# Appendix

## A. Property 1 (Uniqueness)

*Proof.* We prove this by contradiction. Suppose there are two distinct $(p, \lambda, k)$-PiTrusses $H$ and $H'$. According to Definition 2, for all edge schema $(u, v) \in E(H)$ ($(u, v) \in E(H')$ resp.), we have the uphold layer set of $(u, v)$ $\mathcal{L} = ul(u, v, k)$ ($\mathcal{L} = ul(u, v, k)$ resp.) satisfying $|\mathcal{L}| \geq \lambda$ and $p \in \mathcal{L}$. Therefore, for each $(u, v) \in E(H) \cup E(H')$, we have its uphold layer set satisfying $|\mathcal{L}| \geq \lambda$ and $p \in \mathcal{L}$. The graph edge induced by $E(H) \cup E(H')$ should also be a $(p, \lambda, k)$-PiTruss. It contradicts the maximality of $H$ and $H'$. Thus, the property holds. □

## B. Property 2 (Hierarchy)

*Proof.* Suppose $H$ is a $(p, \lambda, k+1)$-PiTruss, and $(u, v) \in E(H)$ is an edge schema in $H$. According to Definition 2, we have the uphold layer set $\mathcal{L} = ul(u, v, k+1)$ satisfying $|\mathcal{L}| \geq \lambda$ and $p \in \mathcal{L}$. Suppose the uphold layer set under $k$ is $\mathcal{L}' = ul(u, v, k)$ of edge schema $(u, v)$. Then we have for each uphold layer $l \in L$, $(u, v, l)$ is contained in at least $k+1$ triangles, so $(u, v, l)$ should also be contained in at least $k$ triangles. Then for each $l \in L$ we have $l \in L'$. We have $L \subseteq L'$ and $|\mathcal{L}'| \geq \lambda$ and $p \in \mathcal{L}'$. This means each edge schema in the $(p, \lambda, k+1)$-PiTruss is also an edge schema in the $(p, \lambda, k)$-PiTruss. Thus, the property holds. □

## C. Property 3 (Containment)

*Proof.* Let $H_1$ be a $(p, \lambda_1, k)$-PiTruss and let $H_2$ be a $(p, \lambda_2, k)$-PiTruss. By Definition 2, we have the uphold layer set of each edge schema $(u, v) \in E(H_2)$ $\mathcal{L} = ul(u, v, k)$ satisfying $|\mathcal{L}| \geq \lambda_2 \geq \lambda_1$. So all the edge schemas in $E(H_2)$ satisfy the pivotal constraint $p$ and the background constraint of $\lambda_1$ w.r.t. $k$. Thus, the property holds. □

## D. Property 4 (Minimum Degree)

*Proof.* Suppose $H$ is a $(p, \lambda, k)$-PiTruss, and $(u, v) \in E(H)$ is an edge schema in $H$. According to Definition 2, we have the uphold layer set $\mathcal{L} = ul(u, v, k)$ satisfying $|\mathcal{L}| \geq \lambda$ and $p \in \mathcal{L}$. In each layer $l \in \mathcal{L}$, we have $u$ and $v$ contained in at least $k-2$ triangles, meaning that $u$ and $v$ have at least $k-2$ common neighbors. Thus $d_l(v) \geq k-1$. □

## E. Theorem 1 (Density)

This proof is the same as the proof of density lower bound in FTCS [14].

## F. Theorem 2 (Edge Connectivity)

*Proof.* Suppose after removing the minimal set of edges $S$, we will get two connected components $C_1$ and $C_2$, where there is no edge $(u, v, l)$ that holds $u \in C_1$, $v \in C_2$, and $l \in L$. We calculate how many edges will be removed to get $C_1$ and $C_2$, i.e., $|S|$.

If $|C_1| = 1$ or $|C_2| = 1$. Let's assume $|C_1| = 1$, which means there is only one vertex $u \in C_1$. Before the removal, $C_1$ and $C_2$ are triangle-connected, which means there are at least two vertices $v_1, v_2 \in C_2$, where edge schemas $(u, v_1), (u, v_2), (v_1, v_2)$ exist. Meanwhile, $\tau(p, \lambda, u, v_1) \geq k$ and $\tau(p, \lambda, v_2) \geq k$ hold. So there will be at least $\lambda$ layers $\mathcal{L} = \{l_1, \ldots, l_\lambda\}$ where $u, v_1$ ($u, v_2$, resp.) has at least $k-3$ common neighbors except $v_2$ ($v_1$, resp.). These common neighbors $w$ all belong to $C_2$ because $|C_1| = 1$. So there will be $(u, w, l_i)$ edge that needs to be removed. In each layer $l_i \in \mathcal{L}$, there will be at least $(k-3)+2$ edges that need to be removed. In total, $|S| \geq \lambda(k-1)$.

If $|C_1| \neq 1$ and $|C_2| \neq 1$. According to triangle-connectivity, there will be two vertices $u_1, u_2 \in C_1$ and $v_1, v_2 \in C_2$, where edge schemas $(u_1, v_1), (u_1, v_2), (u_2, v_1), (u_1, u_2), (v_1, v_2)$ exist. Meanwhile, $\tau(p, \lambda, u_1, v_1) \geq k$, $\tau(p, \lambda, u_1, v_2) \geq k$, and $\tau(p, \lambda, u_2, v_1) \geq k$ hold. Similarly, there will be at least $\lambda$ layers $\mathcal{L}_1 = \{l_1, \ldots, l_\lambda\}$ where $u_1, v_1$ ($u_1, v_2$, resp.) has at least $k-3$ common neighbors except $v_2$ ($v_1$, resp.). We can assume these common neighbors $w$ all belong to $C_2$ to make $|S|$ small. Meanwhile, there will be at least $\lambda$ layers $\mathcal{L}_2 = \{l_1, \ldots, l_\lambda\}$ where $u_1, v_1$ ($u_2, v_1$, resp.) has at least $k-3$ common neighbors except $u_2$ ($u_1$, resp.). In each layer $l_i \in \mathcal{L}_1$, there will be at least $(k-3)$ edges that need to be removed. In each layer $l_i \in \mathcal{L}_2$, there will also be at least $(k-3)$ edges that need to be removed. In total, $|S| \geq 2\lambda(k-3) + 3\lambda$.

In summary, $|S| \geq \lambda(k-1)$. □

## G. Theorem 3 (Diameter)

Research on triangle-connected truss communities in single-layer graphs [15] has proven that the size and diameter of a triangle-connected $k$-truss satisfy the following relationship:

**Theorem 5.** If $d$ is the diameter of a triangle-connected $k$-truss with $n$ vertices, then $d \leq \lfloor \frac{2n}{k+1} \rfloor$

*Proof.* Suppose $t = \{(v_1, l_1) \rightarrow (v_2, l_2) \rightarrow \cdots \rightarrow (v_p, l_p)\}$ is a path of the triangle-connected $(p, \lambda, k)$-PiTruss $H$. We can split $t$ into two sub-paths: $t_1$ contains all the adjacent $(v_i, l_i)$ and $(v_{i+1}, l_{i+1})$ in path $t$ where $v_i \neq v_{i+1}$. While $t_2$ contains all the adjacent $(v_i, l_i)$ and $(v_{i+1}, l_{i+1})$ in path $t$ where $v_i = v_{i+1}$ and $l_i \neq l_{i+1}$. In other words, $t_1$ represents the intra-layer edges in the path, while $t_2$ represents the inter-layer edges in the path. We have $diam(H) = min_{t \in H}(|t|)$ and $|t| = |t_1| + |t_2|$. Next, we construct a $t'$ composed of $t'_1$ and $t'_2$ whose upper bound can be determined.

We first construct a $t'_1$ with an upper bound. We consider on the graph schema $\mathcal{G}$ of $H$. According to the definition of triangle-connected $(p, \lambda, k)$-PiTruss, $G$ is a triangle-connected $k$-truss because each edge schema $(u, v)$ is contained in at least $k-2$ triangles, and each edge schema is triangle-connected in $\mathcal{G}$. So we can denote the diameter of $\mathcal{G}$ as $\{v_1 \rightarrow \cdots \rightarrow v_d\}$. And we can construct $t'_1 = \{(v_1, l_1) \rightarrow \cdots \rightarrow (v_d, l_d)\}$, who satisfies $|t'_1| \leq \lfloor \frac{2(V(H))}{k+1} \rfloor$ according to Theorem 5.

Next, we can construct $t'_2$ and $t'$ based on $t'_1$. Suppose $t_1^* = \{(v_1, l_1^*) \rightarrow (v_2, l_2^*) \rightarrow \cdots \rightarrow (v_d, l_d^*)\}$ where the layer set $L^* = \{l_1^*, \ldots l_d^*\}$ is the optimal layer set such that $|\bigcup_{l_i \neq l_{i+1}} \{(v_i, l_i), (v_{i+1}, \, l_{i+1})\}|$ is the smallest. In other words, using the layer set $L^*$, we can construct the shortest multilayer path $t'$ between $v_1$ and $v_d$. We mark this shortest path as $t^*$, and the corresponding $t'_2$ as $t^*_2$. We can set all the $l_i \in L^*$ to be $p$. Thus, $|t^*_2| = 0$.

In summary, we have

$$\begin{aligned} diam(H) = min_{t \in H}(|t|) \leq |t^*| = |t^*_1| + |t^*_2| \\ = |t^*_1| \leq \lfloor \frac{2|V(H)|}{k+1} \rfloor. \end{aligned}$$

□

### *H. Lemma 1*

*Proof.* (**Irreflexivity.**) Given a $k$-partial class $P$, $P \nless P$ because $k \nless k$.

(**Antisymmetry.**) Given a $k_1$-partial class $P_1$ and a $k_2$-partial class $P_2$. If $P_1 < P_2$, we have $k_1 < k_2$, Thus, $k_2 \nless k_1$, which means $P_2 \nless P_1$.

(**Transitivity.**) Suppose $P_1$ is a $k_1$-partial class of a triangle-connected $(p, \lambda, k_1)$-PiTruss $H_1$, $P_2$ is a $k_2$-partial class of a triangle-connected $(p, \lambda, k_2)$-PiTruss $H_2$, and $P_3$ is a $k_3$-partial class of a triangle-connected $(p, \lambda, k_3)$-PiTruss $H_3$. If $P_1 < P_2$ and $P_2 < P_3$, then $k_1 < k_2 < k_3$. So we have $H_2$ is a subgraph of $H_1$, and $H_3$ is a subgraph of $H_2$. Meanwhile, $\forall (u_1, v_1) \in P_1, (u_3, v_3) \in P_3$, there exist at least one $(u_2, v_2) \in P_2$, s.t. $(u_1, v_1)$ is triangle-connected to $(u_2, v_2)$ in $H_1$, and $(u_2, v_2)$ is triangle-connected to $(u_3, v_3)$ in $H_2$. Since $H_2$ is a subgraph of $H_1$, so $(u_2, v_2)$ must be triangle-connected to $(u_3, v_3)$ in $H_1$. According to the definition of triangle connectivity, $(u_1, v_1)$ is triangle-connected to $(u_3, v_3)$ in $H_1$. Thus, $P_1 < P_3$.

In summary, since the relationship of containment of partial classes possesses the properties of irreflexivity, antisymmetry, and transitivity, this relationship is a strict partial order. □

### *I. Lemma 2*

*Proof.* We prove by contradiction that the Hasse diagram is a forest. Suppose the Hasse diagram is not a forest. Then there exists a node in the diagram representing a $k$-partial class $P$ whose indegree is more than 2. Suppose two of them are $k_1$-partial class $P_1$ and $k_2$-partial class $P_2$.

If $k_1 \neq k_2$, suppose $k_1 < k_2 < k$. Then $\forall (u_1, v_1) \in P_1, (u_2, v_2) \in P_2$, there exists a $(u, v) \in P$ s.t. $(u_1, v_1)$ is triangle-connected to $(u_2, v_2)$ through $(u, v)$. So there should be $P_1 < P_2 < P$. There should not be a link between $P_1$ and $P$ according to the definition of the Hasse diagram, which leads to a contradiction.

If $k_1 = k_2 < k$. We have $P_1 < P$ and $P_2 < P$. Then $\forall (u_1, v_1) \in P_1, (u_2, v_2) \in P_2$, there exists a $(u, v) \in P$ s.t. $(u_1, v_1)$ is triangle-connected to $(u_2, v_2)$ through $(u, v)$. So there should be a $P' = P_1 \cup P_2$ that also holds $P' < P$. The existence of $P'$ contradicts the definition of a partial class, where each partial class is a maximal subset of the edge schema set $E(H)$ of the $(p, \lambda, k)$-PiTruss $H$.

In summary, since the indegree of all partial classes cannot exceed 2, the Hasse diagram must be a forest. □

### *J. Lemma 3*

*Proof.* We first prove each $T_x$ of $T$ represents a PiTC. Suppose $x$ is a $k$-partial class of $(p, \lambda)$-PiTruss. For each node $x_i$ in $T_x$, suppose it is a $k_i$-partial class. Then a subgraph $H$ is formulated by all the edge schemas containing $x_i$ and $x$. According to the definition of partial class, we have $\forall (u, v) \in E(H)$, $\tau(p, \lambda, u, v) \geq k$. Meanwhile, for all triangle schemas $\triangle$ and $\triangle'$ in $H$, they are triangle-connected because they are linked in $T_x$. The maximum of $H$ is guaranteed because each partial class is maximal, and edge schemas from other partial classes not in $T_x$ cannot be added into $H$ because they are not triangle-connected. Thus, each $T_x$ of $T$ represents a PiTC.

Next, we prove each PiTC is represented by a $T_x$ of $T$. For a PiTC subgraph $H$, the edge schemas in $E(H)$ can be categorized into a series of partial classes according to Definition 8. Each of these partial classes reflects a specific node in the tree. Otherwise, the fact that PiTC is the maximal subgraph or that the partial class is the largest subset would be violated. Since a PiTC is triangle-connected, these partial classes can form a tree $T'$. This tree is a subtree of $T$ because the tree nodes not in the subtree are not triangle-connected to $E(H)$, while $T'$ can not lose nodes in the subtree because $H$ is the maximal subgraph. Thus, each PiTC is represented by a $T_x$ of $T$. □

### *K. Theorem 4*

*Proof.* Each edge schema is contained in only one tree node $T_x$. Suppose there is an edge schema $(u, v) \in E(G)$ that is contained in two tree nodes $T_{x1}$ and $T_{x2}$. We have the connected components of $T_{x1}$ and $T_{x2}$ should be triangle-connected by edge schema $(u, v)$, so $x_1$ and $x_2$ should be the same tree node. So the number of all the tree nodes $|T_x|$ and the sum of all $|E(T_x)|$ are both smaller than $|E(G)|$.

In the worst case, each edge schema is a partial class. Each partial class is a node in the index, so the links between nodes can not exceed the links in the minimum spanning tree of all edges in $E(G)$. So the sum of both $|P(T_x)|$ and $|C(T_x)|$ should both be smaller than $|E(G)|$.

In summary, the size of the index tree should not exceed $O(|E(G)|)$. □

### *L. Complexity of the Peeling Framework (Algorithm 1)*

For the initialization before the peeling, $sup$ can be calculated once for all possible $p$ and $\lambda$, counting triangles for each edge will take $O(\sum_{l \in L} |E_l|^{1.5})$ [59]. While it takes $O(|E||L| \log |L|)$ to initialize $\tau(u, v)$ for each combo. The running time of Algorithm 1 is dominated by the time for updating the trussness in lines 10 and 13. During the peeling, each edge $(u, v, l) \in E$ is removed only once. For each edge $(u, v, l)$ removed by line 9 and 12, each common neighbor $w \in N_l(u) \cap N_l(v)$ may trigger an update using Procedure EstTrusness. Suppose we have $d_l(u) \leq d_l(v)$. Then each common neighbor $w \in N_l(u) \cap N_l(v)$ can be found by searching if each $w \in N_l(u)$ has an edge $(v, w, l) \in E_l$. Let $N^{\geq}_l(u)$ be $\{v | v \in N_l(u), d_l(v) \geq d_l(u)\}$. We have $|N^{\geq}_l(u)| \leq 2\sqrt{|E_l|}$ for any $u \in V$ [59]. Thus, the loop in lines 6–14 is executed for at most $\sum_{l\in L}\sum_{u\in V}(d_l(u)\cdot |N_l^{\geq}(u)|) = O(\sum_{l\in L}|E_l|^{1.5})$ times. For the worst case, each loop needs to update the trussness, thus the loop in lines 6–14 takes $O(|L|\log|L|\sum_{l\in L}|E_l|^{1.5})$ time. Putting all together, the time complexity of Algorithm 1 is $O(|L|\log|L|\cdot(|E| + \sum_{l\in L}|E_l|^{1.5}))$.

### M. Complexity of PiTCS-Online (Algorithm 2)

We have analyze the complexity of PEELING in line 1. The rest of Algorithm 2 is dominated by lines 8-11, which enumerates all the triangles once in all the connected components containing $v_q$. Similar to the analysis of Algorithm 1, the enumeration will take $O(|\mathcal{E}(H)|^{1.5})$, where $H$ is the subgraph induced by all the connected components.

### N. Complexity of PiTIndex Construction (Algorithm 3)

Algorithm 3 is dominated by the UF operations. Each triangle will be enumerated only once in Line 7 and Line 12, which will both cost $O(|\mathcal{E}(G)|^{1.5})$ time. For each enumeration, three UF operations are followed (Lines 10–11, Lines 15–16, and Line 18), which will each cost $O(\alpha(|\mathcal{E}(G)|))$ for a single operation ($\alpha()$ is the inverse Ackermann function) [33]. So Lines 6–12 and Lines 18–21 will cost $O(|\mathcal{E}(G)|^{1.5}\alpha(|\mathcal{E}(G)|))$ time in total. Each edge schema will be enumerated once in Line 13, so Lines 13–17 will cost $O(|\mathcal{E}(G)|)$ time. Thus, the overall complexity is $O(|\mathcal{E}(G)|^{1.5}\alpha(|\mathcal{E}(G)|))$.

### O. Case Study on Brain Networks

We use the brain network derived from a previous work [17]. These brain networks focus on resting-state functional MRI (rs-fMRI) data, which measures brain activity by detecting changes in blood flow and oxygenation. The temporal coherence in blood fluctuations suggests the brain functions as a network, with regions as vertices and functional connectivity as edges. Given the large volume of MRI data, voxel aggregation, and various preprocessing strategies are used to reduce dimensionality and minimize noise.

The dataset adopts the AAL atlas which divides the brain into 116 ROIs (Region of Interest). For each patient, the dataset formulates an undirected unweighted graph of 116 vertices. Each edge $(u, v)$ means for the individual, the two regions $u$ and $v$ correlate. We sampled 10 individuals from the datasets to form a multilayer graph. It should be pointed out that such signals are heavily subject to noise caused by different confounding factors.

Detecting functional systems in the brain is an important task in neuroscience. The previous works [6], [14] use multilayer graphs to avoid these noises. However, these methods cannot study the functional brain regions of interested individuals. They can only investigate the common functional regions across multiple individuals. In the brain, the Occipital Pole, located at the extreme posterior end of the cerebral hemisphere, near the back of the skull, is responsible for visual processing. We selected a query vertex from the Occipital Pole, as shown in the Fig. 13(a). As shown in Fig. 13(b), when performing a truss community search on a single layer, the result includes the entire brain rather than being limited to the Occipital Pole due to the presence of noise in individual fMRI. In contrast, PiTCS effectively completes this task. We can set the focus layer to be the layer formulated by the corresponding individual of interest, and set $\lambda = 2$, meaning we want to use the region connections from other individuals to filter the noise. As shown in Fig. 13(c), the community identified by PiTCS lies in the Occipital Pole. When using the same parameters as PiTCS, the community detected by FTCS fails to target the individual of interest. Meanwhile, the result also includes the entire brain rather than being limited to the Occipital Pole.