms have good performance with respect to the lower bounds.

In the future, we will focus on revealing the influence values in the network through distributed methods, and then designing exact controlling pattern to control the network. When the number of controllers is limited, the network may not be controllable, and how to drive the state to arrive at an approximate desired state in finite time is also an interesting problem.

IX. Acknowledgements

This work was supported in part by the National Basic Research Program of China Grants 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61103186, 61033001, 61361136003, and Hong Kong RGC-GRF Grant 714311.

References

[1] V. Colizza, R. P. Satorras, and A. Vespignani. Reaction-Diffusion Process and Metapopulation Models in Heterogeneous Networks. *Nat. Phys.*, 3, 276-282, 2007.
[2] J. M. Dion, C. Commault, and J. V. D. Woude. Generic Properties and Control of Linear Structured Systems: A Survey. *Automation*, 39, 1125-1144, 2003.
[3] P. Erdos, and A. Renyi. On the Evolution of Random Graphs. *Mathematical Institute of the Hungarian Academy of Sciences*, 5, 17-61, 1960.
[4] M. R. Garey, and D. S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Company, New York, USA, 1990.
[5] J. P. Hespanha. Linear Systems Theory. Princeton, New Jersey, 2009.
[6] J. E. Hopcroft, and R. M. Karp. An $n^{5/2}$ Algorithm for Maximum Matching in Bipartite Graphs. *SIAM J. on Comput.*, 2(4), 225-231, 1973.
[7] T. Jia, Y.-Y. Liu, E. Csoka, M. Posfai, J.-J. Slotine, and A.-L. Barabasi. Emergence of Bimodality in Controlling Complex Networks. *Nature Communications*, 4, 1-6, 2013.
[8] R. E. Kalman. Mathmatical Description of Linear Dynamical Systems. *J. Soc. Indus. Appl. Math. Ser.*, A(1), 152-192, 1963.
[9] C.-T. Lin. Structural Controllability. *IEEE Trans. on Automatic Contr.*, 19(3), 201-208, 1974.
[10] R. Liu. The Dracula Dynamic Traffic Network Microsimulation Model. *Simulation Approaches in Transportation Analysis: Recent Advanced and Challenges*, 23-56, 2005.
[11] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabasi. Controllability of Complex Networks. *Nature*, 473(7346): 167-173, 2011.
[12] D. G. Luenberger. Introduction to Dynamic Systems: Theory, Models, & Applications, Wiley, 1979.
[13] J. Marcelino, and M. Kaiser. Critical Paths in a Metapopulation Model of H1N1: Efficiently Delaying Influenza Spreading through Flight Cancellation. *PLoS Currents*, 2012.
[14] T. Nepusz, and V. Tamas. Controlling Edge Dynamics in Complex Networks. *Nature Physics* 8, 568-573, 2012.
[15] T. Opsahl, and P. Panzarasa. Clustering in Weighted Networks. *Soc. Netw.*, 31, 155-163, 2009.
[16] T. Opsahl. Triadic Closure in Two-Mode Networks: Redefining the Global and Local Clustering Coefficients. *Soc. Netw.* 35, 159-167, 2013.
[17] V. M. Popov. Hyperstability of Automatic Control Systems. Springer-Verlag, New York, Berlin,1973.
[18] M. Posfai, Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabasi. Effect of Correlations on Network Controllability. *Sci. Rep.* 3, 1067, 2013.
[19] A. Rahmani, M. Ji, M. Mesbahi, and M. Egerstedt. Controllability of Multi-Agent Systems from a Graph-Theoretic Perspective. *SIAM Journal on Control and Optimization*, 48(1): 162-186, 2009.
[20] J. Ruths, and D. Ruths. Control Profiles of Complex Networks. *Science*, 343(6177), 1373-1376, 2014.
[21] J.-J. E. Slotine, and W. Li. Applied Nonlinear Control. *Prentice-Hall*, 1991.
[22] R. W. Shields, and J. B. Pearson. Structural Controllability of Multi-input Linear Systems. *IEEE Trans. Automat Contr.*, 21, 203-212, 1976.
[23] H. G. Tanner. On the Controllability of Nearest Neighbor Interconnection.. In *IEEE Conference on Decision and Control*, 2004.
[24] D. J. Watts, and S. H. Strogatz. Collective Dynamics of Small-World Networks. *Nature*, 1998.
[25] Z. Yuan, C. Zhao, Z. Di, W.-X. Wang, and Y.-C. Lai. Exact Controllability of Complex Networks. *Nature Communications* 4, 2447, 2013.