Enumeration of Subtrees of 4-Cactus Networks

Qi Yao, Lixin Dong, Feng Li

***Abstract*—For a given network, determining its reliability is of great importance. Network reliability can be categorized into global reliability and local reliability. The number of spanning trees can be used to measure the global reliability under edge failures, while the number of subtrees serves as an indicator of local reliability under both edge and vertex failures. Therefore, the enumeration of subtrees is of significant value in network analysis and design. In this paper, we propose a linear-time algorithm for counting the number of subtrees in 4-cactus networks, based on edge-deletion and vertex-deletion contraction principles. Furthermore, we establish the upper and lower bounds for the number of subtrees in 4-cactus networks and characterize the extremal graphs that achieve these bounds. These results provide valuable insights for evaluating network-related indices and enhancing network reliability.**

***Index Terms*—cactus networks, the number of subtrees, graphs, algorithms**

I. Introduction

With the rapid development of technology and science, we are surrounded by various types of networks. In addition to the more easily understood computer and communication networks, many phenomena in our daily lives can also be abstracted into specific networks. As long as we identify the network nodes and the connections between them, we can derive the corresponding interconnected network. It is clear that a network can be viewed as a connection pattern between the components of a system. The structure of a network can be clearly represented by a graph, where the vertices represent the components in the system, and the edges represent the connections between these components. Additionally, edges can be classified as directed or undirected, leading to the distinction between directed graphs and undirected graphs. As networks play an increasingly important role in our daily lives, the analysis and design of networks have become of significant importance. Furthermore, many networks may experience failures due to the malfunction or failure of nodes and edges, which could lead to network malfunctions, and in severe cases, cause the entire network to fail. Therefore, the study of network reliability is of great practical significance and application value. Researchers have found that certain specific properties of networks can be used to measure their reliability. The number of spanning trees and the number of subtrees are two important properties that can be used to assess network reliability. Counting the spanning trees of a network can be used to evaluate its global reliability, while counting the subtrees can be used to study its local reliability. Compared with spanning tree enumeration, subtree counting involves more complex combinatorial constructions and a greater diversity of structural types. Therefore, the study of subtree enumeration is crucial for network reliability analysis and provides important guidance for network analysis and design.

Manuscript received May 26, 2025; revised July 25, 2025. This research was supported by the Research Foundation from Qinghai Normal University (Grant No. 2023QZR002).

Qi Yao is a postgraduate student at the College of Computer, Qinghai Normal University, Xining 810008, China (e-mail: 81750@163.com).

Lixin Dong is an associate professor at the College of Computer, Qinghai Normal University, Xining 810008, China (Corresponding author to provide phone: +8613897663081; e-mail: 2013040@qhnu.edu.cn).

Feng Li is a professor at the College of Computer, Qinghai Normal University, Xining 810008, China (e-mail: li2006369@126.com).

Let $G = (V(G), E(G))$ be a graph, where $V(G)$ is the vertex set and $E(G)$ is the edge set. Let $n = |V(G)|$ denote the number of vertices, and $m = |E(G)|$ denote the number of edges in G. We use P_n, C_n and K_n to denote the path, cycle, and complete graph on n vertices, respectively. A graph G is said to be connected if there exists a path between any pair of vertices in G. If $V(D) \subseteq V(G)$ and $E(D) \subseteq E(G)$, then the graph D is called a subgraph of G. For a connected graph G, if the removal of a vertex u and all edges incident to u results in a disconnected graph, then u is called a cut vertex of G. In graph theory, a connected graph without any cut vertices is called a block. A subgraph of G is called a block of G if it is a block itself and the addition of any vertex outside the subgraph results in a subgraph that contains a cut vertex. If $V(T) = V(G)$, $E(T) \subseteq E(G)$, and T is a tree, then T is called a spanning tree of G. Let $\tau(G)$ denote the number of spanning trees of G. All acyclic substructures of G are referred to as subtrees of G. Let $\eta(G)$ denote the total number of subtrees of G, $\eta_u(G)$ denote the number of subtrees of G that contain vertex u, and $\eta(G-u)$ denote the number of subtrees of G that do not contain vertex u.

Counting problems are one of the core research areas in combinatorial mathematics, and they have also become a popular topic in graph theory. Since subtree enumeration emerged as a research topic, many scholars have devoted considerable effort to it and have achieved a number of important results. Székely and Wang [1] were the first to study the number of subtrees in trees. They determined the maximum and minimum number of subtrees among all trees with n vertices and characterized the structures of the extremal trees. Yan and Yeh [2] investigated the problem of subtree enumeration in trees using generating functions, based on the idea of assigning weights to vertices and edges. They proposed three linear-time algorithms to compute the total number of subtrees, as well as the number of subtrees containing a specific vertex or a pair of vertices. Their work introduced a novel approach to subtree counting and also characterized the corresponding extremal trees. Székely and Wang [3] further investigated binary trees with the maximum number of subtrees and explored the relationship between the number of subtrees and the Wiener index. Kirk and Wang [4] studied the trees with the maximum number of subtrees under a given maximum degree constraint. Yang, Liu, and others [5] investigated the enumeration of BC-subtrees in trees and characterized the corresponding extremal structures. Dong et al. [6] proposed a linear-time algorithm for investigating the subtree enumeration problem in 3-cactus networks. They

established the upper and lower bounds of the number of subtrees, as well as the entropy of subtrees in 3-cactus networks. As an application, they further presented a linear-time algorithm for computing the number of subtrees in Koch networks. Zhang et al. [7] investigated several properties of the number of subtrees in trees with a given degree sequence. These properties were used to characterize trees with the maximum number of subtrees among all trees with the same degree sequence, and corresponding extremal results were provided. Xiao et al. [8] proposed two recursive relations to compute the number of subtrees in trees and proved that the number of subtrees in all trees with n vertices lies within a specific range. Sun et al. [9] studied subtree enumeration in planar two-tree networks. By introducing virtual edges and applying edge orientation techniques, they proposed two linear-time algorithms to compute the number of subtrees in planar two-tree and planar two-connected networks. As applications, they also provided subtree-counting formulas for Farey networks and GDURT networks. Sun et al. [10] investigated subtree enumeration in two types of self-similar networks. Using generating functions and dual transformations of two-forest, they solved the subtree enumeration problem for these networks and proposed two linear-time algorithms for computing their subtree generating functions. Horibe et al. [11] proposed two algorithms for computing all Characteristic Paths and Subtrees in Structurally Compressed Tree-Structured Data, and implemented both algorithms on a computer.

Cactus networks represent an important class of networks due to their unique structural properties. Numerous researchers have explored their theoretical characteristics and practical applications across various domains. Rautenbach et al. [12] introduced a general definition for cactus networks and focused on their domatic number. Furthermore, they characterized domatically full block-cactus graphs. Ben-Moshe et al. [13] studied the center problem in cactus networks and proposed two efficient algorithms for solving the 1-center and 2-center problems, respectively. Their work holds significant implications for facility location and optimization in networked systems. Liu et al. [14] investigated subsystem reliability based on cactus networks. By employing a probabilistic failure model and the principle of inclusion-exclusion, they derived approximations and upper bounds for the subsystem reliability of multiprocessor systems, considering intersections of up to three subsystems. Arcak et al. [15] studied the diagonal stability of cactus graphs and proposed necessary and sufficient conditions under which directed graphs satisfying the definition of a cactus network exhibit diagonal stability. Building upon their previous work in [14], Liu et al. further investigated subsystem-based reliability of cactus networks in [16]. Using a probabilistic failure model and the principle of inclusion-exclusion, they derived approximations and upper bounds for subsystem reliability by decomposing cactus-based networks into $(n-1)$-dimensional subsystems and fixing a pair of vertices. Ben-Moshe et al. [17] investigated the centdian problem in cactus networks and proposed new efficient sequential and distributed algorithms for identifying all centdian nodes in cycle graphs and cactus graphs. Cactus networks have significant application value in network analysis, design, and the development of high-reliability systems. When tree-like topologies are not suitable for network design, extending to cyclic topologies may enhance the performance of the network. This paper primarily investigates the subtree enumeration problem in 4-cactus networks. A linear-time algorithm is proposed to compute the number of subtrees in 4-cactus networks. Additionally, the minimum and maximum values of the subtree count are determined, and the corresponding extremal graph topologies are characterized.

II. Algorithm for Enumerating Subtrees of 4-Cactus Networks

In this section, we propose a linear-time algorithm for computing the number of subtrees in 4-cactus networks and prove its correctness. Before presenting the subtree enumeration algorithm for 4-cactus networks, we first introduce some important definitions and theorems.

Definition 1. [12] A graph is called a *cactus graph* if each of its blocks is either a cycle or a complete graph K_2. A cactus graph in which every block is a quadrilateral (4-cycle C_4) is called a *4-cactus graph*.

Definition 2. Let $U(t)$ denote the set of all 4-cactus networks containing t quadrilaterals (4-cycles C_4). The following recursive construction defines a 4-cactus network $G_t \in U(t)$:

- For $t = 0$, G_0 consists of a single vertex, referred to as the *initial vertex*.
- For $t = 1$, G_1 is a quadrilateral (4-cycle C_4), referred to as the *initial quadrilateral*. This quadrilateral is also referred to as the *initial square*.
- For $t \geq 1$, G_{t-1} is a 4-cactus network generated by $t-1$ iterations of the recursive construction. At the t-th iteration, a folded "7-shaped" path P_3 is added to the network, and both endpoints of P_3 are connected to a vertex in G_{t-1}. The resulting network G_t is referred to as a 4-cactus network.

For example, Figure 1 illustrates the recursive construction process of a 4-cactus network G_2.

If $V(D) \subset V(G)$ and $E(D) \subset E(G)$, then graph D is called a *proper subgraph* of graph G. Let (u_0, u_1, u_2, u_3) denote a quadrilateral (4-cycle C_4) with vertices u_0, u_1, u_2, u_3. For a positive integer $k \geq 3$, if in graph D, three of the four vertices u_0, u_1, u_2, u_3 in a quadrilateral have degree 2, and the fourth vertex has degree k, then this quadrilateral is called a *k-pendant quadrilateral*, or simply a *pendant quadrilateral*.

According to the definition of a 4-cactus network, when $t \geq 2$, any $G_t \in U(t)$ must contain a pendant quadrilateral, and the corresponding G_t contains at least one vertex of degree no less than 4. If G_t contains a 4-pendant quadrilateral as defined, then the 4-cactus network is said to belong to *Category A*. If G_t does not contain any 4-pendant quadrilateral, then the 4-cactus network G_t is said to belong to *Category B*. The following Theorem 3 presents some properties of 4-cactus networks.

Theorem 3. Let $t \geq 0$ be an integer, and let $G_t \in U(t)$ be a 4-cactus network generated through t iterations. Then the following conclusions hold:

(1) G_t contains t quadrilaterals (4-cycles C_4), $3t + 1$ vertices, and $4t$ edges.

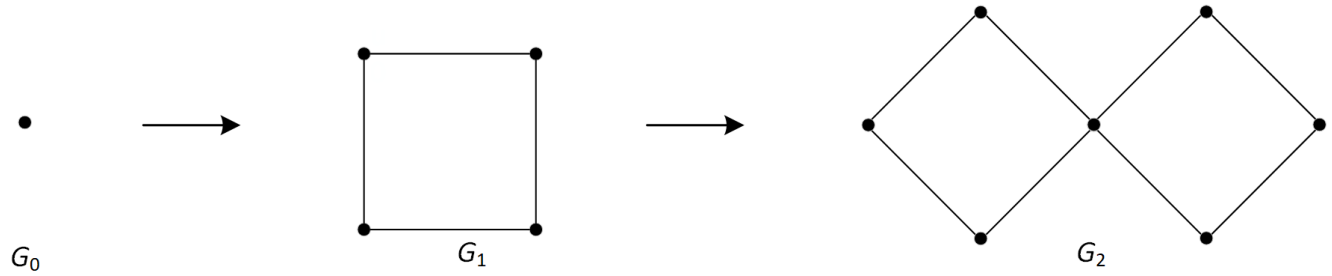

Fig. 1. Recursive Construction Process of the 4-Cactus Network G_2.

(2) If (u_0, u_1, u_2, u_3) is a pendant quadrilateral in G_t, and $d_{G_t}(u_1) = d_{G_t}(u_2) = d_{G_t}(u_3) = 2$, then for $t \geq 2$, u_0 is a cut vertex of G_t.
(3) For $t \geq 3$, every 4-cactus network G in Category B contains two pendant quadrilaterals, (u_0, u_1, u_2, u_3) and (u_0, v_1, v_2, v_3), that share a common vertex u_0.

Proof. Conclusions (1) and (2) follow directly from Definition 2. We now proceed to prove Conclusion (3). Let $G \in U(t)$ be a 4-cactus network in Category B. According to Definition 2, when $t \geq 3$, if $G \in U(3)$, then G has one vertex of degree 6 and nine vertices of degree 2. Therefore, Conclusion (3) holds.

Assume $t \geq 4$, and that Conclusion (3) holds for all values less than t. According to Definition 2, G contains a pendant quadrilateral (x, m_1, m_2, m_3) such that $d_G(m_1) = d_G(m_2) = d_G(m_3) = 2$. Since G belongs to Category B, we have $d_G(x) > 4$. Let $G^* = G - \{m_1, m_2, m_3\}$, then $G^* \in U(t-1)$. If G^* also belongs to Category B, then by the inductive hypothesis, G^* contains two pendant quadrilaterals (u_0, u_1, u_2, u_3) and (u_0, v_1, v_2, v_3) that share a common vertex u_0. Since G belongs to Category B, the vertex u_0 must not be the degree-4 vertex in any 4-pendant quadrilateral in G. If $x \notin \{u_0, u_1, u_2, u_3, v_1, v_2, v_3\}$, then the quadrilaterals (u_0, u_1, u_2, u_3) and (u_0, v_1, v_2, v_3) remain pendant quadrilaterals in G, and thus Conclusion (3) holds. Assume now that $x \in \{u_0, u_1, u_2, u_3, v_1, v_2, v_3\}$. If $x = u_0$, then the three pendant quadrilaterals (u_0, u_1, u_2, u_3), (u_0, v_1, v_2, v_3), and (u_0, m_1, m_2, m_3) all share the common vertex u_0. Therefore, Conclusion (3) also holds. If $x \in \{u_1, u_2, u_3, v_1, v_2, v_3\}$, by symmetry, assume $x = u_1$, then G contains two pendant quadrilaterals: (u_0, u_1, u_2, u_3) and (u_1, m_1, m_2, m_3), with u_1 as their common vertex. Additionally, since $d_G(u_1) = 4$, so the quadrilateral (u_1, m_1, m_2, m_3) is a 4-pendant quadrilateral. Therefore, G belongs to Category A, which contradicts the fact that G belongs to Category B. If G^* belongs to Category A, then G^* contains a 4-pendant quadrilateral $(u_0^*, u_1^*, u_2^*, u_3^*)$, where $d_{G^*}(u_1^*) = d_{G^*}(u_2^*) = d_{G^*}(u_3^*) = 2$. Since G belongs to Category B, we have $x \in \{u_0^*, u_1^*, u_2^*, u_3^*\}$. If $x = u_0^*$, then G contains two pendant quadrilaterals: (x, u_1^*, u_2^*, u_3^*) and (x, m_1, m_2, m_3), which satisfy Conclusion (3). If $x \in \{u_1^*, u_2^*, u_3^*\}$, then G belongs to Category A, which contradicts the assumption that G belongs to Category B. In conclusion, Conclusion (3) is proven. □

According to Definition 2, the topological structure of a 4-cactus network can vary in many ways. Below, we introduce two special types of 4-cactus networks: PA_t^4 and ST_t^4. These two specific types of 4-cactus networks are illustrated in Figures 2 and 3, respectively. PA_t^4 is a quadrilateral path consisting of t quadrilaterals (4-cycle C_4), constructed by connecting PA_{t-1}^4 with a new quadrilateral. This connection is made by identifying a degree-2 vertex of a pendant quadrilateral in PA_{t-1}^4 with one vertex of the new quadrilateral. ST_t^4 consists of t quadrilaterals (4-cycle C_4), all sharing exactly one common vertex, which is referred to as the center of ST_t^4.

Subtree-Counting Algorithm. Algorithm for enumerating subtrees of a 4-cactus network $G_t = (V(G_t), E(G_t))$.

Initialize: Let $G_t = (V(G_t), E(G_t))$ be a 4-cactus network consisting of t quadrilaterals (4-cycle C_4). Its vertex set is $V(G_t) = \{u_0, u_1, v_1, w_1, u_2, v_2, w_2, \ldots, u_t, v_t, w_t\}$, and its edge set is $\{e_1, e_2, \ldots, e_{4t}\}$, where for each $i = 1, 2, \ldots, t$, the vertices u_i, v_i, w_i are newly added in the i-th recursive iteration. Let $\eta(G_t)$ denote the number of subtrees of G_t. Initialize the weight of each vertex as an ordered pair of real numbers $(1, 0)$.

1: **for** $(k = t;\ k \geq 1; k--)$ **do**
2: Let (p_1, q_1), (p_2, q_2), (p_3, q_3), and (p_4, q_4) denote the weights of vertices u_k, v_k, w_k, and the common neighbor of u_k and v_k (excluding w_k), respectively. After deleting vertices u_k, v_k, w_k, update the value of (p_4, q_4) as follows:

$$\begin{cases} p_4 := p_4\big(4p_1p_2p_3 + p_1 + p_2 + p_1p_2 \\ \qquad + p_1p_3 + p_2p_3 + 1\big), \\ q_4 := q_1 + q_2 + q_3 + q_4 + p_1p_2p_3 \\ \qquad + p_1p_3 + p_2p_3 + p_1 + p_2 + p_3. \end{cases} \tag{1}$$

3: **end for**
4: **return** $\eta(G_t) = p_4 + q_4$.

It is easy to see that the Subtree-Counting algorithm performs a total of t iterations, so its time complexity is $O(t)$. When executing the Subtree-Counting Algorithm on a 4-cactus network $G_t \in U(t)$, we can regard the execution order of the algorithm as the reverse of the construction process of G_t. Specifically, during the construction of G_t, a new quadrilateral is added in each iteration; conversely, each iteration of the algorithm's for-loop contracts and removes one quadrilateral. Therefore, the for-loop iterations in the algorithm are valid throughout the process, and by the end of execution, only a single initial vertex remains.

To facilitate understanding of the execution process of the Subtree-Counting Algorithm, two examples, Example 1 and Example 2, are provided below.

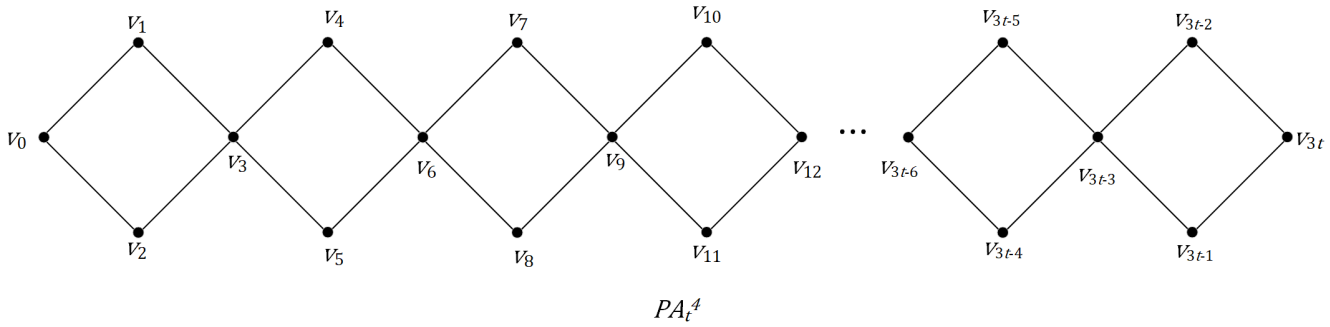

Fig. 2. The 4-cactus network PA_t^4.

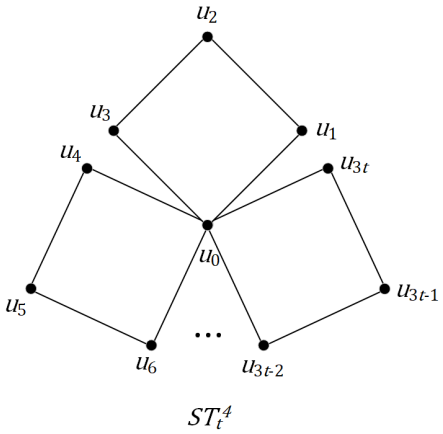

Fig. 3. The 4-cactus network ST_t^4.

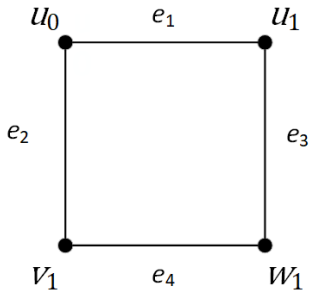

Fig. 4. The 4-cactus network G_1.

Example 1. Let $G_1 = (V, E)$ be a quadrilateral (4-cycle C_4), where $V = \{u_0, u_1, v_1, w_1\}$ and $E = \{e_1, e_2, e_3, e_4\}$, as shown in Figure 4. Then G_1 has six subtrees that do not contain u_0, namely the complete graphs on the vertex sets $\{u_1\}$, $\{v_1\}$, $\{w_1\}$, $\{u_1, w_1\}$, $\{v_1, w_1\}$, and $G_1 - \{u_0\}$. In addition, G_1 has ten subtrees that contain u_0, specifically the complete graphs on $\{u_0\}$, $\{u_0, u_1\}$, and $\{u_0, v_1\}$, and the graphs obtained by deleting a single vertex or a single edge: $G_1 - \{u_1\}$, $G_1 - \{v_1\}$, $G_1 - \{w_1\}$, $G_1 - \{e_1\}$, $G_1 - \{e_2\}$, $G_1 - \{e_3\}$, and $G_1 - \{e_4\}$. Therefore, the total number of subtrees of G_1 is $\eta(G_1) = 6 + 10 = 16$. Applying the Subtree-Counting Algorithm, the initial weights of the vertices u_0, u_1, v_1, and w_1 are all set to $(1, 0)$. According to equation (1), we obtain $p_4 = 10$ and $q_4 = 6$. Consequently, by executing the Subtree-Counting Algorithm, we verify that $\eta(G_1) = 16$.

Example 2. Let G be a 4-cactus network generated through five iterations, as illustrated in Figure 5. By applying the Subtree-Counting Algorithm to G, the process of computing the number of subtrees is visualized step by step in the Figure 5. In each computation step i (where $i = 1, 2, 3, 4, 5$), the three vertices a_i, b_i, c_i are removed. At the final step, the algorithm yields the result: $\eta(G) = 43210 + 1254 = 44464$.

Next, we prove the correctness of the Subtree-Counting Algorithm.

Theorem 4. Let G_t be a 4-cactus network generated through t iterations, with vertex set $V(G_t) = \{u_0, u_1, v_1, w_1, u_2, v_2, w_2, \ldots, u_t, v_t, w_t\}$, and let (p, q) denote the weight of vertex u_0. When the Subtree-Counting Algorithm terminates, the following conclusions hold:

(1) p is the number of subtrees of G_t that contain u_0, denoted by $\eta_{u_0}(G_t)$.
(2) q is the number of subtrees of G_t that do not contain u_0, denoted by $\eta(G_t - u_0)$.
(3) $p + q$ is the total number of subtrees in G_t, denoted by $\eta(G_t)$.

Proof. We apply mathematical induction on t. When $t = 1$, G_1 is a quadrilateral (4-cycle C_4). According to the Subtree-Counting Algorithm, we obtain $p = 10$ and $q = 6$. It has been verified that $\eta(G_1) = 16$. Therefore, the conclusions hold for $t = 1$. Now, assume that $k \geq 2$, and that the above conclusions hold for all $t < k$.

When $t = k \geq 2$, the structure of G_t is illustrated in Figure 6. The following proof will refer to Figure 6 and the notations used therein. From the figure, it can be seen that S_1, S_2, S_3, and S_4 are four vertex-disjoint subgraphs, where: S_1 contains only one vertex u_1 from the initial quadrilateral G_1, S_2 contains only one vertex v_1 from G_1, S_3 contains only one vertex w_1 from G_1, and S_4 contains only one vertex u_0 from G_1.

When the vertices u_1, v_1, w_1, and u_0 are each regarded as a vertex in S_1, S_2, S_3, and S_4, respectively, for convenience of notation, we relabel them as m_1, m_2, m_3, and m_4. In the remainder of the proof of the theorem, we identify $u_1 = m_1$, $v_1 = m_2$, $w_1 = m_3$, and $u_0 = m_4$.

Since G_1 is the initial quadrilateral of G_t, each m_i can be regarded as the initial vertex of the 4-cactus network S_i. For each $i \in \{1, 2, 3, 4\}$, S_i is at most a 4-cactus network generated through $t-1$ recursive iterations. Thus, by

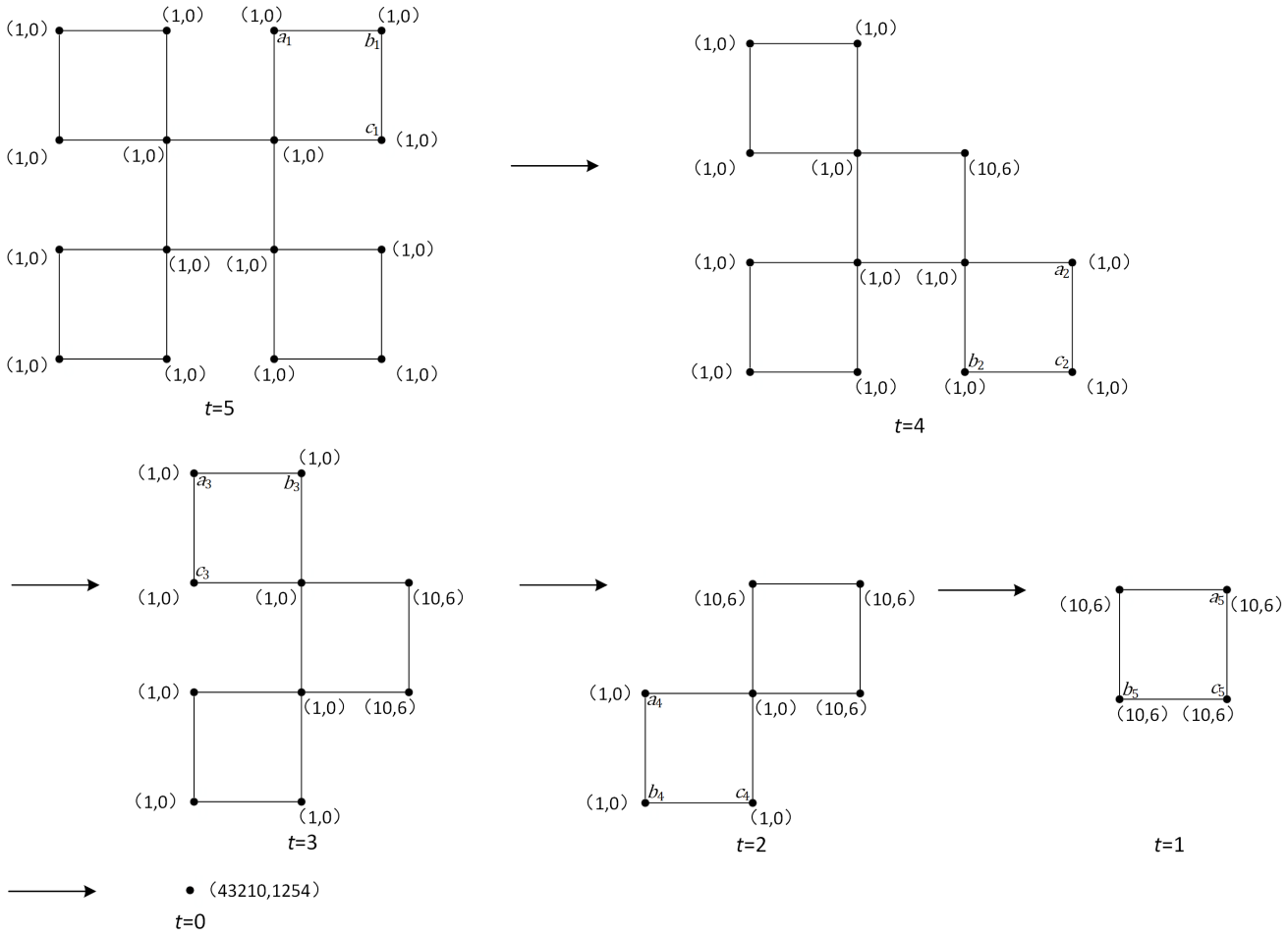

Fig. 5. The process of computing the number of subtrees of G.

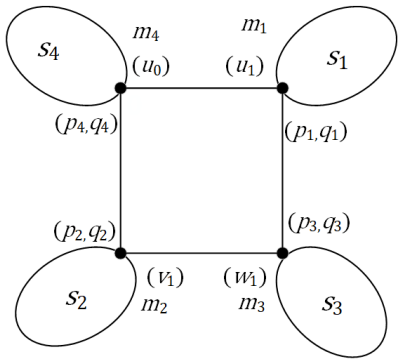

Fig. 6. The graph G_t.

applying the Subtree-Counting Algorithm to each S_i, when the algorithm terminates, only the vertex m_i remains in S_i, and the corresponding weights of m_1, m_2, m_3, and m_4 are (p_1, q_1), (p_2, q_2), (p_3, q_3), and (p_4, q_4), respectively. By the inductive hypothesis, conclusions (a) and (b) hold for each $i \in \{1, 2, 3, 4\}$: (a) p_i is the number of subtrees in S_i that contain m_i; (b) q_i is the number of subtrees in S_i that do not contain m_i.

At this stage of the Subtree-Counting Algorithm, the remaining graph to be processed is $G_1 = (u_0, u_1, v_1, w_1)$, and the current weights of the vertices u_0, u_1, v_1, and w_1 are $(p_4, q_4), (p_1, q_1), (p_2, q_2)$, and (p_3, q_3), respectively. In the final execution of the Subtree-Counting Algorithm, the vertices u_1, v_1, and w_1 are deleted, and according to equation (1), the weight (p, q) of the vertex u_0 is updated as follows:

$$\begin{cases} p = p_4\big(4p_1p_2p_3 + p_1 + p_2 + p_1p_2 \\ \quad + p_1p_3 + p_2p_3 + 1\big), \\ q = q_1 + q_2 + q_3 + q_4 + p_1p_2p_3 \\ \quad + p_1p_3 + p_2p_3 + p_1 + p_2 + p_3. \end{cases} \tag{2}$$

Let $\mathcal{TS}$ denote the set of all subtrees of G_t. We now

partition the set $\mathcal{TS}$ in detail as follows:

$$\begin{aligned}
\mathcal{TS}_0 &= \{T \in \mathcal{TS} : m_1 \in V(T) \text{ and } m_2, m_3, m_4 \notin V(T)\},\\
\mathcal{TS}_1 &= \{T \in \mathcal{TS} : m_2 \in V(T) \text{ and } m_1, m_3, m_4 \notin V(T)\},\\
\mathcal{TS}_2 &= \{T \in \mathcal{TS} : m_3 \in V(T) \text{ and } m_1, m_2, m_4 \notin V(T)\},\\
\mathcal{TS}_3 &= \{T \in \mathcal{TS} : m_4 \in V(T) \text{ and } m_1, m_2, m_3 \notin V(T)\},\\
\mathcal{TS}_4 &= \{T \in \mathcal{TS} : m_1, m_2 \in V(T) \text{ and } m_3, m_4 \notin V(T)\},\\
\mathcal{TS}_5 &= \{T \in \mathcal{TS} : m_1, m_3 \in V(T) \text{ and } m_2, m_4 \notin V(T)\},\\
\mathcal{TS}_6 &= \{T \in \mathcal{TS} : m_1, m_4 \in V(T) \text{ and } m_2, m_3 \notin V(T)\},\\
\mathcal{TS}_7 &= \{T \in \mathcal{TS} : m_2, m_3 \in V(T) \text{ and } m_1, m_4 \notin V(T)\},\\
\mathcal{TS}_8 &= \{T \in \mathcal{TS} : m_2, m_4 \in V(T) \text{ and } m_1, m_3 \notin V(T)\},\\
\mathcal{TS}_9 &= \{T \in \mathcal{TS} : m_3, m_4 \in V(T) \text{ and } m_1, m_2 \notin V(T)\},\\
\mathcal{TS}_{10} &= \{T \in \mathcal{TS} : m_1, m_2, m_3 \in V(T) \text{ and } m_4 \notin V(T)\},\\
\mathcal{TS}_{11} &= \{T \in \mathcal{TS} : m_1, m_2, m_4 \in V(T) \text{ and } m_3 \notin V(T)\},\\
\mathcal{TS}_{12} &= \{T \in \mathcal{TS} : m_2, m_3, m_4 \in V(T) \text{ and } m_1 \notin V(T)\},\\
\mathcal{TS}_{13} &= \{T \in \mathcal{TS} : m_1, m_3, m_4 \in V(T) \text{ and } m_2 \notin V(T)\},\\
\mathcal{TS}_{14} &= \{T \in \mathcal{TS} : m_1, m_2, m_3, m_4 \in V(T)\},\\
\mathcal{TS}_{15} &= \{T \in \mathcal{TS} : m_1, m_2, m_3, m_4 \notin V(T)\}.
\end{aligned}$$

From the partition of the set $\mathcal{TS}$, it follows that $\mathcal{TS} = \bigcup_{i=0}^{15} \mathcal{TS}_i$. According to (a) and (b) above, we have: $|\mathcal{TS}_0| = p_1$, $|\mathcal{TS}_1| = p_2$, $|\mathcal{TS}_2| = p_3$, $|\mathcal{TS}_3| = p_4$, and $|\mathcal{TS}_{15}| = q_1 + q_2 + q_3 + q_4$. Since the edge u_1w_1 is included in every tree in $\mathcal{TS}_5$, it follows that $|\mathcal{TS}_5| = p_1p_3$, and similarly, $|\mathcal{TS}_6| = p_1p_4$, $|\mathcal{TS}_7| = p_2p_3$, $|\mathcal{TS}_8| = p_2p_4$. Because there are no edges u_0w_1 and u_1v_1 in G_t, it follows that $|\mathcal{TS}_4| = |\mathcal{TS}_9| = 0$. Since the edges u_1w_1 and v_1w_1 are included in every tree in $\mathcal{TS}_{10}$, we have $|\mathcal{TS}_{10}| = p_1p_2p_3$, and similarly, $|\mathcal{TS}_{11}| = p_1p_2p_4$, $|\mathcal{TS}_{12}| = p_2p_3p_4$, $|\mathcal{TS}_{13}| = p_1p_3p_4$. Each tree in $\mathcal{TS}_{14}$ includes exactly three edges of G_1, and there are only four distinct eligible combinations of edges in G_1, therefore $|\mathcal{TS}_{14}| = 4p_1p_2p_3p_4$. From the definitions of $\mathcal{TS}_i$ and equation (2), we have:

$$\begin{aligned}
\eta_{u_0}(G_t) &= |\mathcal{TS}_3| + |\mathcal{TS}_6| + |\mathcal{TS}_8| + |\mathcal{TS}_9|\\
&\quad + |\mathcal{TS}_{11}| + |\mathcal{TS}_{12}| + |\mathcal{TS}_{13}| + |\mathcal{TS}_{14}|\\
&= p_4 + p_1p_4 + p_2p_4 + 0 + p_1p_2p_4 + p_1p_3p_4\\
&\quad + p_2p_3p_4 + 4p_1p_2p_3p_4\\
&= p_4\big(4p_1p_2p_3 + p_1 + p_2 + p_1p_2 + p_1p_3\\
&\quad + p_2p_3 + 1\big) = p.
\end{aligned} \tag{3}$$

Similarly, we obtain:

$$\begin{aligned}
\eta(G_t - u_0) &= |\mathcal{TS}_0| + |\mathcal{TS}_1| + |\mathcal{TS}_2| + |\mathcal{TS}_4|\\
&\quad + |\mathcal{TS}_5| + |\mathcal{TS}_7| + |\mathcal{TS}_{10}| + |\mathcal{TS}_{15}|\\
&= p_1 + p_2 + p_3 + 0\\
&\quad + p_1p_3 + p_2p_3 + p_1p_2p_3 + q_1 + q_2\\
&\quad + q_3 + q_4 = q.
\end{aligned} \tag{4}$$

Thus,

$$\eta(G_t) = \sum_{i=0}^{15} |\mathcal{TS}_i| = \eta_{u_0}(G_t) + \eta(G_t - u_0) = p + q.$$

This verifies that, when the Subtree-Counting Algorithm terminates, the outputs correspond to the values stated in (1), (2), and (3) of Theorem 4. Therefore, the correctness of the Subtree-Counting Algorithm is established. □

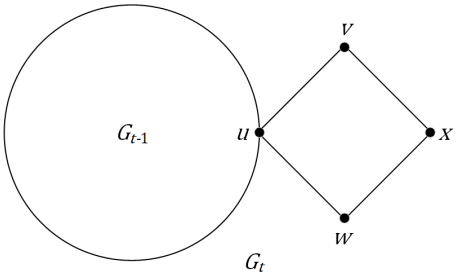

Fig. 7. The 4-cactus network G_t.

III. Upper and Lower Bounds on the Number of Subtrees of 4-Cactus Networks

In this section, we determine the upper and lower bounds on the number of subtrees of 4-cactus networks and derive the corresponding formulas for these bounds. We first present Lemma 5, which will be used in the subsequent proofs.

Lemma 5. [8] Let G be a connected network with $n \geq 3$ vertices. Suppose G_1 and G_2 are two subgraphs of G such that $G = G_1 \cup G_2$, $V(G_1) \cap V(G_2) = \{v\}$, and $E(G_1) \cap E(G_2) = \emptyset$. Then:

$$\begin{aligned}
\eta(G) = \eta(G_1) + \eta(G_2) + (\eta(G_1) - \eta(G_1 - v) - 1)\\
\cdot (\eta(G_2) - \eta(G_2 - v) - 1) - 1.
\end{aligned} \tag{5}$$

Theorem 6. For an integer t, ST_t^4 is the unique 4-cactus network in $U(t)$ that attains the maximum number of subtrees.

Proof. Using mathematical induction on t, it is clear that the theorem holds for $t \in \{0, 1, 2\}$. Let $k \geq 3$ be an integer, and assume that the theorem holds for all $t < k$. Now, we consider the case when $t = k$.

Let G^* be a 4-cactus network containing k quadrilaterals, and let (u, v, w, x) be a pendent quadrilateral such that $d_{G^*}(u) \geq 4$ and $d_{G^*}(v) = d_{G^*}(w) = d_{G^*}(x) = 2$. By Conclusion (2) of Theorem 3, u is a cut vertex. Let $G_1^* = G^* - \{v, w, x\}$ and $G_2^* = (u, v, w, x)$.

By the inductive hypothesis, we have $\eta(G_1^*) \leq \eta(ST_{k-1}^4)$, with equality if and only if $G_1^* = ST_{k-1}^4$. If $G^* = ST_k^4$, then $G_1^* - u = (k-1)P_3$, where P_3 denotes a path with three vertices. If $G^* \neq ST_k^4$, then $(k-1)P_3$ is a proper subnetwork of $G_1^* - u$, which implies that $\eta(G_1^* - u) > \eta((k-1)P_3)$. Since $G_2^* = (u, v, w, x)$ and $G_2^* - u = P_3$, by Lemma 5, it follows that $\eta(G^*) \leq \eta(ST_k^4)$, with equality if and only if $G^* = ST_k^4$. Thus, Theorem 6 is proved. □

Corollary 7 follows directly from the Subtree-Counting Algorithm.

Corollary 7. Let $G_t \in U(t)$ be a 4-cactus network, as shown in Figure 7, where the circular region represents G_{t-1}. Suppose (u, v, w, x) is a pendant quadrilateral of G_t, with degrees $d_{G_t}(v) = d_{G_t}(w) = d_{G_t}(x) = 2$. Also, let $G_{t-1} = G_t - \{v, w, x\}$. Then:

(1) $\eta(G_t) = 9\eta_u(G_{t-1}) + \eta(G_{t-1}) + 6$.
(2) $\eta_w(G_t) = 7\eta_u(G_{t-1}) + 3$.
(3) $\eta_x(G_t) = 6\eta_u(G_{t-1}) + 4$.

Theorem 8. Let $t \geq 3$ be an integer, and $G \in U(t)$ be a 4-cactus network that belongs to Category B. Then, there exists

a 4-cactus network G^* that belongs to Category A such that $\eta(G^*) < \eta(G)$.

Proof. Let $G \in U(t)$ be a 4-cactus network belonging to Category B. From Conclusion (3) of Theorem 3, it follows that G contains two pendant quadrilaterals (u_0, u_1, u_2, u_3) and (u_0, u, w, v), with degree conditions $d_G(u_0) \geq 6$ and $d_G(u_1) = d_G(u_2) = d_G(u_3) = d_G(u) = d_G(w) = d_G(v) = 2$, as shown in Figure 8(a). Let S be as shown in Figure 8. Let p_0 (or q_0) represent the number of subtrees in S that contain u_0 (or do not contain u_0). From the Subtree-Counting Algorithm, it is known that by deleting vertices u_1, u_2, and u_3 from G, the weight of vertex u_0 is reset to (p_{u_0}, q_{u_0}) as follows: $p_{u_0} = 10p_0$ and $q_{u_0} = q_0 + 6$. Then, by deleting vertices u_0, w, and v from G, the weight of vertex u is reset to $\eta_u(G) = 7p_{u_0} + 3$ and $\eta(G - u) = 3p_{u_0} + 3 + q_{u_0}$. Thus, we obtain $\eta(G) = 100p_0 + q_0 + 12$.

Let G^* be the graph shown in Figure 8(b), which G^* is obtained by gluing the vertex u_0 from S and a degree-2 vertex from PA_2^4. From the Subtree-Counting Algorithm, it is known that by deleting vertices u_0, u_1, and u_2 from G^*, the weight of vertex u_3 is reset to (p_{u_3}, q_{u_3}) as follows: $p_{u_3} = 6p_0 + 4$ and $q_{u_3} = 4p_0 + q_0 + 2$. Next, by deleting vertices u_3, u_4, and u_5 from G^*, the weight of vertex u is reset to $\eta_u(G^*) = 6p_{u_3} + 4$ and $\eta(G^* - u) = 4p_{u_3} + q_{u_3} + 2$. Thus, we obtain $\eta(G^*) = 64p_0 + q_0 + 48$. By calculating $\eta(G) - \eta(G^*) = 36p_0 - 36$, and since $p_0 > 1$, it follows that $\eta(G) - \eta(G^*) > 0$. Therefore, Theorem 8 is proved. □

Theorem 9. For an integer t, PA_t^4 is the unique 4-cactus network in $U(t)$ that attains the minimum number of subtrees.

Proof. From Theorem 8, we know that for every 4-cactus network G that belongs to Category B in $U(t)$, there exists a 4-cactus network G^* that belongs to Category A, and G^* has fewer subtrees. Therefore, it is sufficient to prove that PA_t^4 is the unique 4-cactus network in Category A with the minimum number of subtrees.

We will use mathematical induction on t. Let $G_t \in U(t)$ be a 4-cactus network, and (u, v, w, x) be one of its pendant quadrilaterals such that $d_{G_t}(u) = 4$ and $d_{G_t}(v) = d_{G_t}(w) = d_{G_t}(x) = 2$. We will prove the following conclusions: (1) PA_t^4 is the unique 4-cactus network in $U(t)$ that attains the minimum number of subtrees; (2) $\eta_x(G_t)$ achieves its minimum value if and only if $G_t = \mathrm{PA}_t^4$.

For $t = 1$ and $t = 2$, the conclusions are clearly valid. Assume $k \geq 3$, and that the conclusions hold for all $t < k$. Now, consider the case where $t = k$. Let $G_{k-1} = G_k - \{v, w, x\}$. Since $d_{G_{k-1}}(u) = 2$, by Corollary 7 and the induction hypothesis, we have:

$$\eta_x(G_k) = 6\eta_u(G_{k-1}) + 4 \geq 6\eta_u(\mathrm{PA}_{k-1}^4) + 4, \quad (6)$$

$$\begin{aligned} \eta(G_k) &= 9\eta_u(G_{k-1}) + \eta(G_{k-1}) + 6 \\ &\geq 9\eta_u(\mathrm{PA}_{k-1}^4) + \eta(\mathrm{PA}_{k-1}^4) + 6. \end{aligned} \quad (7)$$

Equality holds if and only if $G_k = \mathrm{PA}_k^4$. Thus, Theorem 9 is proved. □

Next, we will calculate the number of subtrees of ST_k^4 and PA_t^4, and provide the specific formulas for the calculations. Let u_0 be the central vertex of ST_k^4, and let v_0 be a degree-2 vertex of a pendant quadrilateral in PA_t^4, as shown in Figures 2 and 3. Using the Subtree-Counting Algorithm, the following recurrence formulas can be derived:

$$\begin{cases} \eta_{u_0}(\mathrm{ST}_t^4) = 10\,\eta_{u_0}(\mathrm{ST}_{t-1}^4), & \eta_{u_0}(\mathrm{ST}_0^4) = 1, \\ \eta(\mathrm{ST}_t^4 - u_0) = \eta(\mathrm{ST}_{t-1}^4 - u_0) + 6, & \eta(\mathrm{ST}_0^4 - u_0) = 0. \end{cases} \quad (8)$$

and

$$\begin{cases} \eta_{v_0}(\mathrm{PA}_t^4) = 6\,\eta_{v_0}(\mathrm{PA}_{t-1}^4) + 4, & \eta_{v_0}(\mathrm{PA}_0^4) = 1, \\ \eta(\mathrm{PA}_t^4 - v_0) = \eta(\mathrm{PA}_{t-1}^4 - v_0) & \\ \qquad + 4\,\eta_{v_0}(\mathrm{PA}_{t-1}^4) + 2, & \eta(\mathrm{PA}_0^4 - v_0) = 0. \end{cases} \quad (9)$$

We will solve the recurrence formulas separately and obtain the following:

$$\eta_{u_0}(\mathrm{ST}_t^4) = 10^t. \quad (10)$$

$$\eta(\mathrm{ST}_t^4 - u_0) = 6t. \quad (11)$$

$$\eta(\mathrm{ST}_t^4) = 10^t + 6t. \quad (12)$$

$$\eta_{v_0}(\mathrm{PA}_t^4) = \frac{1}{5}(9 \cdot 6^t - 4). \quad (13)$$

$$\eta(\mathrm{PA}_t^4 - v_0) = \frac{36}{25}(6^t - 1) - \frac{6t}{5}. \quad (14)$$

$$\eta(\mathrm{PA}_t^4) = \frac{81}{25} \cdot 6^t - \frac{6t}{5} - \frac{56}{25}. \quad (15)$$

By Theorems 6 and 9, $\eta(\mathrm{ST}_t^4)$ and $\eta(\mathrm{PA}_t^4)$ represent the upper and lower bounds, respectively, on the number of subtrees in a 4-cactus network with t quadrilaterals. As shown in equations (12) and (15), it can be observed that as the number of quadrilaterals increases, the number of subtrees in ST_t^4 and PA_t^4 grows exponentially, increasing at a very rapid rate.

IV. Conclusion

This paper investigates the subtree counting problem for 4-cactus networks, proposes a linear algorithm for calculating the number of subtrees in a 4-cactus network, determines the two extreme values of the subtree count for 4-cactus networks, and characterizes the network topologies corresponding to these two extremal graphs. This work provides useful guidance for the analysis and design of highly reliable networks. However, when extending the ideas of this algorithm to n-cactus networks, certain challenges remain. Future research could further explore the subtree counting problem for n-cactus networks and extend the study to other important network properties of cactus networks.

References

[1] L. A. Székely and H. Wang, "On subtrees of trees," *Advances in Applied Mathematics*, vol. 34, no. 1, pp138–155, 2005.

[2] W. Yan and Y.-N. Yeh, "Enumeration of subtrees of trees," *Theoretical Computer Science*, vol. 369, no. 1-3, pp256–268, 2006.

[3] L. A. Székely and H. Wang, "Binary trees with the largest number of subtrees," *Discrete applied mathematics*, vol. 155, no. 3, pp374–385, 2007.

[4] R. Kirk and H. Wang, "Largest number of subtrees of trees with a given maximum degree," *SIAM Journal on Discrete Mathematics*, vol. 22, no. 3, pp985–995, 2008.

[5] Y. Yang, H. Liu, H. Wang, and S. Makeig, "Enumeration of bc-subtrees of trees," *Theoretical Computer Science*, vol. 580, pp59–74, 2015.

[6] L. Dong, H. Zhao, and H.-J. Lai, "Entropy and enumeration of subtrees in a cactus network," *Frontiers in Physics*, vol. 8, p.575648, 2020.

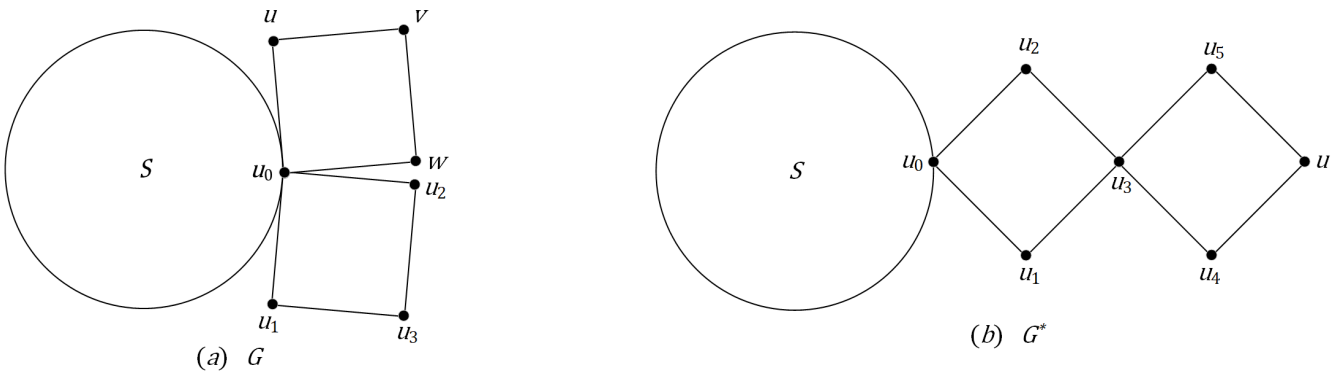

Fig. 8. (a) The Network G. (b) The Network G^*.

[7] X.-M. Zhang, X.-D. Zhang, D. Gray, and H. Wang, "The number of subtrees of trees with given degree sequence," *Journal of Graph Theory*, vol. 73, no. 3, pp280–295, 2013.

[8] Y. Xiao, H. Zhao, Z. Liu, and Y. Mao, "Trees with large numbers of subtrees," *International Journal of Computer Mathematics*, vol. 94, no. 2, pp372–385, 2017.

[9] D. Sun, L. Li, K. Liu, H. Wang, and Y. Yang, "Enumeration of subtrees of planar two-tree networks," *Applied Mathematics and Computation*, vol. 434, p.127404, 2022.

[10] D. Sun, H. Liu, Y. Yang, L. Li, H. Zhang, and A. Fahad, "Enumeration of subtrees of two families of self-similar networks based on novel two-forest dual transformations," *The Computer Journal*, vol. 67, no. 5, pp1652–1662, 2024.

[11] T. Horibe, Y. Itokawa, T. Uchida, Y. Suzuki, and T. Miyahara, "Enumeration algorithms for all characteristic paths and subtrees from structurally compressed tree-structured data," *IAENG International Journal of Computer Science*, vol. 45, no. 1, pp206–218, 2018.

[12] D. Rautenbach and L. Volkmann, "The domatic number of block-cactus graphs," *Discrete Mathematics*, vol. 187, no. 1-3, pp185–193, 1998.

[13] B. Ben-Moshe, B. Bhattacharya, Q. Shi, and A. Tamir, "Efficient algorithms for center problems in cactus networks," *Theoretical Computer Science*, vol. 378, no. 3, pp237–252, 2007.

[14] X. Liu, S. Zhou, J. Liu, and Z. Yu, "Subgraph reliability of the cactus-based networks," in *2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C)*. IEEE, 2021, pp890–896.

[15] M. Arcak, "Diagonal stability on cactus graphs," in *49th IEEE Conference on Decision and Control (CDC)*. IEEE, 2010, pp6553–6558.

[16] X. Liu, S. Zhou, J. Liu, and H. Zhang, "Reliability analysis of the cactus-based networks based on subsystem," *The Computer Journal*, vol. 67, no. 1, pp142–152, 2024.

[17] B. Ben-Moshe, A. Dvir, M. Segal, and A. Tamir, "Centdian computation in cactus graphs," *Journal of Graph Algorithms and Applications*, vol. 16, no. 2, pp199–224, 2012.