Connectivity Determination Algorithm for Complex Directed Networks

Zhiyi Zhong, Lin Lin, Zhihan Jiang, Xin Yuan, *Senior Member, IEEE*, Edith Ngai, *Senior Member, IEEE*, James Lam, *Fellow, IEEE*, and Ka-Wai Kwok, *Senior Member, IEEE*

***Abstract*—Connectivity characterizes the ability of information transmission in systems modeled by complex networks. It is essential to develop an efficient connectivity determination algorithm with low time complexity and minimal storage requirements. To fulfill this need, a connectivity determination algorithm is designed by incorporating Tarjan's algorithm to identify strongly connected components and leveraging a depth-first search idea to traverse the reachability. This algorithm can ascertain strong connectivity, unilateral connectivity, and weak connectivity of complex directed networks. Besides, the accessibility matrix of complex directed networks is computed and visualized through an interface. As this algorithm relies on only two depth-first searches to accomplish connectivity determination tasks, its computational complexity does not exceed $O(n^2)$, where n denotes the number of network nodes. Experiments carried out on some specific networks reveal that the probability of network connections decreases with the increasing number of nodes in directed injective graphs, while in Erdős–Rényi graphs, the likelihood of connections increases as the number of nodes increases. Finally, a comparative example and an application example are provided to demonstrate the effectiveness of the algorithm program.**

***Index Terms*—Complex directed networks, connectivity, depth-first search, random graph, Tarjan's algorithm.**

I. Introduction

In graph theory, *networks* are graphs with weighted arcs, while *complex networks* are characterized by non-trivial topological features, distinguishing them from simple networks. These non-trivial topological features manifest in structural complexity, network evolution, connection diversity, dynamic complexity, node diversity, and meta-complication [1], [2]. Connection diversity encompasses directed arcs and varying weights [1], [3]. Specifically, complex networks with directed arcs are called *complex directed networks*. Complex directed networks find diverse applications across various domains, such as multi-agent systems [4]–[7], biological networks [8]–[13], the World Wide Web [14], [15], traffic dynamics [16], [17], and electrical power grids [18], [19].

Studying the connectivity structure, particularly focusing on strong connectivity, unilateral connectivity, and weak connectivity, plays a crucial role in the applications of complex directed networks [20], [21]. To elaborate, strong connectivity demands that every vertex in the directed graph possesses a directed path to each and every other vertex, unilateral connectivity mandates that each vertex in the directed graph has a directed path either to or from every other vertex, and weak connectivity stipulates that the underlying undirected graph is strongly connected [22].

The significance of determining strong, unilateral, and weak connectivity in complex networks can be exemplified by decentralized systems, leader-follower networks, and mobile computing networks, respectively. In decentralized systems such as multi-robot systems, robots collaborate to achieve globally defined objectives based on decentralized control instructions [23], indicating the absence of a central computing unit. The communication structure among these robots is often modeled as a complex directed network [24], where strong connectivity is crucial for ensuring high-precision information exchange to facilitate efficient cooperation among robotic systems [25]. The importance of unilateral connectivity can be exemplified in leader-follower multi-agent systems [26], where the communication network can also be conceptualized as complex directed networks. Unlike the decentralized system, strong connectivity may not always be obligatory in such scenarios, given the lesser need for bidirectional communication between leaders and followers. Instead, the focus often lies on unilateral connectivity from leaders to followers. Weak connectivity finds relevance in mobile computing [27]. With advancements in mobile communications, mobile servers encounter weakly connected networks with constraints like low bandwidth, high latency, or high expense. Therefore, effectively utilizing weak connectivity is essential for mobile file systems to ensure uninterrupted data access during temporary network or server outages, commonly referred to as disconnected operations [28].

Several algorithms are utilized to determine strong connectivity in complex networks. Initially, the depth-first search

This work was supported in part by the General Research Fund (GRF) under Grant 17201820; in part by the National Natural Science Foundation of China (NSFC) under Grant 62273286; in part by the Research Grants Council of Hong Kong under Grant STG1/E-401/23-N, Grant C4026-21G and Grant 17209021, and in part by the Innovation and Technology Commission (ITC) of the HKSAR Government under the InnoHK initiative, Hong Kong, via Centre for Garment Production Limited. (Zhiyi Zhong and Lin Lin are co-first authors.)

Corresponding authors: Lin Lin and James Lam.

Z. Zhong is with the Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong.

L. Lin is with the Department of Mechanical Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong (email: linlin00wa@gmail.com).

Z. Jiang and E. Ngai are with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Pokfulam, Hong Kong.

X. Yuan is with the School of Electrical and Electronic Engineering, University of Adelaide, Adelaide, SA 5005, Australia.

J. Lam is with the Department of Mechanical Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong, and also with the HKU Shenzhen Institute of Research and Innovation, Shenzhen 518057, China (email: james.lam@hku.hk).

K.-W. Kwok is with the Department of Mechanical Engineering, University of Hong Kong, Hong Kong, and also with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong.

(DFS) algorithm, which is essentially a recursive method for graph traversal, starts at the root node and explores as deeply as possible along a single path before backtracking [29]. Typically, DFS has a time complexity of $O(\max\{n,m\})$, determined by the larger order of magnitude between n and m, where n represents the number of nodes and m represents the number of arcs. However, since DFS needs to be executed on each node, the overall time complexity escalates to $O(n\max\{n,m\})$. Second, Tarjan's algorithm [29] or the Gabow algorithm [30], which builds upon DFS, has a time complexity of $O(\max\{n,m\})$. Specifically, Tarjan's algorithm, which is based on DFS, assigns a unique identifier to each node and computes low-link values to identify strongly connected components (SCCs) in a graph. Third, the Floyd Warshall algorithm, used to find the shortest paths between any two nodes in a directed network, determines node connectivity with a time complexity of $O(n^3)$. However, these algorithms do not account for unilateral connectivity.

Recent research [20] introduces an extended Warshall's algorithm to determine both strong and unilateral connectivity by creating an accessible matrix that records the connectivity between any node pairs within complex directed networks. While the integrated algorithm proposed in [20] for determining strong and unilateral connectivity maintains a high time complexity of $O(n^3)$, surpassing the aforementioned three algorithms, this study aims to explore an enhanced algorithm with reduced time and storage complexity. Furthermore, various studies have investigated the connectivity characteristics of complex networks, including the work by Broder *et al.* [31]. In their research, they utilized a large-scale web crawl to create a comprehensive representation of the entire web graph as a database, and then analyzed aspects such as macroscopic structure, diameter, connectivity, and degree distributions of the complex web network. Furthermore, they introduced Connectivity Server 2, a robust infrastructure featuring optimized data compression, a host database, and high-performance data access on a high-end machine equipped with sufficient Random-access Memory. While Broder *et al.* provide valuable insights into the connectivity structure of the web, their discussion of a BFS-based connectivity determination algorithm lacks emphasis on enhancing its time or space efficiency. This underscores the necessity for the development of more computationally efficient algorithms.

This paper develops a connectivity determination algorithm to assess strong, unilateral, and weak connectivity in complex directed networks. Its time complexity is reduced from $O(n^3)$ in [20] to a maximum of $O(n^2)$, by combining Tarjan's algorithm with the DFS algorithm. In addition, the memory storage demand has been reduced from $O(n^2)$ in [32] to a more efficient $O(n)$ by exclusively tapping into the DFS recursion stack, since the connectivity determination process does not require the accessible matrix. Furthermore, a "Connectivity Calculator for Complex Directed Networks" has been designed to expedite the process of determining connectivity and computing the accessible matrix for a specified complex directed network. Finally, connectivity analysis has been performed on several specific networks. It discovers that as the number of nodes increases in a directed injective graph, achieving connectivity becomes progressively more difficult. In contrast, a larger Erdős–Rényi graph model exhibits a higher propensity to establish connectivity even with exceedingly low edge probabilities.

The remainder of this paper is arranged as follows. Section II presents the system description and problem formulation. We then proceed in Section III to design the connectivity determination algorithm. In Section IV, an analysis of connectivity results for several common types of complex directed networks is provided. Finally, a comparative example and an application example are illustrated in Section V, while Section VI briefly concludes this paper.

II. Problem Formulation

This section will introduce some definitions associated with complex directed networks, and then formulate their strong and unilateral connectivity problems.

Complex directed network is defined as $\mathsf{G} := (\mathsf{V}, \mathsf{E})$, where $\mathsf{V} := \{v_1, v_2, \ldots, v_n\}$ is a node set, and $\mathsf{E} := \{(u,v) \mid u,v \in \mathsf{V}\}$ is an arc set. For $(u,v) \in \mathsf{E}$, u is called the *parent node* of v, while v is called the *child node* of u. For node $u \in \mathsf{V}$, its *in-degree*, denoted by $\mathsf{id}(u)$, is the number of nodes pointing to node u; its *out-degree*, denoted by $\mathsf{or}(u)$, is the number of nodes pointed by node u. Node $u \in \mathsf{V}$ is called *root node* if its in-degree is 0. Let $N_{\text{in}}(u)$ and $N_{\text{out}}(u)$, respectively, be the sets of parent nodes and child nodes for node u. If v_j can be reached from v_i via a series of nodes $(v_1, v_2, \ldots)$, then the sequence of nodes $(v_i, v_1, v_2, \ldots, v_j)$ is called a *path* from v_i to v_j. Besides, node v_j is called the *successor* of node v_i as well as the nodes $v_1, v_2, \ldots$, while node v_i is called the *predecessor* of node v_j as well as the nodes $v_1, v_2, \ldots$. Subsequently, the connectivity definition of G is presented as follows.

Definition 2.1 (See [20]): In G, if there exists a directed path from node $u \in \mathsf{V}$ to node $v \in \mathsf{V}$, then v is said to be reachable from u. If any two nodes in G are reachable to each other, G is said to be *strongly connected*. If, for any two nodes in G, there is at least one node reachable to the other, the network is said to be *unilaterally connected*. After converting the directed network G to an undirected network $\bar{\mathsf{G}}$, if any two nodes within $\bar{\mathsf{G}}$ are reachable to each other, then G is said to be *weakly connected*.

The interconnectedness of these network characteristics is indicated by the Venn diagram in Fig. 1. If a network is strongly connected, it is also unilaterally connected, weakly connected, and irreversible. Similarly, if a network is unilaterally connected, it is also weakly connected and irreversible.

Definition 2.2 (See [20]): For a directed network that is not strongly connected, its maximal subgraph where every pair of nodes is mutually reachable is called a *strongly connected component* (SCC) of this directed network.

Regarding complex directed network G, let $P := (p_{ij})_{n\times n}$ be its *accessible matrix*, defined as

$$p_{ij} = \begin{cases} 1, & v_j \text{ is reachable from } v_i, \\ 0, & v_j \text{ is not reachable from } v_i. \end{cases} \tag{1}$$

To minimize the additional space utilized, *linked lists* and *adjacency lists* are employed. A linked list is a data structure

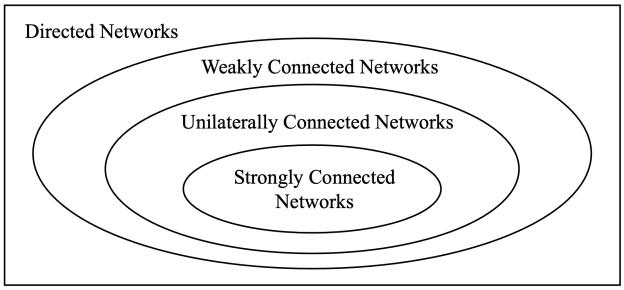


Fig. 1. Venn diagram showing the interrelationship among strongly connected, unilaterally connected, and weakly connected networks.

comprising a data field, like an integer, along with a reference to the next node in the list. On the other hand, an adjacency list depicts a graph using an array of linked lists. Each index of the array serves as the head of a linked list; each head signifies a node, and the corresponding linked list retains all adjacent nodes of that head node.

III. Connectivity Determination Algorithm

In this section, we develop algorithms with reduced computational complexity to consecutively determine the connectivity of complex directed network G, encompassing strong connectivity, unilateral connectivity, and weak connectivity.

A. Strong Connectivity

Here, we design an improved algorithm to determine the strong connectivity of G by combining Tarjan's algorithm and a DFS technique.

First, we use the DFS-based Tarjan's algorithm to find SCCs in G. For every node $u \in \mathsf{V}$, we define two values of the node stored in two arrays, $\mathsf{dfn}[u]$ and $\mathsf{low}[u]$, as follows: $\mathsf{dfn}[u]$ records the depth-first number of u, which is the order number of u to be reached in a DFS. DFS will traverse as far as possible along each branch until a node has no child node and then backtrack. Note that if a node has more than one child node, we assume it goes to the one with a lower serial number (i.e., in Fig. 2, both nodes 2 and 3 are child nodes of node 1, and we do DFS on node 2 first). In Fig. 2, if we start DFS from node 1 (in future examples, we all assume DFS starts from node 1), then it will traverse all the way down to node 2, node 4, and node 7. As node 7 does not have a child node, it will backtrack to node 4 and traverse to node 6. For node 6 also has no child node, and all child nodes of node 4 have been visited, it will backtrack to node 2 and reach node 5. Finally, it goes back to node 1 and visits node 3. As a result, one derives $\mathsf{dfn}[1]=1, \mathsf{dfn}[2]=2, \mathsf{dfn}[3]=7, \mathsf{dfn}[4]=3, \mathsf{dfn}[5]=6, \mathsf{dfn}[6]=5, \mathsf{dfn}[7]=4$.

To introduce $\mathsf{low}[u]$, we first define what a *back arc* is: in G, if the child node v of node u is also its predecessor node, the arc (u,v) is called a *back arc*. Then, $\mathsf{low}[u]$ records the node with the lowest depth-first number that node u can reach:

$$\mathsf{low}[u] = \min\Big\{ \mathsf{dfn}[u], \min\{\mathsf{low}[w] \mid w \text{ is a child node of } u\}, \min\{\mathsf{dfn}[v] \mid (u,v) \text{ is a back arc}\}. \Big\}$$

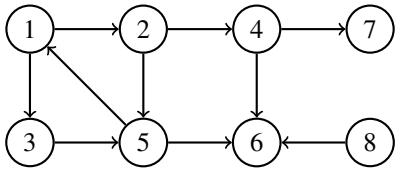


Fig. 2. An illustrative example of a directed network.

In Fig. 2, for node 5, it has $\mathsf{dfn}[5]=6$ but $\mathsf{low}[5]=1$ since it has a back arc to node 1, which has $\mathsf{dfn}[1]=1$. By resorting to $\mathsf{low}[u]$ and $\mathsf{dfn}[u]$, we design Algorithm 1 to calculate the SCCs and determine the strong connectivity of G.

Algorithm 1: Determine the strong connectivity and find all SCCs of G via Tarjan's algorithm.

Input: Node $u \in \mathsf{V}$.
Output: All the SCCs of G and an integer K as the number of SCCs.

```
dfn_counter = 0;                          ▷ dfn number counter
K = 0;                                    ▷ SCC counter
scc[MAX]; ▷ store nodes of each SCC into an array of linked list
for u ∈ V do
    if u is not visited then
        tarjan(u);
Function tarjan(u)
    label u as visited;
    dfn[u] = low[u] = ++dfn;   ▷ initially set low[u] to be the current dfn number
    Stack.push(u);
    for each v ∈ N_out(u) do
        if v is not visited then
            tarjan (v);
            low[u] = min(low[u], low[v]); ▷ update low[u]
        else
            if v in Stack then
                low[u] = min(low[u], dfn[v])   ▷ a back arc detected
    if dfn[u] == low[u] then
        K++;    ▷ an SCC detected and add one to K
        while u != v do
            v = Stack.pop;
            add v into scc[K];
    return;
```

In the initialization of Algorithm 1, we create $\mathsf{dfn\_counter}$ to keep track of the depth-first number during DFS, K to record the number of the SCCs in G, and array $\mathsf{scc}[\cdot]$ to store the group of nodes in each SCC. Based on the DFS, Tarjan's algorithm loops through nodes of the directed graph. For every node u that has not been visited, set $\mathsf{dfn}[u]$ to the temporary $\mathsf{dfn\_counter}$ number ($\mathsf{dfn\_counter}$ increases by 1

when visiting a new node), let the initial $\mathsf{low}[u]$ be equal to $\mathsf{dfn}[u]$, and push u into the stack.

Then, for every child node v of u, if v has been visited and is still in the stack, it turns out that (u,v) is a back arc as u reaches a lower dfn_counter value, and $\mathsf{low}[u]$ will be updated to $\mathsf{dfn}[v]$. Take Fig. 2 as an example, $\mathsf{low}[4]$ initially equals to 6 ($\mathsf{dfn}[4]=6$), but as $(6,1)$ is a back arc, $\mathsf{low}[4]=\mathsf{dfn}[1]=1$. If a child node v of u has not been visited, `tarjan(`v`)` will be executed. After the recursion, $\mathsf{low}[u]$ may be updated by comparing to $\mathsf{low}[v]$: $\mathsf{low}[v]$ is likely to be lower than $\mathsf{low}[u]$, because v may reach the lower node via a back arc. For instance, node 2 has $\mathsf{low}[2]=1$ in Fig. 2, since its child node 5 has an arc back to node 1.

After looping through all child nodes of node u, if $\mathsf{dfn}[u]=\mathsf{low}[u]$, an SCC is detected. To record the detection, 1 is added to the counter K. The top elements of the stack will be popped and added to the array $\mathsf{scc}[\mathsf{K}]$ in sequence until the top element is node u itself. By doing this, all nodes added as a whole will be recognized as an SCC. For example, during `tarjan(7)`, as $\mathsf{dfn}[7]=\mathsf{low}[7]=4$, then only node 7 would be popped from the stack (node 7 is the top node of the stack). Thus, $\{7\}$ will be recognized as an SCC.

In Algorithm 1, variable K counts the number of SCCs and determines whether G is strongly connected: if $\mathsf{K}>1$, network G is not strongly connected. Next, we reveal the effectiveness of Algorithm 1 to determine the strong connectivity of G.

Theorem 3.1: Complex directed network G is strongly connected if and only if Algorithm 1 outputs $\mathsf{K}=1$.

Proof: (Necessity.) Assume that G is strongly connected. Then, for every node $u\in\mathsf{V}$, $\mathsf{low}[u]=1$. It is because the low attribute records the lowest dfn number it could reach, while every node points to the root node with dfn of 1 due to strong connectivity. Hence, there will only be once (when u is the root node) such that $\mathsf{dfn}[u]==\mathsf{low}[u]$ and $\mathsf{K}++$ will only be triggered for once. As the initial value of K is 0, $\mathsf{K}=1$ under the circumstance that K is strongly connected.

(Sufficiency.) If $\mathsf{K}=1$, based on the operation of the algorithm, it implies that there is only one SCC in G. As SCC is the extremely large strong connected subgraph in a graph [22], it turns out that the entire G is strongly connected. ■

Algorithm 1 divides the original graph into several SCCs, from which we can construct a reduced network $\tilde{\mathsf{G}}=(\tilde{\mathsf{V}},\tilde{\mathsf{E}})$, where $\tilde{\mathsf{V}}=[1,\mathsf{K}]$ and $\tilde{\mathsf{E}}=\{(v,w),v,w\in\tilde{\mathsf{V}}\}$. As shown above, Fig. 2 can be reduced to Fig. 3 with 5 nodes: the original SCCs $C_1=\{1,2,3,5\}$, $C_2=\{4\}$, $C_3=\{6\}$, $C_4=\{7\}$, and $C_5=\{8\}$ in Fig. 2 are, respectively, nodes 1, 2, 3, 4, and 5 in Fig. 3.

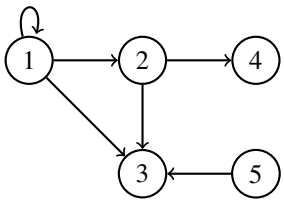


Fig. 3. An SCC-based directed network for the scale reduction of Fig. 2.

Next, accessible matrix can be established by the action of depth-first searching the reduced directed network $\tilde{\mathsf{G}}=(\tilde{\mathsf{V}},\tilde{\mathsf{E}})$.

Algorithm 2: Obtain accessible matrix $\tilde{P}=(\tilde{p}_{ij})_{n\times n}$ of the reduced directed network $\tilde{\mathsf{G}}$ via DFS.

Input: $\tilde{\mathsf{G}}=(\tilde{\mathsf{V}},\tilde{\mathsf{E}})$.
Output: Accessible matrix $\tilde{P}$.

```
P̃ ← 0_{K×K}.  ▷ set all entries of the accessible matrix to be 0
for ũ ∈ Ṽ do
    if id(ũ) == 0 then
        DFS(ũ);
Function DFS(ũ)
    Label ũ as visited;
    for ṽ ∈ N_out(ũ) do
        if ṽ not visited then
            R(ũ) = R(ũ) ∪ DFS(ṽ);  ▷ further do DFS on the unvisited node
        else
            R(ũ) = R(ũ) ∪ R(ṽ);  ▷ concatenate the reachable sets of ũ and ṽ
    for ṽ ∈ R(ũ) do
        p̃_{ṽũ} = 1;  ▷ ṽ is reachable from ũ
    return R(ũ).
```

The fundamental idea of Algorithm 2 is to loop through every node of $\tilde{\mathsf{G}}=(\tilde{\mathsf{V}},\tilde{\mathsf{E}})$ via the DFS. For every node $\tilde{u}$ whose in-degree is 0, set them as the root node of `DFS(`$\cdot$`)`. Then, for every child node $\tilde{v}$ of $\tilde{u}$, if $\tilde{v}$ has not been visited, proceed to `DFS(`$\tilde{v}$`)`. After the recursion, $\mathsf{R}(\tilde{u})$, which is the set of reachable nodes for $\tilde{u}$, can be updated to the union of sets $\mathsf{R}(\tilde{u})$ and $\mathsf{R}(\tilde{v})$ (during the recursion, $\mathsf{R}(\tilde{v})$ has been updated). Then, for every node $\tilde{v}$ in $\mathsf{R}(\tilde{u})$, it holds $\tilde{p}_{\tilde{v}\tilde{u}}=1$. The accessible matrix $\tilde{P}$ of the SCC-based network $\tilde{\mathsf{G}}$ could be established after all nodes have undergone the DFS. For example, the accessible matrix of the SCC-based network in Fig. 3 is

$$\tilde{P}=\begin{bmatrix}1&1&1&1&0\\0&0&1&1&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&1&0&0\end{bmatrix}.$$

Note that every node in $\tilde{\mathsf{G}}=(\tilde{\mathsf{V}},\tilde{\mathsf{E}})$ corresponds to exactly one SCC in G. Hence, if Algorithm 2 indicates that node $\breve{u}\in\tilde{\mathsf{G}}$ is reachable to node $\breve{v}\in\tilde{\mathsf{G}}$, then all the nodes in SCC $C_{\breve{u}}$ are reachable to the nodes in SCC $C_{\breve{v}}$. Thus, the accessible matrix $\tilde{P}$ can reveal the connectivity of both reduced network $\tilde{\mathsf{G}}=(\tilde{\mathsf{V}},\tilde{\mathsf{E}})$ and original network G. Based on $\tilde{P}$, the accessible matrix of the original graph is calculated as

$$P=\begin{bmatrix}1&1&1&1&1&1&1&0\\1&1&1&1&1&1&1&0\\1&1&1&1&1&1&1&0\\0&0&0&0&0&1&1&0\\1&1&1&1&1&1&1&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&1&0&0\end{bmatrix}.$$

Remark 3.1: The adjacency list serves as the input for Algorithm 1. As the algorithm progresses through the original network G, each node is visited, pushed into and popped from a stack once, and all arcs are traversed once. Consequently, the time complexity of Algorithm 1 is $O(\max\{|\,\mathsf{V}\,|,|\,\mathsf{E}\,|\})$, where $|\,\mathsf{V}\,|$ represents the number of nodes and $|\,\mathsf{E}\,|$ denotes the number of arcs. Assuming $|\,\mathsf{V}\,|=n$, then $|\,\mathsf{E}\,|$ is at most $n(n-1)$ in the case of a *directed complete graph*, where every pair of nodes u and v is linked by both (u,v) and (v,u). This results in a time complexity of $O(\max\{|\,\mathsf{V}\,|,|\,\mathsf{E}\,|\})=O(n^2)$. Likewise, Algorithm 2 traverses all nodes and arcs of the reduced network $\tilde{\mathsf{G}}=(\tilde{\mathsf{V}},\tilde{\mathsf{E}})$ with a computational complexity of $O(\max\{|\,\tilde{\mathsf{V}}\,|,|\,\tilde{\mathsf{E}}\,|\})$ = $O(\mathsf{K}^2)$. In the worst-case scenario where $\mathsf{K}=n$, the algorithm still traverses through n nodes, resulting in an overall time complexity of $O(n^2)$. Regarding storage complexity, in the worst case for Algorithm 1, all nodes are stored in both Stack and the recursion stack of the DFS, totaling $O(|\,\mathsf{V}\,|)$. For Algorithm 2, the storage complexity includes the recursion stack of the DFS and the storage space for the accessible matrix, amounting to $O(\mathsf{K}^2)$. It is notable that the general storage complexity of both Algorithm 1 and Algorithm 2 does not surpass $O(n^2)$. However, as the determination of strong connectivity relies solely on Algorithm 1, this task will only occupy space of no more than $O(n)$. In contrast, a recent study [20] introduces an algorithm with a time complexity of $O(n^3)$ for determining the connectivity of G, rendering it more time-intensive compared to our approach.

Algorithm 3: Determine the unilateral connectivity of complex directed network G.

Input: $\tilde{\mathsf{G}}=(\tilde{\mathsf{V}},\tilde{\mathsf{E}})$.
Output: Integer C as the number of unilateral connected pairs.

```
int C = 0;        ▷ the counter to record the number of unilateral connected pairs
for ũ ∈ Ṽ do
    if id(ũ) == 0 then
        DFS(ũ);
Function DFS(ũ)
    Label ũ as visited;
    for ṽ ∈ N_out(ũ) do
        if ṽ not visited then
            R(ũ) = R(ũ) ∪ DFS(ṽ);
        else
            R(ũ) = R(ũ) ∪ R(ṽ);
    Label ũ as used;
    for ṽ ∈ R(ũ) do
        if ṽ is not used || ũ ∉ R(ṽ) then
            C++;      ▷ C++ triggered after checking that ũ ∈ R(ṽ) has not been counted
    return R(ũ).
```

B. Unilateral Connectivity

Following the results on strong connectivity in Subsection III-A, one can further determine the unilateral connectivity of complex directed network G. If G is strongly connected, that is, $\mathsf{K}=1$, it is definite that this network is also unilaterally connected. Then, in this subsection, we will examine the unilateral connectivity of networks with $\mathsf{K}>1$.

Note that unilateral connectivity requires that, for any two nodes $i,j\in\mathsf{G}$, either i is in the reachable set of j or j is in the reachable set of i, that is, $i\in\mathsf{R}(j)$ or $j\in\mathsf{R}(i)$. Thus, to determine the unilateral connectivity, we could count the number of pairs of nodes in which at least one node is reachable to the other by improving Algorithm 2.

Based on Algorithm 2, Algorithm 3 initializes a global variable C at the beginning to count the number of unilaterally connected pairs. In addition, after establishing the reachable set of every node, we mark it as used. Then, at the end of each DFS recursion, we add one to C if, for every node $\tilde{v}\in\mathsf{R}(\tilde{u})$, $\tilde{v}$ is not labeled as used or $\tilde{u}\notin\mathsf{R}(\tilde{v})$ with $\tilde{v}$ having been labeled as used. To be specific, if $\tilde{v}$ is not used, we add one to C; if $\tilde{v}$ is used, it means that the path from $\tilde{v}$ to $\tilde{u}$ has triggered one addition to C, and then we check whether $\tilde{u}\in\mathsf{R}(\tilde{v})$ and add one to C if not. This part of the codes means that we count every two nodes in the SCC-based network $\tilde{\mathsf{G}}$ that are reachable to each other as one pair instead of two to avoid over-counting. Following this, we prove the effectiveness of Algorithm 3 in determining the unilateral connectivity of G.

Theorem 3.2: Complex directed network G is unilateral connected if and only if Algorithm 3 outputs $\mathsf{C}=\frac{\mathsf{K}^2-\mathsf{K}}{2}$.

Proof: (Necessity.) Assume that G is unilaterally connected, then the SCC-based graph $\tilde{\mathsf{G}}$ is also unilaterally connected as each node in $\tilde{\mathsf{G}}$ corresponds an SCC in G. During the execution of Algorithm 3, $\mathsf{C}++$ is triggered for every pair of nodes if one node is reachable to the other node or if both nodes are reachable to each other. Finally, as $\tilde{\mathsf{G}}$ is unilaterally connected, C increments for every pair of nodes, resulting in that C equals to the total number of node pairs in $\tilde{\mathsf{G}}$ as $(\mathsf{K}\times\mathsf{K}-\mathsf{K})\div 2=\frac{\mathsf{K}^2-\mathsf{K}}{2}$.

(Sufficiency.) After determining the value of C, we compare it with the number of nodes in the SCC-based network, K. Given that network $\tilde{\mathsf{G}}$ comprises K nodes, there exist $\frac{\mathsf{K}^2-\mathsf{K}}{2}$ pairs of nodes. If $\mathsf{C}=\frac{\mathsf{K}^2-\mathsf{K}}{2}$, both the reduced network $\tilde{\mathsf{G}}$ and the original network G are unilaterally connected. If not, it indicates that at least one pair of nodes in $\tilde{\mathsf{G}}$ or all nodes within two SCCs of G are not mutually reachable, negating the unilateral connectivity of both $\tilde{\mathsf{G}}$ and G. ■

Remark 3.2: The incremental operation "$\mathsf{C}++$" in Algorithm 3 takes constant time complexity of $O(1)$, thereby not impacting the overall time complexity of our algorithm for determining unilateral connectivity, which remains at $O(n^2)$.

Remark 3.3: In this paper, the main role of the accessible matrix is to depict the detailed connectivity among all nodes. It is not indispensable solely for determining the strong and unilateral connectivity of the complex directed network, as the former hinges on the value of K while the latter relies on C. In contrast to [20], Wang *et al.* utilizes the accessibility matrix for connectivity determination. Similarly, in [32], Zhu *et al.* presented a technique that integrates Tarjan's algorithm

and DFS algorithm to ascertain the controllability of Boolean control networks, which also necessitates the use of the accessibility matrix. Consequently, both approaches exhibit a space complexity of $O(n^2)$. However, if our method only focuses on determining strong and unilateral connectivity, Algorithm 1 and Algorithm 3 suffice, requiring only $O(n)$ extra space instead of $O(\max\{\mathsf{K}^2, n\})$ as noted in Remark 3.1. Hence, the space complexity is lower than $O\left(n^2\right)$ as in [20] and [32].

C. Weak Connectivity

In particular, if complex directed network G is unilaterally connected, it must be weakly connected. It is because unilateral connectivity guarantees that there is a directed path between any two nodes of G; after removing all directions, it is definite that any two nodes of the underlying undirected network $\bar{\mathsf{G}}$ are reachable to each other, which meets the requirement of weak connectivity. Then, for G that is neither strongly connected nor unilaterally connected, we improve Algorithm 1 to determine whether it is weakly connected.

In Algorithm 4, we initially transform the directed network G into the undirected network $\bar{\mathsf{G}}$. We loop through the node list V of G. For each node u, we then loop through its child nodes set, $N_{\text{out}}(u)$. If we find any v in $N_{\text{out}}(u)$ that lacks an arc back to u, we include u in $N_{\text{out}}(v)$ (i.e., adding an arc (v,u)).

Theorem 3.3: Complex directed network G is weakly connected if and only if Algorithm 4 outputs $\mathsf{W} = 1$.

Proof: After converting the directed network G to the undirected network $\bar{\mathsf{G}}$, we run Algorithm 1 again on $\bar{\mathsf{G}}$, while the difference is that we remove $\mathsf{scc}[\cdot]$ and the counter K, but add a new counter W to record the number of SCC in $\bar{\mathsf{G}}$. Thus, the proving process of Theorem 3.3 can be referred to as Theorem 3.1. ∎

Remark 3.4: The change of Algorithm 4 is a nested loop at the beginning: one to loop through all nodes and another to loop through all child nodes of each node. Thus, it will cause an extra time complexity of $O(n^2)$ and there will be no extra space used. However, as the remaining part is the same as Algorithm 1, the overall time complexity of Algorithm 4 still does not exceed $O(n^2)$, and the storage complexity is $O(n)$.

Algorithm 4: Determine the weak connectivity of complex directed network G.

Input: node $u \in \mathsf{V}$
Output: An integer W to determine the weak connectivity

```
dfn_counter = 0;
W = 0; ▷ the counter to record the number of SCC of Ḡ
for u ∈ V do
    if u is not visited then
        for each v ∈ N_out(u) do
            if u ∉ N_out(v) then
                add v into N_out(u); ▷ if there exists an arc (u,v), then add arc (v,u)
for u ∈ V do
    if u is not visited then
        label u as visited;
        tarjan(u);
Function tarjan(u)
    label u as visited;
    dfn[u] = low[u] = ++dfn_counter;
    Stack.push(u);
    for each v ∈ N_out(u) do
        if v is not visited then
            tarjan(v);
            low[u] = min(low[u], low[v]);
        else
            if v in Stack then
                low[u] = min(low[u], dfn[v])
    if dfn[u] == low[u] then
        W++;
        while u != v do
            v = Stack.pop;
    return;
```

IV. Connectivity In Some Specific Networks

Building upon the aforementioned algorithms, the corresponding Python implementation can be found at the following link: https://github.com/zzy1130/Connectivity-Determination-Algorithm-for-Complex-Directed-Networks.git. Furthermore, an interactive visualization interface, named "Connectivity Calculator for Complex Directed Networks", is developed to evaluate the network connectivity and generate the accessible matrix for any given complex directed network.

To use the tool, the user should provide the adjacency matrix of the network as an input, which indicates the presence of arcs between pairs of nodes. Upon inputting the matrix, users can click the "Generate Accessible Matrix" button to trigger the calculation process. The tool will then display the results, including the SCC node list, a representation of the reduced graph, the type of connectivity exhibited by the network, and

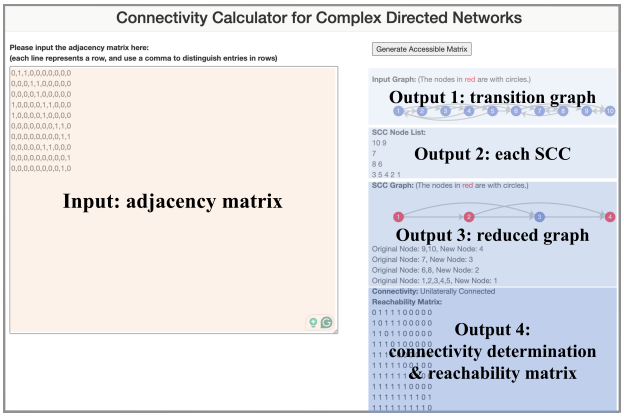


Fig. 4. Visualization interface of the connectivity determination tool.

the accessible matrix (also known as the reachability matrix) that depicts the reachability relationships among nodes.

Subsequently, experiments are carried out using this algorithm on diverse complex directed networks, encompassing a 1000-node directed network, directed injective graphs, and Erdős–Rényi graphs.

A. Connectivity in A Randomly Generated Network

Here, we evaluate the connectivity of a randomly generated complex directed network comprising 1000 nodes and 2000 edges. The network structure is visualized in Fig. 5, where the intensity of the nodal color is proportional to the degree of the node.

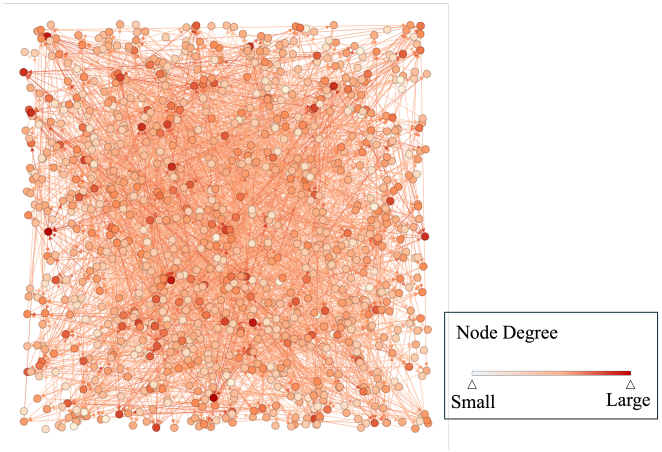


Fig. 5. A complex directed network $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ randomly generated with $|\mathsf{V}| = 1000$ and $|\mathsf{E}| = 2000$.

Following the execution of Algorithm 1, 405 SCCs are identified, indicating that the complex network is not strongly connected. Upon merging these SCCs into nodes, a reduced network is formed, a reduced network encompassing 405 nodes and 576 edges is formed, as illustrated in Fig. 6. Utilizing this reduced network, $\tilde{\mathsf{G}} = (\tilde{\mathsf{V}}, \tilde{\mathsf{E}})$, Algorithm 3 and Algorithm 4 ascertain that this network is neither unilaterally connected nor weakly connected.

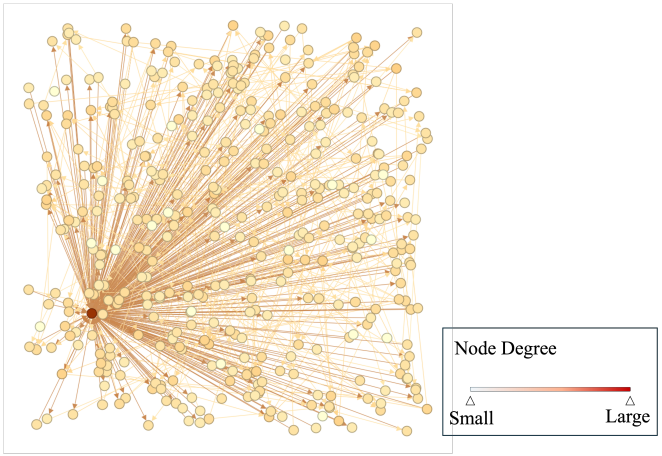


Fig. 6. The SCC-based network $\tilde{\mathsf{G}} = (\tilde{\mathsf{V}}, \tilde{\mathsf{E}})$ corresponding to complex directed network $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ in Fig. 5.

B. Connectivity in Directed Injective Graphs

Directed injective graphs are a specialized type of directed graphs where each node has precisely one outgoing edge. These graphs are commonly employed to depict one-to-one relationships within systems and are particularly useful in task scheduling. In this context, each task points to its succeeding dependent task, establishing a clear execution sequence. In computer science, network structures are utilized to illustrate state transitions in finite state machines, effectively capturing deterministic transitions between states through the use of directed injective graphs.

In this subsection, we conduct an experimental simulation to systematically explore and analyze the connectivity characteristics of directed injective graphs with varying numbers of nodes and edges. For each value of n ranging from 2 to 8, we generate all possible directed networks, where each node $u \in \{1, 2, \ldots, n\}$ is constrained to have only one outgoing edge, that is, $\mathsf{or}(u) = 1$ for all $u \in \{1, 2, \ldots, n\}$. As a result, there will be n^n possible configurations of directed injective graphs with n nodes. Subsequently, for each generated network, we evaluate its connectivity type, classifying it as strong, unilateral, weak, or unconnected. We then track the number of networks falling into each connectivity category for every value of n. The experimental results are visualized in Fig. 7.

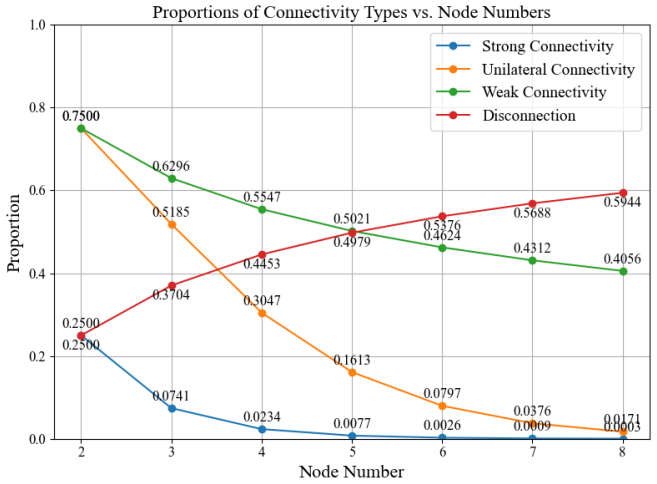


Fig. 7. Proportions of strongly connected, unilaterally connected, weakly connected and unconnected graphs among all directed injective graphs with n ranging from 2 to 8.

As the number of nodes increases, the proportion of weakly connected networks shows a gradual decline, the proportion of unilaterally connected networks exhibits a rapid decrease, and the proportion of strongly connected networks approaches zero but does not reach zero. Conversely, there is a significant increase in the count of unconnected networks. This trend suggests that the probability of network disconnection rises with more nodes, making it progressively more challenging for the directed injective graph to attain strong or unilateral connectivity.

C. Connectivity in Erdős–Rényi Model

The Erdős–Rényi model [33] is a fundamental framework in the realm of random graphs. In this model, a directed graph is formed by randomly connecting nodes: each edge is added to the graph with a probability of p, independently of any other edge. The Erdős–Rényi model finds diverse applications in various fields such as communication networks, social networks, power systems, and sensor networks. In communication networks, this model plays a crucial role in simulating the stochastic characteristics of network topologies. It aids in assessing signal transmission efficiency and network robustness [34]. When it comes to social network analysis, the Erdős–Rényi model is utilized to depict random connections among nodes. This representation helps in understanding the formation of network structures and dynamics in information propagation [35].

Hereafter, we conduct an analysis involving the generation of 1000 samples of Erdős–Rényi graphs across a spectrum of p values, ranging from low to high. Our investigation is centered on networks comprising 100 and 1000 nodes, with p values spanning from 0.001 to 0.2 for the 100-node networks and from 0.0001 to 0.02 for the 1000-node networks. Each dataset encompasses 200 unique p values. Subsequently, our developed algorithm calculates the number of SCCs in each generated graph and then determines the connectivity of the samples, recording the frequency of each connectivity category across 1000 samples for every p value. The outcomes of this analysis are illustrated in Fig. 8 and Fig. 9.

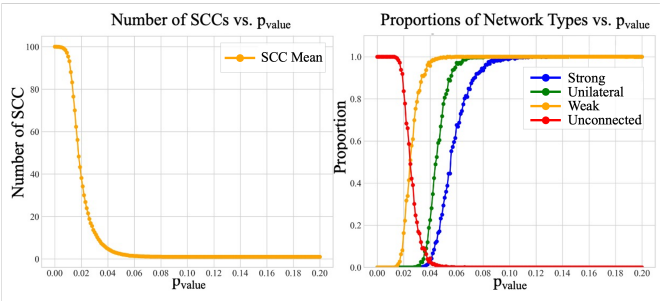


Fig. 8. Average number of SCCs and the proportions of strongly connected, unilaterally connected, weakly connected and unconnected graphs among 1000 Erdős–Rényi graphs with $n = 100$ and p ranging from 0.001 to 0.02.

In Fig. 8, for $n = 100$, several notable patterns are observed within the range of p values:

- The number of SCCs in the generated random graph begins to decrease at the p value of 0.01, aligning with the overall transition in connectivity across all types.
- The proportion of disconnected networks experiences a significant decrease within the p value range from 0.01 to 0.04. Concurrently, there is a decreasing trend in weakly connected networks within this range.
- A significant increase in unilateral connectivity is noted from 0.03 to 0.07, while strong connectivity shows a rise from 0.04 to 0.09.
- The growth rate of strong connectivity is slower compared to unilateral connectivity and even slower than that of weak connectivity.

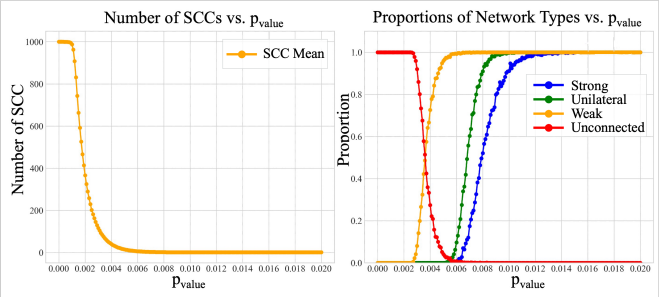


Fig. 9. Average number of SCCs and the proportions of strongly connected, unilaterally connected, weakly connected and unconnected graphs among 1000 Erdős–Rényi graphs with $n = 1000$ and p ranging from 0.0001 to 0.02.

From Fig. 9, it is evident that for $n = 1000$, the general trend of connectivity proportions mirrors the pattern observed for $n = 100$. However, critical points for the escalation of each connectivity type manifest at substantially lower p values. Notably, the proportion of disconnected networks commences a decrease within the p range from 0.0025 to 0.006. Subsequently, there is an uptick in unilateral connectivity from 0.006 to 0.01 and strong connectivity from 0.0063 to 0.012.

Comparing Fig. 8 and Fig. 9, it is apparent that the proportion of strongly connected networks asymptotically approaches 1 at an edge probability of $p = 0.10$ for $n = 100$. In contrast, for $n = 1000$, this convergence transpires at a markedly lower threshold of $p = 0.012$. This observation suggests that larger networks are more inclined to achieve connectivity even with exceedingly low edge probabilities.

V. Comparison and Application

In this section, to indicate the superiority and applicability of the obtained results, we apply our connectivity determination techniques to the numerical example in [20] and a biological example in [36].

A. A Conceptual Example

Here, we study the example in [20], whose transition graph G is shown in Fig. 10.

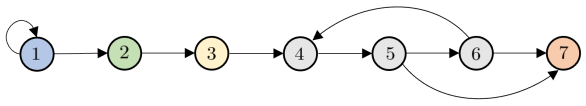


Fig. 10. Transition graph of seven-node directed network in [20].

By utilizing Algorithm 1, all SCCs are identified as follows with $\mathsf{K} = 5$:

$$C_1 = \{1\}, C_2 = \{2\}, C_3 = \{3\}, C_4 = \{4,5,6\}, C_5 = \{7\}.$$

In Fig. 10, nodes within the same SCC are depicted in the same color. This leads to the formation of the reduced directed network $\tilde{\mathsf{G}} = (\tilde{\mathsf{V}}, \tilde{\mathsf{E}})$ as shown in Fig. 11. Subsequently,

employing Algorithm 2, the accessible matrix of $\tilde{\mathsf{G}}$ can be constructed as

$$\tilde{P} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

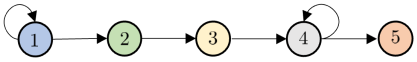


Fig. 11. Reduced graph of seven-node directed network in Fig. 10.

Meanwhile, utilizing Algorithm 3, C is determined to be 10. As $\mathsf{K} = 5 > 1$, in accordance with Theorem 3.1, G is not strongly connected. Since $\mathsf{C} = 10 = \frac{\mathsf{K}^2-\mathsf{K}}{2}$, according to Theorem 3.2, G is unilaterally connected. Furthermore, with the previous confirmation of unilateral connectivity for G, it is also inferred to be weakly connected.

For the connectivity determination method in [20], it first employs Warshall's algorithm to compute the accessible matrix of the whole graph as

$$P = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

This step requires $n^3 + n^2$ bit operations. The number of nonzero elements of the matrix is counted to check whether it is equal to n^2 so that the strong connectivity can be determined with n^2 bit operations. Thus, strong connectivity determination would cost $O(n^3)$. Besides, the unilateral connectivity algorithm in [20] includes calculating the number of non-zero elements in the accessible matrix (requiring $n^2 - n$ operations), sorting columns in ascending order using insertion sort (requiring $0.5n(n-1)$ operations), grouping columns based on equivalent values (requiring n operations), another insertion sort to sort rows of the matrix (requiring $0.5n(n-1)$ operations), calculating nonzero elements in the upper right corner of the resultant matrix and checking whether it is equal to $0.5n(n-1)$ (requiring $0.5n(n-1)$ operations). In total, its unified algorithm has a time complexity of $O\left(n^3\right)$ and a space complexity of $O\left(n^2\right)$.

Upon the comparison between the method in [20] and the established method here on the same numerical example, it could be found that our algorithm significantly reduces the time complexity in determining strong and unilateral connectivity of complex directed networks from $O\left(n^3\right)$ to $O\left(n^2\right)$. For space usage, as our method does not need the accessible matrix to determine strong and unilateral connectivity if not concerning about the connections among all nodes, space complexity is also reduced from $O\left(n^2\right)$ in [20] to $O(n)$ here. To better illustrate the comparison, an intuitive table is provided below.

TABLE I
COMPARISONS ON COMPUTATIONAL COMPLEXITY OF CONNECTIVITY DETERMINATION ALGORITHMS

	Time Complexity	**Space Complexity**
Method in [20]	$O(n^3)$	$O(n^2)$
Method here	$O(n^2)$	$O(n)$

B. A Biological Example

Consider the model of p53 response to DNA damage [37]. Fundamentally, this model revolves around the interactions between p53 and Mdm2. Here, the protein p53 responds to DNA damage by regulating the cell cycle and promoting DNA repair. The protein Mdm2 exists in two forms: cytoplasmic (Mdm2c) and nuclear (Mdm2n). Protein p53 promotes the accumulation of Mdm2c, which then translocates to the nucleus and impedes the export of Mdm2n. Conversely, Mdm2n facilitates the degradation of p53 through ubiquitination, regulating p53 levels to prevent excessive activity. DNA damage influences the degradation of Mdm2 in the nucleus, whereas p53 counteracts DNA damage signals by endorsing DNA repair mechanisms. The state transition graph of this model [36] can be depicted by complex directed network G as in Fig. 12, where each node symbolizes a model state, and each arc signifies a state transition.

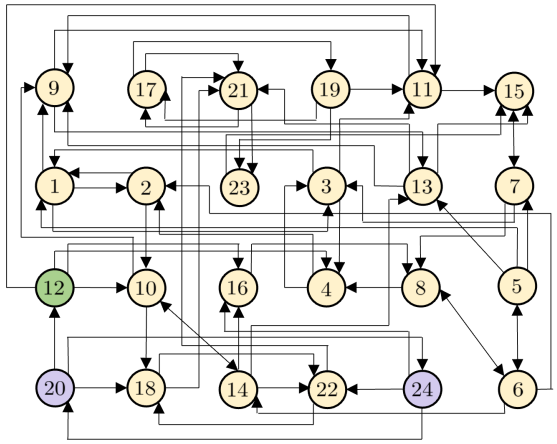


Fig. 12. State transition graph of the model on p53 response to DNA damage.

Based on Algorithm 1, all the SCCs in G can be identified as follows:

$$C_1 = \{20, 24\}, C_2 = \{12\}, C_3 = \mathsf{V} \backslash (C_1 \cup C_2),$$

where nodes of the same color in Fig. 12 belong to the same SCC. It derives the reduced directed network $\tilde{\mathsf{G}} = (\tilde{\mathsf{V}}, \tilde{\mathsf{E}})$ as shown in Fig. 13, which consists of only three nodes.

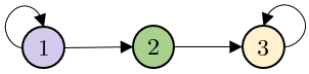


Fig. 13. Reduced complex directed network $\tilde{\mathsf{G}} = (\tilde{\mathsf{V}}, \tilde{\mathsf{E}})$.

Then, by executing Algorithm 2, we derive the accessibility matrix of $\tilde{\mathsf{G}}$ as

$$\tilde{P} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Through Algorithms 1 and 3, or by analyzing the accessibility matrix $\tilde{P}$, it can be inferred that this biological network is not strongly connected, but it is unilaterally connected and weakly connected.

VI. Conclusion and discussion

In this paper, a connectivity determination algorithm has been designed to determine the strong connectivity, unilateral connectivity, and weak connectivity of complex directed networks. An accessible matrix has also been calculated. It is worth pointing out that our proposed algorithm has greatly reduced the time complexity from $O(n^3)$ in [20] to not exceeding $O\left(n^2\right)$. Besides, the space complexity of our algorithm depends on the maximum size of the recursion stack and the size of the accessible matrix. In the worst-case scenario, where all nodes are within the stack and the number of SCCs equals the number of nodes, the space complexity amounts to $O\left(n^2\right)$. Importantly, the connectivity determination process only incurs a space complexity of $O\left(n\right)$. Using the developed connectivity determination algorithm, a series of experiments and discussions have been carried out to investigate the proportions of strongly connected, unilaterally connected, weakly connected and unconnected samples within all generated directed injective graphs or Erdős–Rényi models as the number of nodes increases. Finally, we have provided a numerical example and a biological example to elucidate the connectivity determination procedure.

References

[1] S. H. Strogatz, "Exploring complex networks," *Nature*, vol. 410, no. 6825, p. 268–276, 2001.

[2] S. Zhu, J. Cao, L. Lin, L. Rutkowski, J. Lu, and G. Lu, "Observability and detectability of stochastic labeled graphs," *IEEE Transactions on Automatic Control*, vol. 68, no. 12, pp. 7299–7311, 2023.

[3] J. Zhang, J. Lu, M. Xing, and J. Liang, "Synchronization of finite field networks with switching multiple communication channels," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 3, pp. 2160–2169, 2021.

[4] L. Lin, J. Cao, J. Lam, S. Zhu, S.-i. Azuma, and L. Rutkowski, "Leader-follower consensus over finite fields," *IEEE Transactions on Automatic Control*, vol. 69, no. 7, pp. 4718–4725, 2024.

[5] J. J. Liu, J. Lam, and K.-W. Kwok, "Positive consensus of fractional-order multiagent systems over directed graphs," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 11, pp. 9542–9548, 2022.

[6] Y. Cheng, "Fully distributed event-triggered output synchronization of heterogeneous multi-agent systems over directed switching networks," *Journal of the Franklin Institute*, vol. 359, no. 4, pp. 1706–1723, 2022.

[7] L. Wang, Z.-G. Wu, and J. Lam, "Necessary and sufficient conditions for security of hidden markov Boolean control networks under shifting attacks," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 1, pp. 321–330, 2022.

[8] L. Lin, J. Cao, J. Lam, L. Rutkowski, G. M. Dimirovski, and S. Zhu, "A bisimulation-based foundation for scale reductions of continuous-time Markov chains," *IEEE Transactions on Automatic Control*, vol. 69, no. 9, pp. 5743–5758, 2024.

[9] X. Ding, J. Lu, and X. Chen, "Stabilization of logical dynamic networks via event-triggered switching signals and its application," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 6, pp. 4393–4402, 2022.

[10] F. Martins Lopes, R. Marcondes Cesar Junior, and L. da F. Costa, "Gene expression complex networks: Synthesis, identification, and analysis," *Journal of Computational Molecular Cell Biology*, vol. 18, no. 10, pp. 1353–67, 2011.

[11] S. Zhu, J. Lu, S.-i. Azuma, and W. X. Zheng, "Strong structural controllability of Boolean networks: Polynomial-time criteria, minimal node control, and distributed pinning strategies," *IEEE Transactions on Automatic Control*, vol. 68, no. 9, pp. 5461–5476, 2023.

[12] S. Zhu, J. Cao, L. Lin, J. Lam, and S.-i. Azuma, "Toward stabilizable large-scale Boolean networks by controlling the minimal set of nodes," *IEEE Transactions on Automatic Control*, vol. 69, no. 1, pp. 174–188, 2024.

[13] L. Lin and J. Lam, "Observability categorization for Boolean control networks," *IEEE Transactions on Network Science and Engineering*, vol. 11, no. 1, pp. 1374–1386, 2024.

[14] A. Scharnhorst, "Complex networks and the web: Insights from non-linear physics," *Journal of Computer-Mediated Communication*, vol. 8, no. 4, p. JCMC845, 2003.

[15] Z. Zhao, Y. Mao, Y. Liu, L. Song, Y. Ouyang, X. Chen, and W. Ding, "Towards efficient communications in federated learning: A contemporary survey," *Journal of the Franklin Institute*, vol. 360, no. 12, pp. 8669–8703, 2023.

[16] S. Chen, W. Huang, C. Cattnai, and G. Altieri, "Traffic dynamics on complex networks: A survey," *Mathematical Problems in Engineering*, vol. 2012, pp. 1–23, 2012.

[17] A. Babu, T. Kavitha, R. P. de Prado, B. D. Parameshachari, and M. Woźniak, "Hopav: Hybrid optimization-oriented path planning for non-connected and connected automated vehicles," *IET Control Theory & Applications*, vol. 17, no. 14, pp. 1919–1929, 2023.

[18] L. Cuadra, S. Salcedo-Sanz, J. Del Ser, S. Jiménez-Fernández, and Z. Woo Geem, "A critical review of robustness in power grids using complex networks concepts," *Energies*, vol. 8, no. 9, pp. 9211–9265, 2015.

[19] Q. Su, S. Li, Y. Gao, X. Huang, and J. Li, "Observer-based detection and reconstruction of dynamic load altering attack in smart grid," *Journal of the Franklin Institute*, vol. 358, no. 7, pp. 4013–4027, 2021.

[20] Z. Wang, Y. Wu, Y. Xu, and R. Lu, "An efficient algorithm to determine the connectivity of complex directed networks," *IEEE Transactions on Cybernetics*, vol. 52, no. 7, pp. 7164–7171, 2022.

[21] C. H. Comin, T. Peron, F. N. Silva, D. R. Amancio, F. A. Rodrigues, and L. d. F. Costa, "Complex systems: Features, similarity and connectivity," *Physics Reports*, vol. 861, pp. 1–41, 2020.

[22] L. Lovász, "Connectivity in digraphs," *Journal of Combinatorial Theory, Series B*, vol. 15, no. 2, pp. 174–177, 1973.

[23] W. Ren and N. Sorensen, "Distributed coordination architecture for multi-robot formation control," *Robotics and Autonomous Systems*, vol. 56, no. 4, pp. 324–333, 2008.

[24] L. Lin, J. Cao, S. Zhu, and P. Shi, "Minimum-time and minimum-triggering impulsive stabilization for multi-agent systems over finite fields," *Systems & Control Letters*, vol. 155, p. 104991, 2021.

[25] L. Sabattini, C. Secchi, and N. Chopra, "Decentralized estimation and control for preserving the strong connectivity of directed graphs," *IEEE Transactions on Cybernetics*, vol. 45, no. 10, pp. 2273 – 2286, 2015.

[26] L. Lin, J. Cao, X. Liu, G. Lu, and M. Abdel-Aty, "Cluster synchronization of finite-field networks," *IEEE Transactions on Cybernetics*, 2023.

[27] M. Satyanarayanan, "Mobile computing," *Computer*, vol. 26, no. 9, pp. 81–82, 1993.

[28] L. B. Mummert, M. R. Ebling, and M. Satyanarayanan, "Exploiting weak connectivity for mobile file access," *ACM SIGOPS Operating Systems Review*, vol. 29, no. 5, pp. 143–155, 1995.

[29] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 146–160, 1972.

[30] H. N. Gabow, "Path-based depth-first search for strong and biconnected components," *Information Processing Letters*, vol. 74, no. 3, pp. 107–114, 2000.

[31] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener, "Graph structure in the web," *Computer networks*, vol. 33, no. 1-6, pp. 309–320, 2000.

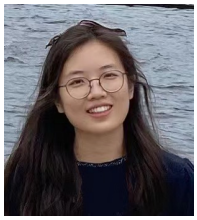
[32] Q. Zhu, Y. Liu, J. Lu, and J. Cao, "Further results on the controllability of Boolean control networks," *IEEE Transactions on Automatic Control*, vol. 64, no. 1, pp. 440–442, 2019.

[33] P. Erdős and A. Rényi, "On random graphs I," *Publicationes Mathematicae Debrecen*, vol. 6, no. 290-297, p. 18, 1959.

[34] D. S. Callaway, M. E. Newman, S. H. Strogatz, and D. J. Watts, "Network robustness and fragility: Percolation on random graphs," *Physical review letters*, vol. 85, no. 25, p. 5468, 2000.

[35] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.
[36] G. Stoll, E. Viara, E. Barillot, and L. Calzone, "Continuous time Boolean modeling for biological signaling: application of gillespie algorithm," *BMC Systems Biology*, vol. 6, no. 1, pp. 1–18, 2012.
[37] W. Abou-Jaoudé, D. A. Ouattara, and M. Kaufman, "From structure to dynamics: frequency tuning in the p53–mdm2 network: I. logical approach," *Journal of Theoretical Biology*, vol. 258, no. 4, pp. 561–577, 2009.

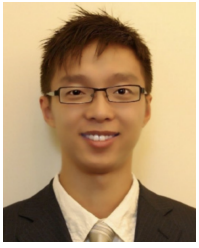
Zhiyi Zhong is currently an undergraduate student at the Department of Computer Science, The University of Hong Kong. He was a part-time Research Assistant at the Department of Mechanical Engineering, The University of Hong Kong, from September 2022 to December 2022 and from October 2023 to January 2024. His research interests include networked collective intelligence and control theory.



Lin Lin (S'21) is currently a Postdoctoral Fellow at the University of Hong Kong, Hong Kong. Prior to this, she held a Postdoctoral Fellow position at the Department of Electrical Engineering, City University of Hong Kong, Hong Kong. She received her Ph.D. degree in Engineering from the Department of Mechanical Engineering at The University of Hong Kong in 2024. She has visiting positions in Australia, England, and Japan. Her research interests include networked collective intelligence, logical networks, control theory, and reinforcement learning.

As a graduate student, Dr. Lin was a recipient of the IETI PhD Fellowship Award, the Lotfi Zadeh Best Paper Award Finalist, and the Outstanding Master Degree Thesis Award from the Chinese Institute of Electronics.

Zhihan Jiang (S'23) received the B.E. and M.E. degrees in computer science and technology from Xiamen University, Xiamen, China, in 2018 and 2021, respectively. She is currently pursuing the Ph.D. degree with the Department of Electrical and Electronic Engineering, The University of Hong Kong. Her research interests include Data Analytics and Visualization, Ubiquitous Computing, and Mobile Computing.



Xin Yuan (Senior Member, IEEE) received the B.E. (Hons.) and Ph.D. degrees from The University of Adelaide, Adelaide, SA, Australia, in 2016 and 2021, respectively. He is currently undertaking research with The University of Adelaide and working on discovering new approaches for the cognitive agent-based systems with artificial general intelligence, cognitive computational architectures, and AI-empowered engineering solutions for animal welfare and enrichment.

Edith C.H. Ngai is currently an Associate Professor in the Department of Electrical and Electronic Engineering, The University of Hong Kong. Before joining HKU in 2020, she was an Associate Professor in the Department of Information Technology, Uppsala University, Sweden. Her research interests include Internet-of-Things, edge intelligence, smart cities, and smart health. She was a VINNMER Fellow (2009) awarded by the Swedish Governmental Research Funding Agency VINNOVA. Her co-authored papers received a Best Paper Award in QShine 2023 and Best Paper Runner-Up Awards in ACM/IEEE IPSN 2013 and IEEE IWQoS 2010. She was an Area Editor of IEEE Internet of Things Journal from 2020 to 2022. She is currently an Associate Editor in IEEE Transactions of Mobile Computing, IEEE Transactions of Industrial Informatics, Ad Hoc Networks, and Computer Networks. She has served as a program chair in IEEE ISSNIP 2015, IEEE GreenCom 2022, and IEEE/ACM IWQoS 2024. She received a Meta Policy Research Award in Asia Pacific in 2022. She was selected as one of the N²Women Stars in Computer Networking and Communications in 2022. She is a Distinguished Lecturer in IEEE Communication Society in 2023-2024.

James Lam (Fellow, IEEE) received the B.Sc. (First Hons.) degree in mechanical engineering from the University of Manchester, Manchester, U.K., in 1983, and the M.Phil. in control and operational research and Ph.D. degree in control engineering from the University of Cambridge, Cambridge, U.K., in 1985 and 1988, respectively.

Prior to joining the University of Hong Kong, Pokfulam, Hong Kong, in 1993, where he is currently a Chair Professor of control engineering, he was a faculty member with the City University of Hong Kong and the University of Melbourne. His research interests include model reduction, robust synthesis, delay, singular systems, stochastic systems, multidimensional systems, positive systems, networked control systems, and vibration control.

Dr. Lam is a Croucher Scholar, Croucher Fellow, and Distinguished Visiting Fellow of the Royal Academy of Engineering, and a Cheung Kong Chair Professor. He is a Chartered Mathematician, Chartered Scientist, Chartered Engineer, Fellow of the Institute of Electrical and Electronic Engineers, Fellow of the Institution of Engineering and Technology, Fellow of the Institute of Mathematics and Its Applications, Fellow of the Institution of Mechanical Engineers, and Fellow of the Hong Kong Institution of Engineers. He is the Editor-in-Chief for *IET Control Theory and Applications*, *Journal of The Franklin Institute*, *Proc. IMechE Part I: Journal of Systems and Control Engineering*, *IET Journal of Engineering*, and *Franklin Open*, a Subject Editor for *Journal of Sound and Vibration*, an Editor for *Asian Journal of Control*, a Section Editor for a Consulting Editor for *International Journal of Systems Science*, an Associate Editor for *Automatica* and *Multidimensional Systems and Signal Processing*. He is a Highly Cited Researcher in Engineering from 2014 to 2020, Cross-Fields in 2021, and Computer Science in 2015. He is a Member of Academia Europaea, and an Academician of the International Academy of Systems and Cybernetic Sciences.

Ka-Wai Kwok (Senior Member, IEEE) received the B.Eng. and M.Phil. degrees from the Department of Automation and Computer-Aided Engineering [currently Mechanical and Automation Engineering (MAE)], The Chinese University of Hong Kong (CUHK), Hong Kong, in 2003 and 2005, respectively, and the Ph.D. degree from the Hamlyn Centre for Robotic Surgery, Department of Computing, Imperial College London, London, in 2012.

After then, he was awarded the Croucher Foundation Fellowship 2013, which supported his research jointly supervised by Advisors with The University of Georgia, Athens, GA, USA; and Brigham and Women's Hospital, Boston, MA, USA; and Harvard Medical School, Boston. He is currently a Professor with MAE, CUHK. Before then, he served as a Faculty for the Department of Mechanical Engineering, The University of Hong Kong (HKU), Hong Kong, from 2014 to 2024. He is also a Co-Founder and the Director of Agilis Robotics Ltd., Hong Kong, aiming at advancing the interventional endoscopy with small, flexible robotic instruments, and their intelligent control systems. He has participated in various designs of robotic devices/interfaces for endoscopy and MRI-guided interventions. To date, he has co-authored $>$165 peer-reviewed articles with $>$80 Clinical Fellows and $>$160 Scientists/Engineers. His research interests focus on surgical robotics, intra-operative image processing, and their uses of control and intelligent systems.

Dr. Kwok's multidisciplinary work has been recognized with various ($>$10) awards in international conferences/journals, such as the largest flagship conferences of robotics, ICRA, and IROS. He was the recipient of the ICRA Best Conference Paper Award in 2018 and the IROS Toshio Fukuda Young Professional Award in 2020. He also obtained awards in his early career for robotics, such as the Early Career Awards 2015/2016 offered by Research Grants Council of Hong Kong, the Actuators 2020 Young Investigator Award, the HKU 2019 and 2020 Outstanding Young Researcher Award, the HKU Young Innovator Award 2020, and the HKU Research Output Prize 2021 and 2022. Being IEEE Senior Member, he serves with editor boards for IROS 2017–22, ICRA 2019–23, IEEE Robotics and Automation Magazine, as well as Annals of Biomedical Engineering and Proceedings of the Institution of Mechanical Engineers—Part I: Journal of Systems and Control Engineering. He is the Principal Investigator of Group for Interventional Robotic and Imaging Systems, which has ($>$5) inventions licensed/transferred from University to industry in support for their commercialization.