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 controllability of networks in linear time-invariant (LTI) model, a specific network analysis is still uncertain to confirm edges of each category for further edge protection. Given a large, sparse, *Erdős-Rényi* random digraph with a precomputed maximum matching in LTI model 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, while it 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 removal of a critical edge gains the minimum number of inputs to control residual network; removing a redundant edge never affects currently minimal inputs; removing an ordinary link changes the control configuration, except for 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 is in some maximum matchings, it is an ordinary edge; if it is in 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 using the best-known maximum matching algorithms [5] [6] for several times is too computationally massive to be a solution. 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 classifying an edge. This is 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 adjacent to them are ordinary edges; edges involves into the known maximum matching without adjacent to any maximally-matchable edge are critical edges; edges not maximally matchable and out of the known maximum matching are redundant edges.

For our contribution, we efficiently classify 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; 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 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 shows the impact strength of system

vertex noted by v_i on another one noted by v_j; $\mathbf{B}$ is the input matrix, and $\mathbf{B} \in \mathbb{R}^{N\times M}$. Each $b_{ij} \in \mathbf{B}$ and $b_{ij} \neq 0$ shows the impact strength of any input noted by u_j on a system vertex 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 is controllable via M inputs described by equation 1, 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 $\text{rank}(\mathbf{C}) = N$ [11] [8], which is called the controllability rank condition.

However, value of entries of matrix $\mathbf{A}$ and $\mathbf{B}$ are known by approximation except for zero-entries [12], which prevents against using the rank condition to verify if a system of equation 1 is controllable or not. Besides, time complexity of calculating the rank of the matrix $\mathbf{C}$ is $O(2^N)$ [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 avoid these two constrains, structural controllability [12] [15] was raised, and it is defined below:

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, 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 a $b_{ij} \in \mathbf{B}$, $\alpha : b_{ij} \to \overrightarrow{\langle u_j, v_i\rangle}$, where $\overrightarrow{\langle u_j, v_i\rangle} \in E_2$, $u_j \in V_2$ and $v_i \in V_1$. With $G(\mathbf{A}, \mathbf{B})$, Lin [12] defined following items to give 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, with stem S_0 and buds $B_1, B_2, \ldots, B_l$, $S_0 \cup B_1 \cup B_2 \cup \ldots B_l$ 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.*

Then, conditions of structural controllability 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.*

A structurally 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 by its state equation has nodes $\{n_1, n_2, n_3\}$, input b_1, and edges $\{\overrightarrow{\langle b_1, n_1\rangle}, \overrightarrow{\langle n_1, n_2\rangle}, \overrightarrow{\langle n_1, n_3\rangle}, \overrightarrow{\langle n_3, n_2\rangle}, \overrightarrow{\langle n_2, n_3\rangle}\}$, 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. Strictly based on the rank condition [8] and referring to structurally controllable systems except for pathological cases, Liu *et al.* [1] generalized the minimal input theorem [1] to fully control networks in LTI model:

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 a 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, an 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 it 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 M_0, which also constructs a control configuration. We thus model the problem of classifying an edge of D into one of those categories by analysing the number of maximum matchings of D involving it. Nevertheless, we do not find any maximum matcing of D for the purpose of efficiently executing entire classification. Rather, finding maximally-matchable edges with respect M_0 in D is determined as the 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 set and edge set, 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, on which people proposed to predict new interaction among existing nodes in a social network by analysing proximity among nodes. Meanwhile, this problem was developed by Kunter and Golbeck [18], to further infer the amount of trust of an edge between two vertices according to edges with known trust values. Later, Leskovec $et\ al.$ [19] defined the sign or lable of edges of online social networks as either negative or positive based on 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 determined by social psychology. By then, Chiang $et\ al.$ [20] reviewed some existing algorithms and methods used for the link prediction problem, 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 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. 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 maximally-matchable edges of a general graph has been pervasively studied over recent decades. Generally, 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. Initially, Rabin and Vazirani [23] designed a randomized algorithm finding 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 algorithms, with the same the purpose, Carvalho and Cheriyan [25] found edges in at least one perfect matching, called ear decomposition of a matching-covered graph. Their deterministic algorithm runs in $O(nm)$, and m represents the number of edges. Besides, Costa $et\ al.$ [26] found maximally-matchable edges in a bipartite graph. They 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, all maximally-matchable edges are obtained, and the time complexity is $O(nm)$. Compared with 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. In detail, Tassa applied the breath-first search(BFS) [27] to find some arcs in a digraph mapped by the input bipartite graph, as a way to find 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 corresponding to valid maximally-matchable edges of the input bipartite graph may be missed. As a result, Tassa's method can not 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 execution, where we 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 and solving edge classification

We define $M_0^{'}$ as any different maximum matching of D from M_0. Then, impact of removing any edge $e \in E$ on the maximum matching of $D - e$ is shown below:

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 $M_0^{'}$ if $e \in M_0$ and $e \notin M_0^{'}$; or it is smaller than M_0 by one in cardinality if e is in all maximum matchings of D.*

Proof. If $e \notin M_0$, removing e does not influence M_0. Thus, M_0 is still a maximum matching of $D-e$. If $e \in M_0$, and $e \notin M_0^{'}$. After removing it, $M_0^{'}$ would not be influenced, and $M_0^{'}$ is thus a maximum matching of $D-e$. If e is in all maximum matchings of D, matching M_0-e is obtained. Assume M_0-e is not maximal, and a matching with cardinality $|M_0|$ exists, it means removal of e can not influence a maximum matching of D, while e is in all maximum matchings of D is contradicted. Thus $M_0 - e$ could be a maximum matching of $D - e$. □

Corollary 1. *In $D = (V, E)$ of definition 6, by theorem 3, theorem 2, any $e \in E$ is a critical edge if $e \in M_0$ and $e \in M_0^{'}$; or an ordinary category if $e \in M_0$ and $e \notin M_0^{'}$; or a redundant category if $e \notin M_0$ and $e \notin M_0^{'}$.*

Proof. If $e \in E$ is in all maximum mathcings of D, by theorem 3, its removal leads M_0-e as a maximum matching of $D-e$. By theorem 2, the minimum number of inputs of $D-e$ is increased by one, and e is thus a critical edge. If $e \in M_0$ and $e \notin M_0^{'}$, by theorem 2, 3, removal of e does not influence $M_0^{'}$, which constructs a control configuration in $D - e$ with the same minimum number of inputs as before. Thus, e is an

ordinary edge. If 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, a single edge classification can be modelled into checking the number of maximum matchings having it. However, finding all maximum matchings of D is quite massive. Because any two maximum matchings are vertex-adjacent, $M_0^{'}$ can be derived by edge replacement into M_0, where edges out of M_0 replacing edges of M_0 to construct $M_0^{'}$ are called the maximally-matchable edges with respect to M_0 [7]. With maximally-matchable edges, we conclude:

Corollary 2. *By corollary 1, in $D = (V, E)$ of definition 6, with $e \in M_0$ and $M_0^{'}$, if e is not adjacent to any maximally-matchable edge related to M_0, $e \in M_0^{'}$, and e is a critical edge; otherwise, with $e \notin M_0^{'}$, while e and all maximally-matchable edges are in ordinary category.*

Proof. In $D = (V, E)$, because the maximally-matchable edges related to M_0 are the arcs in any $M_0^{'}$ but excluded by M_0, if $e \in M_0$ is not adjacent to any maximally-matchable edge, $e \in M_0^{'}$ and e is in all maximum matchings of D. By corollary 1, e is a critical edge. If e is adjacent to a maximally-matchable edge related to M_0 and $e \notin M_0^{'}$, removing either one of them, $M_0^{'}$ or M_0 can not be influenced and they are two ordinary edges by corollary 1 and theorem 3. □

By corollary 2, arcs neither in M_0 nor the maximally-matchable are redundant links and classifying all arcs of D into critical, redundant or ordinary category can be solved by finding maximally-matchable edges related to M_0.

B. Maximally-matchable Edges

We map D into a bipartite graph, noted by $B = (V_B, E_B)$ to find all maximally-matchable edges with respect to M_0.

Definition 7 ($B = (V_B, E_B)$)**.** *Given a bijection β and $D = (V, E)$ of definition 6, $B = (V_B, E_B)$ with $|E_B| = |E|$, $V_B = V_B^+ \cup V_B^-$ is obtained. For each $\overrightarrow{\langle v_i, v_j \rangle} \in E$, $\beta : \overrightarrow{\langle v_i, v_j \rangle} \rightarrow (v_i^+, v_j^-)$, where $(v_i^+, v_j^-) \in E_B$, $v_i^+ \in V_B^+$, $v_j^- \in V_B^-$. M_B is 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 edge of B with respect to M_B. Thus, we find all maximally-matchable edges of B with respect to M_B, which systematically are defined below:

Definition 8 (**Alternating Single 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 single 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 $(v_i^+, v_k^-) \in M_B$, replacing (v_i^+, v_k^-) with (v_i^+, v_j^-) produces a maximum matching: $M_B \setminus (v_i^+, v_k^-) \cup (v_i^+, v_j^-)$. Similarly, if $v_j^- \in M_B$, $v_i^+ \notin M_B$, a maximum matching would be also 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 $M_B \setminus (v_i^+, v_k^-) \cup (v_i^+, v_j^-)$, where $v_j^- \notin M_B$. By definition 8, (v_i^+, v_j^-) is an alternating link with respect to M_B. □

Definition 9 (**Alternating Cycle**). *In $B = (V_B, E_B)$ of definition 7, with $\{m_1, m_2, \ldots, m_t\} \subseteq M_B$, a matching set $\{e_1, e_2, \ldots, e_t\} \nsubseteq M_B$ $(1 < t \leq |M_B|)$ is an alternating cycle, if $\{m_1, e_1, m_2, e_2, \ldots, m_t, e_t\}$ is a cycle alternatively involving edges of and out of M_B.*

Definition 10 (**Alternating Path**). *In $B = (V_B, E_B)$ of definition 7, with $\{m_1, m_2, \ldots, m_t\} \subseteq M_B$, a matching set $\{e_1, e_2, \ldots, e_t\} \nsubseteq M_B$ $(1 < t \leq |M_B|)$ is an alternating path if $\{m_1, e_1, m_2, e_2, \ldots, m_t, e_t\}$ is a path alternatively involving edges of and out of M_B.*

Lemma 1. *In $B = (V_B, E_B)$ of definition 7, both alternating cycle and alternating path related to M_B are maximally-matchable edge sets.*

Proof. An alternating cycle $\{e_1, e_2, \ldots, e_t\} \nsubseteq M_B$ $(1 < t \leq |M_B|)$, replaces $\{m_1, m_2, \ldots, m_t\} \subseteq M_B$ that are adjacent to it, can obtain a maximum matching: $M_B \setminus \{m_1, m_2, \ldots, m_t\} \cup \{e_1, e_2, \ldots, e_t\}$. With respect to M_B, any alternating path in B can also construct a different maximum matching from M_B by replacing edges of M_B that are adjacent to them. □

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. Then, some properties of alternating paths and cycles are deduced:

Theorem 5. *In $B = (V_B, E_B)$ of definition 7, any minimal maximally-matchable edge set related to M_B is classified into either an alternating cycle, or an alternating path.*

Proof. By lemma 1, alternating cycle and path are maximally-matchable edge sets. Also, they are minimal maximally-matchable edge sets. because any $e_i (1 \leq i < t, 1 \leq t \leq |M_B|)$ of an alternating cycle or path is adjacent to two edges of M_B, either e_i or $\{e_1, e_2, \ldots, e_{t-1}\}$ can not construct a different maximum matching from M_B by edge replacement after removing e_i or e_t.

Assuming a minimal maximally-matchable edge set is neither alternating cycle nor path. In one aspect, if its edges are all among nodes of M_B, it should be 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. In other aspect, if such set is incident to nodes out of M_B, and it is still minimal, there should be only one alternating link of

definition 8, while remaining edges are not only among nodes of M_B but also not an alternating cycle. As a result, such edge set can be only an alternating path. □

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

Proof. Assuming a node shared by two alternating paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$, respectively exists. And this shared node is involved into a path, which alternatively involves edges of and out of M_B and has more edges out of M_B than that of M_B. However, a matching bigger than M_B in cardinality emerges, which contradicts with the maximality of M_B. Thus, any two alternating paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$ respectively, must be nonadjacent. □

Theorem 7. *In $B = (V_B, E_B)$ of definition 7, any two distinct alternating paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$ respectively, are not adjacent to a same alternating cycle.*

Proof. By definition 9,10, an alternating cycle can be adjacent to an alternating path. Assuming an alternating cycle is adjacent to two distinct alternaitng paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$ respectively. From two shared edges by these two distinct alternating paths and the alternating cycle, a path between v_i^+ and v_j^- alternatively involving edges of and out of M_B would exist, which is bigger than M_B in cardinality. Thus, an contradiction exists and any two distinct alternating paths incident to $v_i^+ \notin M_B$ and $v_j^- \notin M_B$can not be adjacent to a same alternating cycle. □

V. Execute entire edge classification

All maximally-matchable links of $B = (V_B, E_B)$ of definition 7 are essential for entire edge classification by corollary 2. To find them, we use a digraph derived from B. In detail, this digraph is noted by $D^{'} = (V^{'}, E^{'})$, where $E^{'}$ and $V^{'}$ are initially empty. Given $B = (V_B, E_B)$ of definition 7, any edge of $\{E_B - M_B\}$ is directed from V_B^+ to V_B^-. With a bijection ω, for any $m_i \in M_B (1 \leq i \leq |M_B|)$, $\omega : m_i \rightarrow u_i$, where u_i is a vertex, and we define that any $u_i \in S_c$, $S_c \subseteq V^{'}$. After direction and mapping operations, all nodes and edges existing are added into $V^{'}$ and $E^{'}$ respectively, leading $|E^{'}| = |E_B - M_B|$. We also define that $E^{'} = \{e_i^{'} | 1 \leq i \leq |E^{'}|\}$ and $V^{'} = \{v_i^{'} | 1 \leq i \leq |V^{'}|\}$.

With respect to M_B, any alternating single link of B is related to an arc in $D^{'}$, while any alternating path and cycle of B is related to a directed path and directed cycle in $D^{'}$. Next, we find all maximally-matchable edges with respect to M_B in $D^{'}$ by following algorithms.

A. Find arcs related to alternating links

Algorithm 1 finds arcs of $D^{'}$ related to alternating single links. We define S_a, S_b as two sets of returned arcs. $Adj(v_i^{'})$ denotes a set of nodes adjacent to $v_i^{'}$, and any node of $Adj(v_i^{'})$ is noted by $v_k^{'}$.

Proof. Initially, labelling nodes of S_c is in $O(|V^{'}|)$ time to know if a node is in or out of S_c rather than searching it in S_c. Firstly, a node $u_i \in S_c$ is chosen, then, **for** loop considers each adjacent node of it. If $v_k^{'}$ is out of S_c, arc involving $v_k^{'}$ and u_i is mapped by an alternating single link of B by definition 8, which is added into S_a or S_b. If $\nexists v_k^{'} \in Adj(u_i)$, **for** loop terminates and u_i is removed 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, where S_a and S_b containing arcs related to alternating single links of B are returned. For the worst case running time, since choosing all nodes of S_c takes $O(|V^{'}|)$ time, and each adjacent node of any $u_i \in S_c$ is examed once only, examing all adjacent nodes of all nodes of S_c cost $O(|E^{'}|)$ time. Thus, time complexity is $O(|V^{'}| + |E^{'}|)$ excluding deriving $D^{'}$ and S_c. □

Algorithm 1 Find arcs related to alternating single links

Input: $D^{'} = (V^{'}, E^{'})$, S_c
Output: Arcs of $D^{'}$ related to alternating single links of B

```
Label nodes of S_c
S_a = ∅; S_b = ∅
while S_c ≠ ∅ and u_i ∈ S_c do
    for Adj(u_i) ≠ ∅ and v_k' ∈ Adj(u_i) do
        Adj(u_i) = Adj(u_i) − v_k'
        if v_k' out of S_c and ⟨v_k', u_i⟩ then
            S_a = S_a + ⟨v_k', u_i⟩
        else if v_k' out of S_c and ⟨u_i, v_k'⟩ then
            S_b = S_b + ⟨u_i, v_k'⟩
    S_c = S_c − u_i
return S_a; S_b
```

B. Find arcs related to edges of alternating Paths and cycles

Algorithm 2 traverses directed paths of $D^{'}$ to find arcs related to edges of all alternating paths and some cycles of B. We define an arc of S_a by $e_i^{'}$, $e_i^{'} \in S_a$. We define P_0 as an arc set and $P(P_0)$ as 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 2 Find edges of alternating paths and cycle via $D^{'}$

Input: $D^{'} = (V^{'}, E^{'})$, S_a
Output: Arcs of $D^{'}$ related to alternating paths, cycles of B

```
while S_a ≠ ∅ and e_i' ∈ S_a do
    P_0 = ∅
    E' = E' − e_i'; P_0 = P_0 + e_i'
    for P(P_0) ≠ ∅ and e_j' ∈ P(P_0) do
        P_0 = P_0 + e_j'; E' = E' − e_j'
    return P_0
```

Proof. By definition 10, any alternating path in $B = (V_B, E_B)$ of definition 7 contains an alternating single link, and mapps into an directed path of $D^{'} = (V^{'}, E^{'})$ starting from an arc of S_a. Therefore, this procedure finds arcs pointed by each arc of S_a through directed paths. Firstly, any $e_i^{'} \in S_a$

is chosen, added into P_0. Then, the **for** loop finds all arcs pointed by $e_j^{'}$ through a directed path starting from $e_i^{'}$. If $P(P_0) \neq \emptyset$, e_i currently must point an arc $e_j^{'}$, which is added into P_0 to following search. Since $e_j^{'}$ is pointed by $e_i^{'}$ via a path, $e_j^{'}$ currently is related to an edge of an alternating path of B. If $P(P_0) = \emptyset$, all arcs pointed from $e_i^{'}$ have been traversed. Additionally, once an arc of $P(P_0)$ also points a node of an arc of P_0, a cycle is produced by it. Hence, P_0 contains the arcs of $D^{'}$ related to edges of alternating cyclse and 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 from another arc of S_a via paths as before. When $S_a = \emptyset$, due to edge removal from $E^{'}$, this procedure terminates. For time complexity except for precomputing S_a by algorithm V-B and obtaining $D^{'} = (V^{'}, E^{'})$, 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, time complexity is $O(|E^{'}|)$. □

According to theorem 6, 7, searching arcs of $D^{'}$ of alternating pahts and cycles of B from S_a does not influence searching that by S_b in $D^{'} = (V^{'}, E^{'})$, if $S_b \neq \emptyset$, algorithm V-B can be slightly modified to finding arcs of $D^{'}$ related to edges of alternating paths or alternating paths and cycles with respect to M_B in $O(|E^{'}|)$ time. In detail, $P(P_0)$ represents the arc set involving arcs out of P_0 and pointing arcs of P_0.

C. Search Arcs related to alternating cycles

By definition 9, any alternating cycle with respect to M_B of B is related to a directed cycle in $D^{'} = (V^{'}, E^{'})$, we thus find arcs of cycles of $D^{'}$ related to the alternating cycles with respect to M_B of $B = (V_B, E_B)$ by searching strongly connected components due to following theorem:

Theorem 8. *In $D^{'} = (V^{'}, E^{'})$, any arc of a strongly connected component must be involved into a directed cycle.*

Proof. Firstly, a strongly connected component is a component of a digraph whose every vertex can visit any others through a directed path [28] [29]. Assuming that an arc noted by $\overrightarrow{\langle v_i^{'}, v_j^{'}\rangle} \in E^{'}$ belongs to a strongly connected component, but it is out of any cycle. Then, assuming that any distinct node $v_k^{'}$ of a same component can be visited by $v_j^{'}$ through a directed path, while $v_k^{'}$ can not visit $v_i^{'}$ 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 of $D^{'} = (V^{'}, E^{'})$ can be effectively identified by using the well-known algorithm designed by Tarjan [29] in linear time, and each arc of the identified components will be returned in next algorithm.

Proof. By using the algorithm of [29], time complexity of identifying all strongly connected components is $O(|V^{'}| + |E^{'}|)$. Then, each arc of all found strongly connected components are labelled in $O(|E^{'}|)$ time, which is finally returned. Except for obtaining $D^{'} = (V^{'}, A^{'})$, time complexity of this procedure is $O(|V^{'}| + |E^{'}|)$. □

Algorithm 3 Find arcs mapped by alternating cycles

Input: $D^{'} = (V^{'}, A^{'})$
Output: Arcs of $D^{'}$ related to edges of alternating cycles of B
1: Find strongly connected components of $D^{'}$ by the algorithm of [29].
2: Label arcs of each identified strongly connected components.
3: **return** Each labelled arc

D. Entire edge classification of D = (V, E)

We now classify all arcs of $D = (V, E)$ of definition 6. We define a set involving all returned arcs by the algorithm V-A, V-B and V-C as $E_0^{'}$, and $E_0^{'} \subseteq E^{'}$. Besides, e_i is defined as any arc of M_0, where $e_i \in M_0$.

Algorithm 4 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: **return** D has no ordinary links; Arcs of M_0 are critical links; Arcs out of M_0 are redundant links.
3: **else if** $E_0^{'} \neq \emptyset$ **then**
4: Identify edges of E_B related to edges of $E_0^{'}$
5: Identify arcs of E via identified edges of E_B
6: **return** Identified arcs of E in line 5 are ordinary links.
7: Label each identified arcs of E
8: **for** $M_0 \neq \emptyset$ **and** $e_i \in M_0$ **do**
9: **if** e_i adjacent to an arc labelled **then**
10: **return** e_i is an ordinary link.
11: **else if** e_i not adjacent an arc labelled **then**
12: **return** e_i is a critical link.
13: $E = E - e_i$
14: **return** Arcs not labelled and of E are redundant links.

Proof. Initially, because $E_0^{'}$ collects all returned arcs by previous three algorithms, it can be obtained in $O(1)$ time after executing the algorithm V-A,V-B,V-C. 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 $E_0^{'} \neq \emptyset$, edges of B related to arcs of $E_0^{'}$ are identified in $O(|E_B|)$ time and those identified edges of B are also used to identify arcs of D by definition 7 also in $O(|E_B|)$ time, which are ordinary links by corollary 2. After this, identified edges of E are labelled in $O(|E|)$ time to find critical links of D. Specifically, each $e_i \in M_0$ is chosen to check if it is adjacent to a labelled edge by checking edges adjacent to it.

If so, $e_i \in M_0$ is an ordinary link; otherwise, it is a critical link. Then, e_i is removed from E. Because $M_0 \subseteq E$ and each chosen e_i is removed from E, **for** loop terminates when $M_0 = \emptyset$. Finally, unlabelled arcs of remaining E must be the redundant links. For the worst-case execution time of this algorithm, because each edge of M_0 is chosen once only and each labelled arcs would be checked twice at most in line 9. Above all, time complexity is $O(|E_B| + |E|)$ except for obtaining $E_0^{'}$, M_0 and $B = (V_B, E_B)$. □

E. Time complexity analysis

The worst-case execution time of entire edge classification is the sum of the time complexity of the algorithm from V-A to V-D except for precomputing M_0 of $D = (V, E)$ of defintion 6. By definition 7 and obtaining $D^{'}$, there are $|E_B| = |E|$, $2|V| \geq |V_B|$, and $|E^{'}| < |E_B|$, $|V^{'}| < |V_B|$. Besides, mapping D into B of definition 7 thus costs $\Theta(|E|)$ time, and then using B to obtain $D^{'} = (V^{'}, E^{'})$ also costs $\Theta(|E|)$ time. Also, time complexity of algorithm V-A, V-B, V-C and V-D, can be thus represented by $O(|V| + |E|)$, $O(|E|)$, $O(|V| + |E|)$ and $O(|E|)$ respectively. Eventually, in the worst case, classifying all arcs of $D = (V, E)$ into critical, redundant and ordianry categories respectively, is executed in $O(|V| + |E|)$ time excluding precomputation of M_0 of $D = (V, E)$.

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 one of 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—) algoithm 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.