Security-Aware Network Analysis for Network Controllability

Shuo Zhang
School of Mathematics and Information Security
Royal Holloway University of London
Egham UK TW20 0EX
Email: MYVA375@live.rhul.ac.uk

Stephen D. Wolthusen
School of Mathematics and Information Security
Royal Holloway University of London
Egham UK TW20 0EX
Email: stephen.wolthusen@rhul.ac.uk

***Abstract*—Although people use critical, redundant and ordinary categories to concisely distinguish the importance of edges in maintaining the controllability of networks in linear time-invariant (LTI) model, a specific network analysis is still uncertain to confirm edges of each category and guide further edge protection. Given a large, sparse, *Erdős-Rényi* random digraph that is in LTI model and has a known maximum matching, as an input network. We address the problem of efficiently classifying its all edges into those categories. By the minimal input theorem, classifying an edge into one of those categories is modeled into analysing the number of maximum matchings having it, and is solved by finding maximally-matchable edges via a bipartite graph mapped by the input network. In the worst case, entire edge classification is executed in linear time except for precomputing a maximum matching of the input network.**

I. Introduction

Controllability of complex networks [1] is one of network properties, it guarantees the networks in LTI model to be controllable via external inputs, and it can be measured by the minimum number of inputs. Besides, network controllability is vulnerable to malicious attack or random failure on edges [2] [3], which increases the minimum number of inputs to fully control the residaul network. To clarify the importance of an edge in maintaining network controllability, Liu *et al.* [1] raised critical, redundant, and ordinary categories: a critical edge exists in all control configurations, its removal gains the minimum number of inputs to control resulting network; a redundant edge is out of any control configuration, and its removal never affects current inputs; removing an ordinary link changes the control configuration, while its removal does not change the minimum number of inputs. Exactly knowing edges of each category is forward-looking to defend network controllability against a single edge removal. Yet, a specific network analysis to confirm all edges for those categories in a general LTI-model network is still uncertain.

Given a large, sparse, *Erdős-Rényi* random digraph that is in LTI model and has a known maximum matching as an input network. We thus address the problem of efficiently classifying edges of an input network into critical, redundant and ordinary categories respectively. Since the minimum input theorem [1] proved that the maximum matching not only determines the minimal inputs to fully control a network in LTI model but also constructs a control configuration, given an edge of the input network, classifying it into one of three categories can be modeled into analysing the number of maximum matchings involving it. Specifically, if an edge out of any maximum matching, it is a redundant edge; if it constructs some maximum matchings, it is an ordinary edge; if it constructs all maximum matchings, it is a critical edge. However, the number of maximum matchings of a general digraph increases exponentially with network size [4], and the best-known maximum matching algorithms [5] [6] are also computationally massive to use for several times, finding the maximum matchings involving this edge can not solve the problem efficiently. Instead, finding the maximally-matchable edges [7] of an input network, which is out of the konwn maximum matching but involved into others, solves the problem of confirming the category of an edge. Because we can efficiently find all maximally matchable edges and we also conclude that all maximally-matchable edges and edges of the known maximum matching and adjacent to them are ordinary edges; edges involves into the known maximum matching without adjacent to any maximally-matchable edge are critical edges; non-maximally matchable edges and out of the known maximum matching are redundant edges.

For our contribution, we efficiently classify all edges of an input network into critical, redundant and ordinary categories respectively by finding all maximally-matchable edges, in linear time except for precomputing the known maximum matching of an input network.

Following paper is structured: section II introduces the network controllability; section III reviews previous related work about edge classification and maximally-matchable edges; section IV models an edge classification and shows all kinds of maximally-matchable links; section V executes entire edge classification. Section VI concludes this paper.

II. Network Controllability

A controllable system can be driven from any initial state to any final state by properly using external inputs within limited time [8] [9] [10]. A linear-time invariant system can be described by a state equation [11]:

$$\dot{x}(t) = \mathbf{A}x(t) + \mathbf{B}u(t) \tag{1}$$

where system vector $x(t) = (x_1(t), x_2(t), \ldots, x_N(t))^T$ captures state of each system vertex of V at time t; $\mathbf{A}$ is a system matrix, and $\mathbf{A} \in \mathbb{R}^{N\times N}$, for each non-zero entry $a_{ij} \in \mathbf{A}$ $(1 \leq i, j \leq N)$, it represents the impact strength of vertex v_i on v_j; $\mathbf{B}$ is the input matrix, and $\mathbf{B} \in \mathbb{R}^{N\times M}$. Each $b_{ij} \in \mathbf{B}$ and $b_{ij} \neq 0$ represents the impact strength of any input u_j on a system vertex of v_i; input vector $u(t) = (u_1(t), u_2(t), \ldots, u_M(t))^T$ holds M external inputs at time t. A system of equation 1 is controllable via M inputs, if and only if the matrix $\mathbf{C} \in \mathbb{R}^{N\times NM}$ and $\mathbf{C} = [\mathbf{B}, \mathbf{AB}, \mathbf{A}^2\mathbf{B}, \ldots, \mathbf{A}^{N-1}\mathbf{B}]$, has full rank, noted by rank$(\mathbf{C}) = N$ [11] [8], which is called the controllability rank condition.

However, value of entries of $\mathbf{A}$ and $\mathbf{B}$ of equation 1 may be just known by approximation except for zero-entries [12], which prevents against using the rank condition to verify whether a system described by equation 1 is controllable or not. Besides, calculating the rank of the matrix $\mathbf{C}$ takes $O(2^N)$ time [1], which is computationally prohibitive, especially for large-scale systems. Virtually, it is also said that effectively using the rank condition is limited to a few dozen system nodes at most [13] [14]. To more effectively indicate whether a system described by equation 1 is controllable or not, structural controllability [12] [15] was raised, in which the structural controllability is defined:

Definition 1 (**Structural Controllability** [12]). *A system of equation 1 is structurally controllable iff there exists a completely controllable system with the same structure as it.*

According to the rank condition [8] and this definition, structural controllability is the necessary but not sufficient condition of complete controllability. Additionally, Lin [12] assessed structurally controllable LTI systems through system's graphic interpretation. A diraph noted by $G(\mathbf{A}, \mathbf{B}) = (V_1 \cup V_2, E_1 \cup E_2)$ is mapped from a system described by equation 1. With a bijection α, for $a_{ij} \in \mathbf{A}$, $\alpha : a_{ij} \to \overrightarrow{\langle v_j, v_i\rangle}$, where $v_i, v_j \in V_1, \overrightarrow{\langle v_j, v_i\rangle} \in E_1$. For $b_{ij} \in \mathbf{B}$, $\alpha : b_{ij} \to \overrightarrow{\langle u_j, v_i\rangle}$, where $\overrightarrow{\langle u_j, v_i\rangle} \in E_2$, $u_j \in V_2$ and $v_i \in V_1$. Then, Lin [12] defined following items to conclud conditions of structural controllability:

Definition 2 (**Inaccessibility** [12]). *Any $v_i \in V_1$ is inaccessible if it can not be approached through a directed path starting from any $u_j \in V_2$ in $G(\boldsymbol{A}, \boldsymbol{B})$.*

Definition 3 (**Dilation of Digraphs** [12]). *In $G(\boldsymbol{A}, \boldsymbol{B})$, $T_1 \subseteq V_1$, $T_2 \subseteq V_1 \cup V_2$ pointing nodes of T_1. $G(\boldsymbol{A}, \boldsymbol{B})$ contains a dilation iff $|T_1| > |T_2|$, where $|T_1|$ and $|T_2|$ are the cardinality of T_1 and T_2.*

Definition 4 (**Stem and Bud** [12]). *In $G(\boldsymbol{A}, \boldsymbol{B})$, a stem is a directed path. A bud is a directed cycle pluse an arc such as $\{\{\overrightarrow{\langle v_1, v_2\rangle}, \overrightarrow{\langle v_2, v_3\rangle}, \ldots, \overrightarrow{\langle v_j, v_1\rangle}\}, \overrightarrow{\langle v_{j+1}, v_j\rangle}\}$, and $\overrightarrow{\langle v_{j+1}, v_j\rangle}$ is called a distinguished edge.*

Definition 5 (**Cactus** [12]). *Any stem of definition 4 is a cactus. Besides, a stem S_0 and buds $B_1, B_2, \ldots, B_l$, then, $S_0 \cup B_1 \cup B_2 \cup \ldots B_l$ is a cactus if the tail of the distinguished edge of any B_i $(1 \leq i \leq l)$ is not the top vertex of S_0 but is the only common vertex of $S_0 \cup B_1 \cup B_2 \cup \ldots B_{i-1}$. A set of vertex-disjoint cacti is called a cactus.*

Based on those items above, conditions of a structurally controllable system are given:

Theorem 1 (**Lin's Structural Controllability Theorem** [12]). *The following three statements are equivalent:*

1) *A system of equation 1 is structurally controllable.*
2) *The digraph $G(\boldsymbol{A}, \boldsymbol{B})$ contains neither inaccessible nodes nor dilation.*
3) *$G(\boldsymbol{A}, \boldsymbol{B})$ is spanned by a cactus.*

Strictly based on the rank condition [8], a structural controllable system can be completely controllable for almost all values of entries of $\mathbf{A}$ and $\mathbf{B}$ of equation 1 except for some pathological cases with certain constrains [12], [15]. For example, a system's graphic interpretation includs nodes $\{n_1, n_2, n_3, n_4\}$, input b_1, and edges $\{\overrightarrow{\langle b_1, n_1\rangle}, \overrightarrow{\langle n_1, n_2\rangle}, \overrightarrow{\langle n_2, n_3\rangle}, \overrightarrow{\langle n_3, n_2\rangle}, \overrightarrow{\langle n_3, n_4\rangle}\}$, then, this system is structurally controllable because its digraph excludes inaccessible nodes and dilation by theorem 1. But this system is not controllable if edge weight is one, because the rank of matrix $\mathbf{C}$ [11], [8] is less than four. In contrast, some structurally controllable systems are always completely controllable such as the system above, whose digraph excludes $\overrightarrow{\langle n_3, n_2\rangle}$, because rank of $\mathbf{C}$ is independent with the value of each non-zero entries. Therefore, Liu *et al.* [1] generalized the minimal input theorem [1]:

Theorem 2 (**Minimal Input Theorem** [1]). *The minimum number of inputs to fully control a network $G(\boldsymbol{A}) = (V_1, E_1)$ is one if there is a perfect matching. Otherwise, inputs directly drive the unmatched nodes related to maximum matching.*

A maximum matching of any graph is a set of maximum number of edges without sharing common nodes. In digraphs, a head of an arc of a maximum matching is called a matched node, otherwise, it is unmatched related to a maximum matching. When all vertices are matched, the digraph is said to have a perfect matching. After this, our input digraph is defined:

Definition 6 (**Input Digraph**). *A large, sparse* Erdős-Rényi *random digraph in the LTI model is determined as an input network D = (V, E), where $V = \{v_i | 1 \leq i \leq N\}$, $E = \{\overrightarrow{\langle v_i, v_j\rangle} | 1 \leq i, j \leq N, i \neq j\}$. Particularly, it excludes parallel arcs, selfloops, and isolated nodes, while includes a maximum matching noted by M_0 and precomputed by algorithm [6] or [5].*

By theorem 2, our input digraph $D = (V, E)$ of definition 6 is controllable by the minimum number of inputs due to the precomputed maximum matching M_0. Besides, since the minimal inputs directly force the unmatched nodes, the maximum matching constructs a control configuration. We thus model the problem of classifying an edge of D into those categories by analysing the number of maximum matchings of D involving each edge. Nevertheless, we do not find any maximum matcing of D for the purpose of efficiently executing entire classification. Rather, we find maximally-matchable edges with respect M_0 in D as a solution.

III. Related Work

The problem of edge classification [16] always attracts the attention of various research areas, especially in artificial intelligence and data mining over years. Yet, it is very seldom to see that there exists the secure-aware edge classification, let along to protect the network controllability against attack or failure on edges. Generally, given a graph $G = (V, E)$ (a social network mostly), where V and E are vertex and edge sets, a subset $E_0 \subseteq E$ has been labeled or classified in advance, then, edge classification problem is raised to determine the labels on or categories of edges of $\{E - E_0\}$. Chronologically, this problem was initially formalized by Liben-Nowell $et\ al.$ [17], called the link-predition problem at the very beginning. Given a social network, people proposed to accurately predict new interaction among existing nodes by analysing proximity among nodes. At the same time, the link-prediction problem was developed by Kunter and Golbeck [18], to further infer the amount of trust value of an edge between two vertices according to edges with known trust values in a given social network. Later, Leskovec $et\ al.$ [19] defined the sign or lable of edges of online social networks as either negative or positive in accordance of the attitude from the generator to the recipient of an edge, which is thus called the edge sign prediction problem and people seeked to reliably predict the sign of a single edge, where lables of remaining edges have been completely known according to social psychology. By then, Chiang $et\ al.$ [20] reviewed some existing algorithms and methods used for the link prediction problem at that time. And Yang $et\ al.$ [21] illustrated that a sign of an edge of social networks can be accurately inferred by user's behavior of decision making. In recent years, researchers of [22] used matrix factorization to predict lables of either positive or negative of multiple edges of social networks compared with single edge prediction of [19]. Up to date, since these previous methods of edge classification problems are based on specific characteristics of networks, Aggarwal $et\ al.$ [16] argued that they can not be well applied into an arbitrary network with various settings and without specific assumptions of a given general network. In this case, they correspondingly raised a general way acccording to the weighted Jaccard coefficient as the foundmental proximity metric to accurately predict sign of each edge of general graphs. By contrast, in our work, we already have three lables: critical, redundant and ordinary, defined by Liu $et\ al.$ [1], while there is no previously labeled edges, and we do not predict the lable of each edge. Rather, we accurately confirm edges of each category by searching all maximally-matchable edges in an input network.

Searching all maximally-matchable edges of a general graph has been pervasively studied over recent decades. Generally speaking, any edge is said to be maximally-matchable with respect to a maximum matching if and only if it can construct a different maximum matching by the edge replacement together with other edges or solely. Initially, people found maximally-matchable edges with respect to a perfect matching, where all nodes are incident to a maximum matching. Rabin and Vazirani [23] created an algorithm randomized to find all maximally-matchable edges in general graphs containing a perfect matching with time complexity of $O(n^{2.376})$, where n is the nummber of graph nodes. Then, in [24], with general graphs, a distinct randomized algorithm finding the Gallai–Edmonds decomposition was given, as a way to find maximally-matchable edges in polynominal time of $O(n^{2.38})$. For deterministic algorithm of finding all maximally-matchable edges, Carvalho and Cheriyan [25] found edges that occurs in at least one perfect matching, called ear decomposition of a matching-covered graph. Their deterministic algorithm runs in $O(nm)$, in which m represents the number of edges. Besides, finding maximally-matchable edges in a bipartite graph, Costa $et\ al.$ [26] decomposed a bipartite graph into three partitions: E_1 whose edges belonging to all maximum matchings; E_0 whose edges out of any maximum matching; edges involved into E_w is neither in E_1 nor E_0. By finding E_1 and E_w, they obtained all maximally-matchable edges in the bipartite graph, and the time complexity of their solution is $O(nm)$. In comparison of the worst-case execution time, Tassa [7] claimed that finding all maximally-matchable edges in a bipartite graph with a known maximum matching is reduced to $O(n + m)$ time. She classified all maximally-matchable edges into few categories. Reviewing her method, we found a problem. Specifically, Tassa applied the breath-first search(BFS) [27] to find some arcs in a digraph that is mapped from the input bipartite graph, as a way to identify some kinds of maximally-matchable edges. However, the BFS algorithm can not traverse all arcs of a digraph except for tree digraphs, it means that some arcs that are incident to any two traversed nodes and are corresponding to valid maximally-matchable edges of the input bipartite graph may be missed. As a result, Tassa's method can not virtually always find all maximally-matchable edges in a bipartite graph with a known maximum matching. By contrast, our input network is a random digraph, our algorithms are all deterministic and we only concern the worst case, which accurately find all maximally-matchable edges of an input network in linear time except for precomputing the known maximum matching of the input network.

IV. Preliminaries

A. Modelling an Edge Classification

To model an edge classification, we firstly show the impact of removing an edge, noted by $e \in E$, on the maximum matching of $D - e$.

Theorem 3. *In $D = (V, E)$ of definition 6, $e \in E$ is removed. Then, the maximum matching of $D - e$ is M_0 if $e \notin M_0$; or it is different from M_0 with the same cardinality if $e \in M_0$ and out of other maximum matchings; or it is smaller than M_0 by one in cardinality if e is in all maximum matchings of D.*

Proof. When $e \notin M_0$, removing e does not influence M_0. Thus, M_0 is still a maximum matching of $D - e$. When $e \in M_0$, and e is excluded by another maximum matching of D. After removing it, the maximum matching excluding e would not be influenced, and it is still maximum matching of $D - e$. When e is in all maximum matchings of D, matching $M_0 - e$ is obtained in $D - e$. Assume this matching is not maximal, and a matching with the same cardinality as M_0 exists, it means there is a maximum matching of D can not be influenced by removing e. However, it contradicts with the condition that e is in all maximum matchings of D. Therefore, $M_0 - e$ is a maximum matching of $D - e$. □

Corollary 1. *In $D = (V, E)$ of definition 6, according to theorem 3, theorem 2, any $e \in E$ is classified into critical category if e is involved into all maximum matchings of D; or e is classified into ordinary category if $e \in M_0$ and out of other maximum matchings; or e is classified into redundant category if it is out of any maximum mathcing of D.*

Proof. If $e \in E$ is in all maximum mathcings of D, by theorem 3, its removal results in $M_0 - e$ as the maximum matching of $D - e$. By theorem 2, the minimum number of inputs of $D - e$ is thus increased by one. So e is a critical edge. If $e \in M_0$ and e is out of another maximum matching, by theorem 2, 3, removal of e does not influence another control configuration, which would still exist in $D - e$ with the same minimum number of inputs as before. Thus, e is an ordinary edge. If $e \in E$ is out of any maximum matching of D, by theorem 2, 3, removal of e can not influence any existing control configuration of D. Thus, e is a redundant edge. □

By corollary 1, we thus model an edge classification of $D = (V, E)$ of definition 6 into checking how many maximum matchings involve it. However, as mentioned before, finding all maximum matchings of a large digraph such as D is computationally prohibitively. Therefore, finding all maximum matchings of D can not solve the problem of classifying all arcs of D into critical, redundant and ordinary categories, respectively.

Instead, because any two maximum matchings must share one or more common nodes, we can obtain different maximum matchings from M_0 by edge replacement rather than recomputation. Edges out of M_0 and replacing edges of M_0 to construct a different maximum matching are called the maximally-matchable edges with respect to M_0 [7]. Obviously, maximally-matchable edges are also involved into other maximum matchings of D, while are excluded by M_0. Therefore, we conclude:

Lemma 1. *In $D = (V, E)$ of definition 6, given any $e \in M_0$, if e is not adjacent to any maximally-matchable edges with respect to M_0, e is in all maximum matchings; if e is adjacent to any maximally-matchable edge at its head or tail, e is out of one or more maximum matchings.*

Proof. In $D = (V, E)$ of definition 6, because the maximally-matchable edges with respect to M_0 are the arcs in another maximum matchings but excluded by M_0, if e is adjacent to a maximally-matchable edge, e is excluded by a different maximum matching from M_0 at least. If e is not adjacent to any maximally-matchable edge, it means that e can not be excluded by any maximum matching in D. Thus, e is involved into all maximum matchings of D. □

Corollary 2. *According to corollary 1 and lemma 1, in n $D = (V, E)$ of definition 6, with respect to M_0, any maximally-matchable edge is an ordinary link. Given any $e \in M_0$, if e is adjacent to a maximally-matchable edge, e is an ordinary link. Otherwise, e is a critical link.*

By corollary 2, arcs neither in M_0 nor the maximally-matchable are redundant links. Finally, our problem of of classifying all arcs of D into critical, redundant and ordinary categories, respectively is solved by finding all maximally-matchable edges with respect to M_0.

B. Maximally-matchable Edges

To find all maximally-matchable edges of $D = (V, E)$ of definition 6 with respect to M_0, we map D into a bipartite graph, noted by $B = (V_B, E_B)$:

Definition 7 ($B = (V_B, E_B)$). *Given a bijection β and $D = (V, E)$ of definition 6, a bipartite graph $B = (V_B, E_B)$, $|E_B| = |E|$, $V_B = V_B^+ \cup V_B^-$ is obtained. For each arc of D, there is $\beta : \overrightarrow{\langle v_i, v_j \rangle} \rightarrow (v_i^+, v_j^-)$, $(v_i^+, v_j^-) \in E_B$, $v_i^+ \in V_B^+$, $v_j^- \in V_B^-$. M_B notes the maximum matching mapped from M_0 of D in B.*

By definiton 7, any maximally-matchable edges of D with respect to M_0 is mapped into a maximally-matchable link of B with respect to M_B. Thus, we find all maximally-matchable edges of B with respect to M_B. With respect to M_B, we define all kinds of maximally-matchable edges of B:

Definition 8 (**Alternating Link**). *In $B = (V_B, E_B)$ of definition 7, with respect to M_B, any edge $(v_i^+, v_j^-) \in E_B$ is an alternating link if either $v_i^+ \in M_B, v_j^- \notin M_B$ or $v_i^+ \notin M_B, v_j^- \in M_B$.*

Theorem 4. *$B = (V_B, E_B)$ of definition 7 holds at least one different maximum matching from M_B, iff any single edge $(v_i^+, v_j^-) \notin M_B$ is an alternating link with respect to M_B.*

Proof. When (v_i^+, v_j^-) is an alternating link with respect to M_B, if $v_i^+ \in M_B$, $v_j^- \notin M_B$, and there must be $(v_i^+, v_k^-) \in M_B$, by replacing (v_i^+, v_k^-) with (v_i^+, v_j^-), a maximum matching is obtained: $\{M_B - (v_i^+, v_k^-), (v_i^+, v_j^-)\}$. Similarly, if $v_j^- \in M_B$, $v_i^+ \notin M_B$, and a maximum matching by replacing an edge of M_B and incident to v_j^- would be obtained.

When a maximum matching of B is obtained by replacing an edge noted by $(v_i^+, v_k^-) \in M_B$ with an edge $(v_i^+, v_j^-) \notin M_B$, and it can be expressed by $\{M_B - (v_i^+, v_k^-), (v_i^+, v_j^-)\}$. Therefore, $v_j^- \notin M_B$, and by definition 8, (v_i^+, v_j^-) is an alternating link with respect to M_B. □

Additionally, maximally-matchable edge sets are defined in the following:

Definition 9 (Alternating Cycle). *With respect to M_B of $B = (V_B, E_B)$ of definition 7, with $\{m_1, m_2, \dots, m_t\} \subseteq M_B$, a matching set $\{e_1, e_2, \dots, e_t\} \nsubseteq M_B$ $(1 < t \leq |M_B|)$ is an alternating cycle, if either $e_i \cap m_i \in V_B^+$ $(1 \leq i \leq t)$, $e_j \cap m_{j+1} \in V_B^-(1 \leq j < t)$ and $e_t \cap m_1 \in V_B^-$; OR, $e_i \cap m_i \in V_B^-$ $(1 \leq i \leq t)$, $e_j \cap m_{j+1} \in V_B^+(1 \leq j < t)$ and $e_t \cap m_1 \in V_B^+$.*

Definition 10 (Crossed Alternating Path). *With respect to M_B of $B = (V_B, E_B)$ of definition 7, with $\{m_1, m_2, \dots, m_t\} \subseteq M_B$, a matching set $\{e_1, e_2, \dots, e_t\} \nsubseteq M_B$ $(1 < t \leq |M_B|)$ is a crossed alternating path if either $e_i \cap m_i \in V_B^+$ $(1 \leq i < t)$, $e_j \cap m_{j+1} \in V_B^-(1 \leq j < t)$ and $e_t \cap V_B^+ \notin M_B$, $e_t \cap m_1 \in V_B^-$; OR, $e_i \cap m_i \in V_B^-$ $(1 \leq i < t)$, $e_j \cap m_{j+1} \in V_B^+(1 \leq j < t)$ and $e_t \cap m_1 \in V_B^+$, $e_t \cap V_B^- \notin M_B$.*

Definition 11 (Uncrossed Alternating Path). *With respect to M_B of $B = (V_B, E_B)$ of definition 7, with $\{m_1, m_2, \dots, m_t\} \subseteq M_B$, a matching set $\{e_1, e_2, \dots, e_t\} \nsubseteq M_B$ $(1 < t \leq |M_B|)$ is an uncrossed alternating path if either $e_i \cap m_i \in V_B^+$ $(1 \leq i \leq t)$, $e_j \cap m_{j+1} \in V_B^-(1 \leq j < t)$ and $e_t \cap V_B^- \notin M_B$; OR, $e_i \cap m_i \in V_B^-$ $(1 \leq i \leq t)$, $e_j \cap m_{j+1} \in V_B^+(1 \leq j < t)$ and $e_t \cap V_B^+ \notin M_B$.*

Lemma 2. *With respect to M_B of $B = (V_B, E_B)$ of definition 7, the alternating cycle, crossed alternating path, and uncrossed alternating path are all able to construct different maximum matchings from M_B by edge replacement.*

Proof. We prove that in B, each existing alternating cycle, crossed alternating path and uncrossed alternating path with respect to M_B can respectively construct a different maximum matching. An alternating cycle like $\{e_1, e_2, \dots, e_t\} \nsubseteq M_B$ $(1 < t \leq |M_B|$, replaces its adjacent $\{m_1, m_2, \dots, m_t\} \subseteq M_B$ can obtain a maximum matching: $\{M_B - \{m_1, m_2, \dots, m_t\}, \{e_1, e_2, \dots, e_t\}\}$. With respect to M_B, the crossed alternating path and the uncrossed alternating path in B can also respectively construct a different maximum matching from M_B by replacing matching links of M_B that are separately adjacent to them. □

According to lemma 2, multiple disjoint alternating links, corssed and uncrossed alternating paths can also construct a different maximum matching from M_B by edge replacement. By contrast, in $B = (V_B, E_B)$ of definition 7, we call a matching set the minimal maximally-matchable edge set if its cardinality is bigger than one, and a removal of its any edge would cause either the removed edge or the remaining matching to no longer able to oconstruct a different maximum matching from M_B by edge replacement.

Theorem 5. *In $B = (V_B, E_B)$ of definition 7, any minimal maximally-matchable edge set with respect to M_B can be only classified into one of those matching sets defined from definition 9-11.*

Proof. Based on lemma 2, alternating cycle, crossed alternating path, and uncrossed alternating path are all able to construct different maximum matchings from M_B by edge replacement. Also, these matching sets of definition 9-11 are the minimal maximally-matchable edge sets, because any e_i for $1 \leq i < t, 1 \leq t \leq |M_B|$, of any those sets is adjacent to two edges of M_B. Then, either e_i or $\{e_1, e_2, \dots, e_{t-1}\}$ can not construct different maximum matching from M_B by edge replacement when e_i or e_t is removed.

Assume there is another kind of minimal maximally-matchable edge set except for them defined from definition 9-11. On the one hand, if its edges are all among nodes of M_B, and adjacent to the same number of edges of M_B. Because any edge out of M_B and incident to nodes of M_B is adjacent to two edges of M_B, such egde set can be only the alternating cycle of definition 9. On the other hand, if there exists the edge incident to nodes out of M_B, and this matching set is still minimal, there can be only one alternating link of definition 8, while remaining edges are among nodes of M_B, and they can not construct a alternating cycle. As a result, such edge set can be either crossed or uncrossed alternating path. □

Additionally, we also conclude the distribution for crossed and uncrossed alternating paths via follwoing statement:

Theorem 6. *In $B = (V_B, E_B)$ of definition 7, any two crossed or uncrossed alternating paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$, respectively. Then, they must be vertex-disjoint.*

Proof. Assume there exists one shared node by two crossed or uncrossed alternating paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$, respectively. Then, starting from this shared node, a path alternatively involving edges of and out of M_B would exist, and it contains more edges out of M_B than edges of M_B. In this case, we can obtain a matching bigger than M_B in cardinality. However, it contradicts with the maximality of M_B. Therefore, any two crossed or uncrossed alternating paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$ respectively, must be disjoint. □

Besides, according to the theorem 6, we deduce the relationship among the alternating cycle, crossed or uncrossed alternating path:

Corollary 3. *In $B = (V_B, E_B)$ of definition 7, by theorem 6, given any two crossed or uncrossed alternating paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$, respectively. Then, they can not be adjacent to a same alternating cycle.*

Proof. By definition 9-11, any alternating cycle shares common edges with the crossed or uncrossed alternating path that is adjacent to it at one or more nodes. Assume this alternating cycle is adjacent to two crossed or uncrossed alternaitng paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$ respectively. From the shared edges between crossed or uncrossed alternating path and this alternating cycle, a path alternatively involving edges of and out of M_B would exist. As mentioned before, existence of a such alternating path contradicts with the maximality of M_B. Therefore, any alternating cycle can not be adjacent to any two crossed or uncrossed alternating paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$ at the same time. □

V. Entire Edge Classification

We firstly identify all maximally-matchable links of $B = (V_B, E_B)$ of definition 7, where a digraph mapped from B is used:

Definition 12. *Digraph $D' = (V', E')$ with $|E'| = |E_B - M_B|$ is mapped from $B = (V_B, E_B)$ of definition 7. Firstly, any edge of E_B is directed from V_B^+ to V_B^-. Then, given a bijection ω, for each edge $m_i \in M_B$ with $1 \leq i \leq |M_B|$, there is $\omega : m_i \to u_i$ and $u_i \neq V_B$. Finally, all remaining nodes and arcs belong to V' and E'. Also, $V' = \{v_i' | 1 \leq i \leq |V'\}$, and $E' = \{e_j' | 1 \leq j \leq |E'|\}$.*

By definition 9-11, and definition 12, each crossed and uncrossed alternating path of $B = (V_B, E_B)$ of definition 7, with respect to M_B mapps into a directed path of D', while any alternating cycle of B with respect to M_B mapps into a directed cycle of D'. Then, Our entire classifiaction process is shown as follow:

Algorithm 1 Entire Egde Classification Outline

```
Input: $D = (V, E)$, $M_0$
Output: A category for each edge of $E$
1: Map $D$ with $M_0$ into $B = (V_B, E_B)$ with $M_B$ by definition 7.
2: Map $B$ into $D' = (V', E')$ by definition 12.
3: Find arcs of $D'$ related to maximally-matchable edges of $B$ by using algorithms of V-A, V-B and V-C.
4: Classify edges of $D$ into critical, redundant and ordinary categories by algorithm of V-D.
```

A. Find arcs related to alternating links

This algorithm finds arcs related to alternating links. S_a, S_b denote two sets of returned arcs, while $S_c \subseteq V'$ notes a node set mapped from an edge of M_B, and $v_i' \in S_c$. $Adj(v_i')$ denotes nodes adjacent to v_i', and any node of $Adj(v_i')$ is noted by $v_k' \in Adj(v_i')$.

Algorithm 2 Identify arcs mapped by alternating links

```
Input: $D' = (V', E')$, $S_c$
Output: Arcs of $D'$ mapped by alternating links of $B$
1: Label nodes of $S_c$
2: $S_a = \emptyset$ and $S_b = \emptyset$
3: while $S_c \neq \emptyset$ and $v_j' \in S_c$ do
4:     for $\exists v_k' \in Adj(v_i')$ do
5:         $Adj(v_i') = Adj(v_i') - v_k'$
6:         if $v_k'$ out of $S_c$ then
7:             $S_a = S_a + \overrightarrow{\langle v_k', v_i' \rangle}$ or $S_b = S_b + \overrightarrow{\langle v_i', v_k' \rangle}$
8:     $S_c = S_c - v_i'$
9: return $S_a$ and $S_b$
```

Proof. Initially, since S_c involves the nodes that are mapped from M_B, by definition 12, S_c can be derived in $O(|V'|)$ time. Then, all nodes of S_c are labelled in $O(|V'|)$ time to distinguish if a node is in or out of S_c rather than searching it in S_c. Firstly, a node $v_i' \in S_b$ is chosen. **for** loop considers each adjacent node of v_i' by definition 8 and 12. If an adjacent node of v_i' is out of S_b, this arc is mapped from an alternating link of B and added into either S_a or S_b. Once $\nexists v_i' \in Adj(v_i')$, all nodes adjacent to v_i' are checked and removed from $Adj(v_i')$ and this procedure removes v_i' from S_c. After this, each adjacent node of a newly-chosen node of remaining S_c would still be considered as before. Finally, $S_c = \emptyset$ terminates this procedure due to node removal of S_c. At that time, S_a and S_b containing arcs corresponding to alternating links of B are returned. For time complexity, operation of choosing all nodes of S_c takes $O(|V'|)$ time. Due to line 5, 8, examing adjacent nodes of every node of S_c is executed in $O(|E'|)$ time. Thus, total running time of this procedure is $O(|V'|+|E'|)$ excluding obtaining D'. □

The following algorithms find arcs of D' related to edges of alternating paths and alternating cycles of B, according to arcs of S_a and S_b by the definition 9, 10, 11, and 12.

B. Find Arcs related to edges of Alternating Paths and Cycles

If $S_a \neq \emptyset$, following algorithm traverses directed paths of D' to find arcs mapped from all edges of crossed, uncrossed alternating paths, and alternating cycles of B.

We denote an arc of S_a by e_i', $e_i' \in S_a$. We use P_0 to present an arc set and $P(P_0)$ represents a set of arcs out of P_0 and pointed by arcs of P_0. Any arc of $P(P_0)$ is noted by $e_j' \in P(P_0)$.

Algorithm 3 Find edges of alternating paths and cycle via D'

```
Input: $D' = (V', E')$, $S_a \neq \emptyset$, $P_0$, $n$
Output: Arcs mapped from alternating paths or cycles of $B$
1: while $\exists e_i' \in S_a$ do
2:     $P_0 = \emptyset$
3:     $E' = E' - e_i'$ and $P_0 = P_0 + e_i'$
4:     for $\exists e_j' \in P(P_0)$ do
5:         $P_0 = P_0 + e_j'$ and $E' = E' - e_j'$
6:     return $P_0$
7: return $E'$
```

Proof. By definition 10, 11 and 12, any edge of a crossed or an uncrossed alternating path in $B = (V_B, E_B)$ of definition 7 mapps into an arc of $D' = (V', E')$ within a directed path starting from an arc of S_a, while any edge of an alternating cycle mapps into an arc of a cycle among nodes of S_c.

Accordingly, this procedure finds edges approached by each arc of S_a, because they are in either a directed paths or directed cycles. Firstly, any $e_i' \in S_a$ is chosen, added into P_0. Then, the **for** loop finds all arcs approached by e_i' through a directed path starting from e_i'. If $P(P_0) \neq \emptyset$, e_i' currently must point an arc noted by e_j', then, e_j' is added into P_0 to following search. Since $e_j^{'}$ is approached by $e_i^{'}$ within a path, $e_j^{'}$ currently corresponds to an edge of a crossed or uncrossed alternating path of B. If $P(P_0)=\emptyset$, all arcs approached from $e_i^{'}$ have been traversed. In particular, once an arc of $P(P_0)$ also points a node of an arc of P_0 in a same path, a cycle is produced by it, while this arc is approached by e_i through a path. Therefore, P_0 contains the arcs of $D^{'}$ correspond to edges of alternating cycle or crossed or uncrossed alternating paths. Then P_0 is returned. After this, another arc of S_a is chosen from remaining $E^{'}$, and P_0 is emptied to collect directed paths and cycles approached from an arc of S_a as before. When $S_a=\emptyset$, due to edge removal from $E^{'}$, this procedure terminates. For the running time of this procedure, except for precomputing S_a by **algorithm** of V-B and obtaining $D^{'}=(V^{'},E^{'})$ of definition 12, since each traversed arc of $E^{'}$ is removed from $E^{'}$ and added into P_0, each arc of $E^{'}$ is thus traversed once at most. As a result, total running time is thus $O(|E^{'}|)$. □

According to theorem 6 and corollary 3, searching arcs from S_a does not influence searching arcs by S_b in $D^{'}=(V^{'},E^{'})$. Thus, when $S_b\neq\emptyset$, algorithm V-B can be slightly modified to finding arcs of $D^{'}$ corresponding to edges of crossed or uncrossed alternating paths or alternating paths with respect to M_B in $O(|E^{'}|)$ time. In detail, returned $E^{'}$ by algorithm V-B is used as an input, $P(P_0)$ represents the arc set invoving arcs out of and pointing arcs of P_0, because an arc of $D^{'}$ corresponding to an edge of crossed or uncrossed alternating paths or alternating cycles of B now approaches an arc of S_b along with a directed path.

C. Search Arcs Mapped by Alternating Cycles

By definition 9 and definition 12, as mentioned before, any alternating cycle with respect to M_B of B is mapped into a directed cycle in $D^{'}=(V^{'},E^{'})$, we thus find arcs of cycles of $D^{'}$ mapped from the alternating cycles disjoint with any crossed or uncrossed alternating paths with respect to M_B of $B=(V_B,E_B)$. To do this, we search strongly connected components according to following theorem:

Theorem 7. *In $D^{'}=(V^{'},E^{'})$, any arc of a strongly connected component must be involved into a directed cycle.*

Proof. A strongly connectedd component is a component of a digraph whose every vertex can approach any others through a directed path [28] [29]. Assume that an arc noted by $\overrightarrow{\langle v_i^{'},v_j^{'}\rangle}\in E^{'}$ is involved into a strongly connected component, but it is out of any cycle. Then, any distinct node $v_k^{'}$ of a same component can be approached by $v_j^{'}$ through a directed path, while $v_k^{'}$ can not approach $v_i^{'}$ via a directed path because $\overrightarrow{\langle v_i^{'},v_j^{'}\rangle}\in E^{'}$ is excluded by any directed cycle. However, in this case, $v_i^{'}$ and $v_k^{'}$ are in a same component is contradicted. Therefore, any arc $\overrightarrow{\langle v_i^{'},v_j^{'}\rangle}\in E^{'}$ in a strongly connected component is involved into a directed cycle. □

The strongly connected components can be effectively identified by using the algorithms of [29]. In the following algorithm, the strongly components among nodes of the returned $E^{'}$ after finding arcs via S_b will be identified and each of arc of the components will be returned, where such arc set returned by previous algorithm is represented by S_d and $V(S_d)$ notes the nodes of S_d.

Algorithm 4 Find arcs mapped by alternating cycles

```
Input: $S_d$
Output: Arcs of $D^{'}$ mapped by alternating cycles of $B$
1: $S_d \subseteq D^{'}$ and $S_d \neq \emptyset$
2: Find strongly connected components among $S_d$ by the algorithm of [29].
3: Label arcs of each identified strongly connected components.
4: return Labelled arcs
```

Proof. By using the algorithm of [29], among the arc set S_d, the total running time can be represented by $O(|S_d|+|V(S_d)|)$. Because $|S_d|<|E^{'}|$ and $|V(S_d)|<|V^{'}|$, except for obtaining S_d, running time of this procedure is $O(|V^{'}|+|E^{'}|)$ with the digraph $D^{'}=(V^{'},E^{'})$ of definition 12. □

D. Arc classification of an Input Digraph

After obtaining some arcs of $D^{'}=(V^{'},E^{'})$ of definition 12, by which, we classify all arcs of $D=(V,E)$ of definition 6. In notation, we denote all returned arcs by algorithms of V-A, V-B and V-C by $E_0^{'}$, and $E_0^{'}\subseteq E^{'}$. Besides, e_i notes arc of M_0 and e_j represents a maximally-matchable edge.

Algorithm 5 Classify all arcs of D

```
Input: $D=(V,E)$, $M_0$, $B=(V_B,E_B)$, $M_B$, $E_0^{'}$
Output: Classified arcs of $D$
1: if $E_0^{'}=\emptyset$ then
2:     $D$ has no ordinary links
3:     return Arcs of $M_0$ are critical links and Arcs out of $M_0$ are redundant links.
4: else if $E_0^{'}\neq\emptyset$ then
5:     Identify edges of $E_B$ via $E_0^{'}$
6:     Identify arcs of $E$ via identified edges of $E_B$
7:     Label each identified arc of $E$ with maximally-matchable
8:     return Maximally-matchable arcs of $E$ are ordinaly link.
9:     for $\exists e_j \in E$ do
10:        $E=E-e_j$
11:        while $e_j$ adjacent to $e_i\in M_0$ do
12:            return $e_i$ is ordinary link.
13:            $M_0=M_0-e_i$
14:    return Arcs of $M_0$ are critical links and Arcs of $E$ are redundant links.
```

Proof. If $E_0^{'}=\emptyset$, there is no any maximally-matchable edges in B with respect to M_B, which further means that D excludes any maximally-matchable edge with respect to M_0. By corollary 2, all arcs of M_0 are critical links, and others are redundant links. If $E_0^{'} \neq \emptyset$, by definition 12, edges of B mapping into them would be identified in $O(|E_B|)$ time and those identified edges of B is also used to identify arcs of D by definition 7 in $O(|E_B|)$ time, which are labelled in $O(|E_B|)$ time to know if an arc of E is maximally-matchable or not. Then, by corollary 2, following procedure checks if a maximally-matchable edge is adjacent to $e \in M_0$ or not to confirm ordinary links of M_0. In detail, each maximally-matchabe edge e_j is chosen and removed from E. If $\exists e_i \in M_0$ is adjacent to e_j, it is an ordinary link and removed from M_0. When all maximally-matchable edges are examed, all ordinary links of D are returned, while those remaining arcs of M_0 and E are critical and redundant links respectively. Since each maximally-matchable edge and each edge of M_0 are considered once at most, the total running time of this procedure is $O(|E_B| + |E|)$, except for obtaining $B = (V_B, E_B)$, M_B and $E^{'}$. $\square$

E. Time complexity analysis

Reviewing algorithm 1, except for precomputing M_0 of $D = (V, E)$ of defintion 6, total running time to classify entire arcs of D is the sum of the running time of each steps. By definition 7 and 12, there are: $|E_B| = |E|$, $|E_B| > |E^{'}|$, and $|V^{'}| < |V_B|$, $2|V| \geq |V_B|$ then, mapping D into B of definition 7 thus costs $\Theta(|E|)$ time, and then mapping B into $D^{'}$ of defintion 12 takes also $\Theta(|E|)$ time. Also, running time of algorithms of V-A, V-B, V-C and V-D, can be also represented by $O(|V|+|E|)$, $O(|E|)$, $O(|V|+|E|)$ and $O(|E|)$ respectively. Eventually, classifying all arcs of $D = (V, E)$ into critical, redundant and ordianry categories, is executed in $O(|V| + |E|)$ time.

VI. Conclusion

Edges of minimal-input controllable networks in LTI model are identified by critical, redundant and ordinary categories to show the importance of each involved edge in maintaining network controllability or the minimum number of inputs. Nevertheless, an efficient classification method seems still in lack. To solve this problem, we use a one-to-one mapped bipartite graph by the given input network to find all kinds of maximally-matchable edges in linear time, which plays a critical role in determinging what arcs should be classified into which categories. According to the adjacency between each arc of the known maximum matching and maximally-matchable edges, we can easily classify all arc of an input network in linear time except for precomputation of a maximum matching of the input network. For our future work, we would like to define few categories to show the importance of vertices in maintaining network controllability, and then classify all vertices of an input network into them efficiently.

References

[1] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Controllability of complex networks," *Nature*, vol. 473, no. 7346, pp. 167–173, 2011.

[2] J. Ruths and D. Ruths, "Robustness of network controllability under edge removal," in *Complex Networks IV*. Springer, 2013, pp. 185–193.

[3] D. Parekh, D. Ruths, and J. Ruths, "Reachability-based robustness of network controllability under node and edge attacks," in *Signal-Image Technology and Internet-Based Systems (SITIS), 2014 Tenth International Conference on*. IEEE, 2014, pp. 424–431.

[4] L. Zdeborová and M. Mézard, "The number of matchings in random graphs," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2006, no. 05, p. P05003, 2006.

[5] S. Micali and V. V. Vazirani, "An o (v— v— c— e—) algorithm for finding maximum matching in general graphs," in *Foundations of Computer Science, 1980., 21st Annual Symposium on*. IEEE, 1980, pp. 17–27.

[6] J. E. Hopcroft and R. M. Karp, "An n^5/2 algorithm for maximum matchings in bipartite graphs," *SIAM Journal on computing*, vol. 2, no. 4, pp. 225–231, 1973.

[7] T. Tassa, "Finding all maximally-matchable edges in a bipartite graph," *Theoretical Computer Science*, vol. 423, pp. 50–58, 2012.

[8] R. Kalman, "On the general theory of control systems," *Automatic Control, IRE Transactions on*, vol. 4, no. 3, pp. 110–110, 1959.

[9] D. Luenberger, "Introduction to dynamic systems: theory, models, and applications," 1979.

[10] J.-J. E. Slotine, W. Li *et al.*, *Applied nonlinear control*. prentice-Hall Englewood Cliffs, NJ, 1991, vol. 199, no. 1.

[11] R. E. Kalman, "Mathematical description of linear dynamical systems," *Journal of the Society for Industrial and Applied Mathematics, Series A: Control*, vol. 1, no. 2, pp. 152–192, 1963.

[12] C. T. Lin, "Structural controllability," *Automatic Control, IEEE Transactions on*, vol. 19, no. 3, pp. 201–208, 1974.

[13] B. Liu, T. Chu, L. Wang, and G. Xie, "Controllability of a leader–follower dynamic network with switching topology," *IEEE Transactions on Automatic Control*, vol. 53, no. 4, pp. 1009–1013, 2008.

[14] 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*, vol. 48, no. 1, pp. 162–186, 2009.

[15] R. Shields and J. Pearson, "Structural controllability of multiinput linear systems," *IEEE Transactions on Automatic control*, vol. 21, no. 2, pp. 203–212, 1976.

[16] C. Aggarwal, G. He, and P. Zhao, "Edge classification in networks," in *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*. IEEE, 2016, pp. 1038–1049.

[17] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *journal of the Association for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[18] U. Kuter and J. Golbeck, "Sunny: A new algorithm for trust inference in social networks using probabilistic confidence models," in *AAAI*, vol. 7, 2007, pp. 1377–1382.

[19] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting positive and negative links in online social networks," in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 641–650.

[20] K.-Y. Chiang, N. Natarajan, A. Tewari, and I. S. Dhillon, "Exploiting longer cycles for link prediction in signed networks," in *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 2011, pp. 1157–1162.

[21] S.-H. Yang, A. J. Smola, B. Long, H. Zha, and Y. Chang, "Friend or frenemy?: predicting signed ties in social networks," in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2012, pp. 555–564.

[22] P. Agrawal, V. K. Garg, and R. Narayanam, "Link label prediction in signed social networks." in *IJCAI*, 2013.

[23] M. O. Rabin and V. V. Vazirani, "Maximum matchings in general graphs through randomization," *Journal of Algorithms*, vol. 10, no. 4, pp. 557–567, 1989.

[24] J. Cheriyan, "Randomized o(m(—v—)) algorithms for problems in matching theory," *SIAM Journal on Computing*, vol. 26, no. 6, pp. 1635–1655, 1997.

[25] M. H. D. Carvalho *et al.*, "An o (ve) algorithm for ear decompositions of matching-covered graphs," *ACM Transactions on Algorithms (TALG)*, vol. 1, no. 2, pp. 324–337, 2005.

[26] M.-C. Costa, "Persistency in maximum cardinality bipartite matchings," *Operations Research Letters*, vol. 15, no. 3, pp. 143–149, 1994.

[27] T. H. Cormen, *Introduction to algorithms*. MIT press, 2009.

[28] J. Bang-Jensen and G. Z. Gutin, *Digraphs: theory, algorithms and applications*. Springer Science & Business Media, 2008.

[29] R. Tarjan, “Depth-first search and linear graph algorithms,” *SIAM journal on computing*, vol. 1, no. 2, pp. 146–160, 1972.