1 Foundations of Driver Reconfiguration

This section provides the rigorous graph-theoretic justification for the **Theorem (Driver Exchange Principle)** and validates the feasibility of the **Definition (Admissible Segments)** used in the main text. We establish that specific alternating paths can act as atomic operators to reconfigure driver nodes while strictly preserving the driver budget.

We first formalize several key properties of matchings and alternating paths within a single network layer, which are the building blocks of the CLAP framework. Let $\mathcal{B} = (V^+ \cup V^-, E_{\mathcal{B}})$ be the bipartite representation of a graph, and let $\mathcal{M}$ be a matching in $\mathcal{B}$ [1], [2].

Definition 1 (Matching and Saturation)**.** A *matching* is a set $\mathcal{M} \subseteq E_{\mathcal{B}}$ of pairwise vertex-disjoint edges. The saturated vertex set of $\mathcal{M}$ is

$$V_{\mathcal{M}} = \{\, x \in V^+ \cup V^- : \exists\, e \in \mathcal{M} \text{ with } x \in e \,\}.$$

A vertex x is *matched* if $x \in V_{\mathcal{M}}$; otherwise, it is *unmatched*.

In the context of structural controllability, a node $v \in V$ is a driver node if and only if its corresponding vertex $v^- \in V^-$ is unmatched (i.e., $v^- \notin V_{\mathcal{M}}$). Also see classical references on matching theory for background [1], [2]

Definition 2 (Alternating Path with Start-Parity)**.** A simple path $p = (x_0, \ldots, x_k)$ in $\mathcal{B}$ is *alternating* with respect to $\mathcal{M}$ if its edges alternate between being in $\mathcal{M}$ and not in $\mathcal{M}$. We say an alternating path starting at $a^- = x_0$ satisfies the *start-parity condition* if its first edge $\{x_0, x_1\}$ belongs to $\mathcal{M}$ if and only if a^- is a matched vertex. Formally,

$$\{x_0, x_1\} \in \mathcal{M} \iff a^- \in V_{\mathcal{M}}. \tag{1}$$

An alternating path satisfying this condition is denoted $p_{\mathcal{M}}(a^-, b^-)$.

Remark 1. This condition encapsulates two distinct physical scenarios required for reconfiguration: **Case A (Driver Elimination):** If a^- is unmatched ($a^- \notin V_{\mathcal{M}}$), the path must start with a non-matching edge ($e_1 \notin \mathcal{M}$). This initiates a process to maximize the matching; **Case B (Driver Creation):** If a^- is matched ($a^- \in V_{\mathcal{M}}$), the path must start with its incident matching edge ($e_1 \in \mathcal{M}$). This initiates a process to free a^- from its current matching.

Lemma 1 (Normalization of Start-Parity)**.** *Let $\mathcal{M}$ be a matching, and let $a^-, b^- \in V^-$. If there exists any $\mathcal{M}$-alternating path from a^- to b^-, then there also exists an $\mathcal{M}$-alternating path $p_{\mathcal{M}}(a^-, b^-)$ that satisfies the start-parity condition* (1)*.*

Proof: If $a^- \notin V_{\mathcal{M}}$, then a^- has no incident edge in $\mathcal{M}$. Consequently, any alternating path from a^- must begin with an edge not in $\mathcal{M}$, which already satisfies the condition. If $a^- \in V_{\mathcal{M}}$, let $e^\star = \{a^-, y^+\}$ be its unique incident edge in $\mathcal{M}$. For any existing alternating path q from a^- to b^-, we can construct a new alternating walk by first traversing $e^\star$ to y^+ and then following an alternating path from y^+ derived from q. After removing any induced cycles, we obtain a simple alternating path p from a^- to b^- whose first edge is $e^\star \in \mathcal{M}$, thus satisfying the condition. □

Lemma 2 (Endpoint Parity)**.** *Let $p_{\mathcal{M}}(a^-, b^-)$be an $\mathcal{M}$-alternating path starting at $a^- = x_0$ that satisfies the start-parity condition. Let $b^- = x_k$ be the endpoint. Then, the matching status of the final edge $e_k = \{x_{k-1}, x_k\}$ strictly corresponds to the matching status of b^-:*

$$e_k \in \mathcal{M} \iff b^- \in V_{\mathcal{M}}. \tag{2}$$

Equivalently, in terms of non-matching edges:

$$e_k \notin \mathcal{M} \iff b^- \notin V_{\mathcal{M}}. \tag{3}$$

Proof: Every internal vertex of an alternating path is incident to exactly one edge of $\mathcal{M}$ that is on the path. The parity of the path length determines the matching status of the final edge relative to the first. A formal analysis of the path's structure shows that the matching status of the final edge must align with the matching status of the final vertex b^- to avoid violating the matching property at b^- after a symmetric difference operation. □

Proposition 1 (Validity of Atomic Reconfiguration)**.** *Let $p_{\mathcal{M}}(s^-, t^-)$ be an $\mathcal{M}$-alternating path satisfying the start-parity condition. Construct a new set of edges $\mathcal{M}' = \mathcal{M} \triangle E(p)$. Then $\mathcal{M}'$ is a valid matching, and $|\mathcal{M}'| = |\mathcal{M}|$ (Budget Preserving). The matched set changes exactly at the endpoints: $V_{\mathcal{M}'} = (V_{\mathcal{M}} \setminus \{s^-\}) \cup \{t^-\}$ (if s^- was matched) or similar combinations based on endpoint status.*

2 Feasibility of Atomic Updates

The CLAP-S algorithm applies a discovered CLAP as a single atomic transaction. This involves a batched symmetric difference operation on the matchings of each layer. For this operation to be well-defined, the alternating paths corresponding to different segments within the same layer must not share any edges. This section proves that *shortest* CLAPs inherently satisfy this non-overlapping property.

Lemma 3 (Edge-disjointness of alternating paths in a shortest CLAP)**.** *Let $\mathcal{P} = \big((v_0 \xrightarrow{\ell_1} v_1), \ldots, (v_{k-1} \xrightarrow{\ell_k} v_k)\big)$ be a shortest CLAP. Then, within any fixed layer $\ell \in \{1, 2\}$, the witness $\mathcal{M}_\ell$-alternating paths of the segments taken in layer ℓ are pairwise edge-disjoint.*

Proof: Proof by contradiction. Suppose a shortest CLAP $\mathcal{P}$ contains two segments in layer ℓ whose witness paths share an edge e. This implies the path $\mathcal{P}$ traverses layer ℓ, leaves, and eventually returns to reuse edge e. By graph-theoretic properties of alternating paths, the cycle or detour formed between the first and second traversal of e can be excised (short-cut) without violating the parity constraints of the endpoints. Removing this redundant loop yields a new valid CLAP $\mathcal{P}'$ with strictly fewer segments or shorter total length. This contradicts the assumption that $\mathcal{P}$ is a shortest CLAP. Thus, witnesses must be edge-disjoint. □

Corollary 1 (Atomic Update Validity)**.** *Let $\mathcal{P}$ be a shortest CLAP. Let W_ℓ be the union of edges in all witness paths for segments belonging to layer ℓ. Then, the update $\mathcal{M}'_\ell = \mathcal{M}_\ell \triangle W_\ell$ is a valid matching operation (i.e., no edge conflicts occur), and it correctly implements the driver exchanges for all segments in $\mathcal{P}$ simultaneously.*

3 Structural Decomposition and Optimality Certificate

This section provides the rigorous proof for **Theorem (CLAP-or-Optimal)**. We proceed by constructing a structural decomposition of the discrepancy between the current state $(\mathcal{M}_1, \mathcal{M}_2)$ and any hypothetical alternative state $(\widehat{M}_1, \widehat{M}_2)$. We demonstrate that any reduction in the objective function implies the existence of a specific topological structure—a CLAP—in the current state.

3.1 Node Signatures and Difference Taxonomy

Let $(\mathcal{M}_1, \mathcal{M}_2)$ be the current state with driver sets $D_\ell = D(\mathcal{M}_\ell)$, and let $(\widehat{M}_1, \widehat{M}_2)$ be an arbitrary comparator state in the feasible set with driver sets $\widehat{D}_\ell = D(\widehat{M}_\ell)$. To formally track the reconfiguration of driver nodes, we introduce the state difference vector.

Definition 3 (State Difference Vector)**.** For each node $v \in V$, the signature $\delta(v) = (\delta_1(v), \delta_2(v)) \in \{-1, 0, +1\}^2$ is defined by the change in driver status for each layer:

$$\delta_\ell(v) = \mathbf{1}_{\{v \in \widehat{D}_\ell\}} - \mathbf{1}_{\{v \in D_\ell\}}. \tag{4}$$

Here, $\delta_\ell(v) = -1$ denotes the loss of driver status ("releasing" a driver), $\delta_\ell(v) = +1$ denotes the gain of driver status ("recruiting" a driver), and 0 indicates no change.

The budget-constrained nature of the problem imposes a strict conservation law on these signatures.

Lemma 4 (Budget Conservation Law)**.** *Since the driver budgets k_1 and k_2 are structural invariants (i.e., $|D_\ell| = |\widehat{D}_\ell| = k_\ell$), the net change in driver status within each layer must sum to zero:*

$$\sum_{v \in V} \delta_\ell(v) = 0, \quad \textit{for } \ell \in \{1, 2\}. \tag{5}$$

While the signature $\delta(v)$ captures the flow of driver assignments (constrained by the conservation law), the impact on the objective function $\Delta = |\mathrm{DD}_1| + |\mathrm{DD}_2|$ depends on the node's **initial classification**. A node contributes to optimization if and only if it transitions from a difference set (DD) to a consistent set (CMS or CDS).

Table 1 presents the taxonomy of node roles, combining the signature and initial state. We designate nodes that reduce Δ as **Eliminators**. Note that an Eliminator can function either by "releasing" a driver (becoming CMS) or by "absorbing" a difference (becoming CDS).

TABLE 1
Taxonomy of Node Roles. The optimization goal is to maximize the number of "Eliminators" (Impact -1). The specific type of Eliminator depends on whether the node loses a driver status (Release) or gains a driver status to match the other layer (Absorb).

Role	Initial State	Signature δ	Transition Description	Δ-Impact
Difference Eliminators (Endpoints of Improving Paths)				
L1-Release (L_{rel})	DD_1	$(-1, 0)$	$\mathrm{DD}_1 \to$ CMS (Lose L1)	$-\mathbf{1}$
L2-Release (R_{rel})	DD_2	$(0, -1)$	$\mathrm{DD}_2 \to$ CMS (Lose L2)	$-\mathbf{1}$
L1-Absorb (L_{abs})	DD_2	$(+1, 0)$	$\mathrm{DD}_2 \to$ CDS (Gain L1)	$-\mathbf{1}$
L2-Absorb (R_{abs})	DD_1	$(0, +1)$	$\mathrm{DD}_1 \to$ CDS (Gain L2)	$-\mathbf{1}$
Difference Generators (Penalties)				
L1-Create (N_1)	CMS	$(+1, 0)$	CMS $\to \mathrm{DD}_1$	$+\mathbf{1}$
L2-Create (N_2)	CMS	$(0, +1)$	CMS $\to \mathrm{DD}_2$	$+\mathbf{1}$
L1-Split	CDS	$(-1, 0)$	CDS $\to \mathrm{DD}_2$	$+\mathbf{1}$
L2-Split	CDS	$(0, -1)$	CDS $\to \mathrm{DD}_1$	$+\mathbf{1}$
Neutrals (Internal Path Nodes)				
Consistent Flip	CMS/CDS	$(\pm 1, \pm 1)$	CMS $\leftrightarrow$ CDS	0
Swap	$\mathrm{DD}_1/\mathrm{DD}_2$	$(\mp 1, \pm 1)$	$\mathrm{DD}_1 \leftrightarrow \mathrm{DD}_2$	0

Proposition 2 (Contribution Analysis)**.** *The total reduction in difference mass is the net count of Eliminators minus Generators.*

Remark 2 (Diversity of CLAP Endpoints). A CLAP connects a node $s \in \mathrm{DD}_1$ to a node $t \in \mathrm{DD}_2$. Depending on which layer the path segments initiate/terminate in, the endpoints manifest as different Eliminator types. For example:

- If a CLAP starts in Layer 2 at $s \in \mathrm{DD}_1$, then s gains L2 status ($\delta = (0, +1)$) and becomes CDS. This is an **L2-Absorb** role.
- If a CLAP ends in Layer 1 at $t \in \mathrm{DD}_2$, then t gains L1 status ($\delta = (+1, 0)$) and becomes CDS. This is an **L1-Absorb** role.

Regardless of the specific combination of Release or Absorb types, the pair (s, t) contributes $(-1) + (-1) = -2$ to the difference mass, confirming the Δ-reducing property of CLAPs.

3.2 Construction of the Layer-Labeled Meta-Graph

While the node taxonomy in Table 1 defines the functional role of individual nodes, the feasibility of transitions is constrained by the network topology. We now construct a **Layer-Labeled Meta-Graph** $\mathcal{K}$ to capture the structural connectivity between these nodes.

The construction leverages the standard symmetric difference property of matchings. For each layer $\ell \in \{1, 2\}$, consider the bipartite symmetric difference graph $H_\ell = (V^+ \cup V^-, \mathcal{M}_\ell \triangle \widehat{M}_\ell)$. This graph consists of disjoint components, which are either alternating cycles or alternating paths.

Remark 3 (Relevance of Path Components). Alternating cycles in H_ℓ do not alter the set of saturated vertices in V^-; hence, they do not change the driver set D_ℓ. Only alternating path components in H_ℓ with endpoints in V^- correspond to a shift in driver locations (one endpoint loses driver status while the other gains it). We project these paths onto the node set V.

Definition 4 (Layer-Labeled Meta-Graph)**.** The meta-graph $\mathcal{K} = (V, \mathcal{E})$ is an undirected multigraph constructed as follows: For each layer $\ell \in \{1, 2\}$ and for every path component in H_ℓ that connects two vertices $u^-, v^- \in V^-$, add an undirected edge $\{u, v\}$ to $\mathcal{E}$ with the label $\lambda(\{u, v\}) = \ell$.

This graph $\mathcal{K}$ creates a direct topological link between the node roles defined in subsection 3.1.

Lemma 5 (Structural Properties of $\mathcal{K}$)**.** *The meta-graph $\mathcal{K}$ satisfies the following properties:*

1) ***Maximum Degree:*** *Each node $v \in V$ has at most one incident edge labeled 1 and at most one incident edge labeled 2. Thus, the maximum degree is 2.*
2) ***Component Types:*** *Every connected component of $\mathcal{K}$ is either a simple cycle or a simple path.*
3) ***Alternating Labels:*** *Along any path or cycle in $\mathcal{K}$, the edge labels must alternate (e.g., $1, 2, 1, 2, \ldots$).*

Proof: A node v corresponds to a unique vertex $v^- \in V^-$ in the bipartite graph. In any single layer ℓ, v^- can be an endpoint of at most one path component in H_ℓ (since $\mathcal{M}_\ell$ and $\widehat{M}_\ell$ are matchings). Therefore, $\deg_{\mathcal{K}}(v) \leq 2$. The alternating property follows because a node cannot be an endpoint for two distinct paths in the same layer ℓ simultaneously. □

To relate this static structure to the optimization process, we assign a direction to each edge based on the flow of driver status.

Definition 5 (Admissible Orientation)**.** For each edge $\{u, v\} \in \mathcal{E}$ with label ℓ, we direct the edge from the node releasing the driver status to the node absorbing it. Formally, we orient the edge as $u \xrightarrow{\ell} v$ if:

$$\delta_\ell(u) = -1 \quad \text{and} \quad \delta_\ell(v) = +1. \tag{6}$$

By the definition of alternating paths in H_ℓ, exactly one endpoint must be in D_ℓ (where $\delta_\ell = -1$) and the other in $\widehat{D}_\ell$ (where $\delta_\ell = +1$). Thus, this orientation is unique and well-defined for every edge.

Let $\overrightarrow{\mathcal{K}}$ denote the directed meta-graph. The edges in $\overrightarrow{\mathcal{K}}$ correspond precisely to the **Admissible Segments** defined in the main text.

Corollary 2 (Equivalence to CLAP)**.** *A directed path in $\overrightarrow{\mathcal{K}}$ that originates at a node $s \in DD_1$ and terminates at a node $t \in DD_2$ is structurally equivalent to a Cross-Layer Augmenting Path (CLAP). The label-alternating property of $\overrightarrow{\mathcal{K}}$ ensures the layer-alternation condition of the CLAP.*

To provide a concrete visualization of the structural correspondence between the abstract paths in $\overrightarrow{\mathcal{K}}$ and the physical reconfigurations in the bipartite layers, we illustrate CLAP structures of varying lengths in Fig. 1. Panel A demonstrates the atomic driver exchange (justified in subsection 1), while Panels B-D depict how these exchanges are chained through relay nodes to form valid CLAPs.

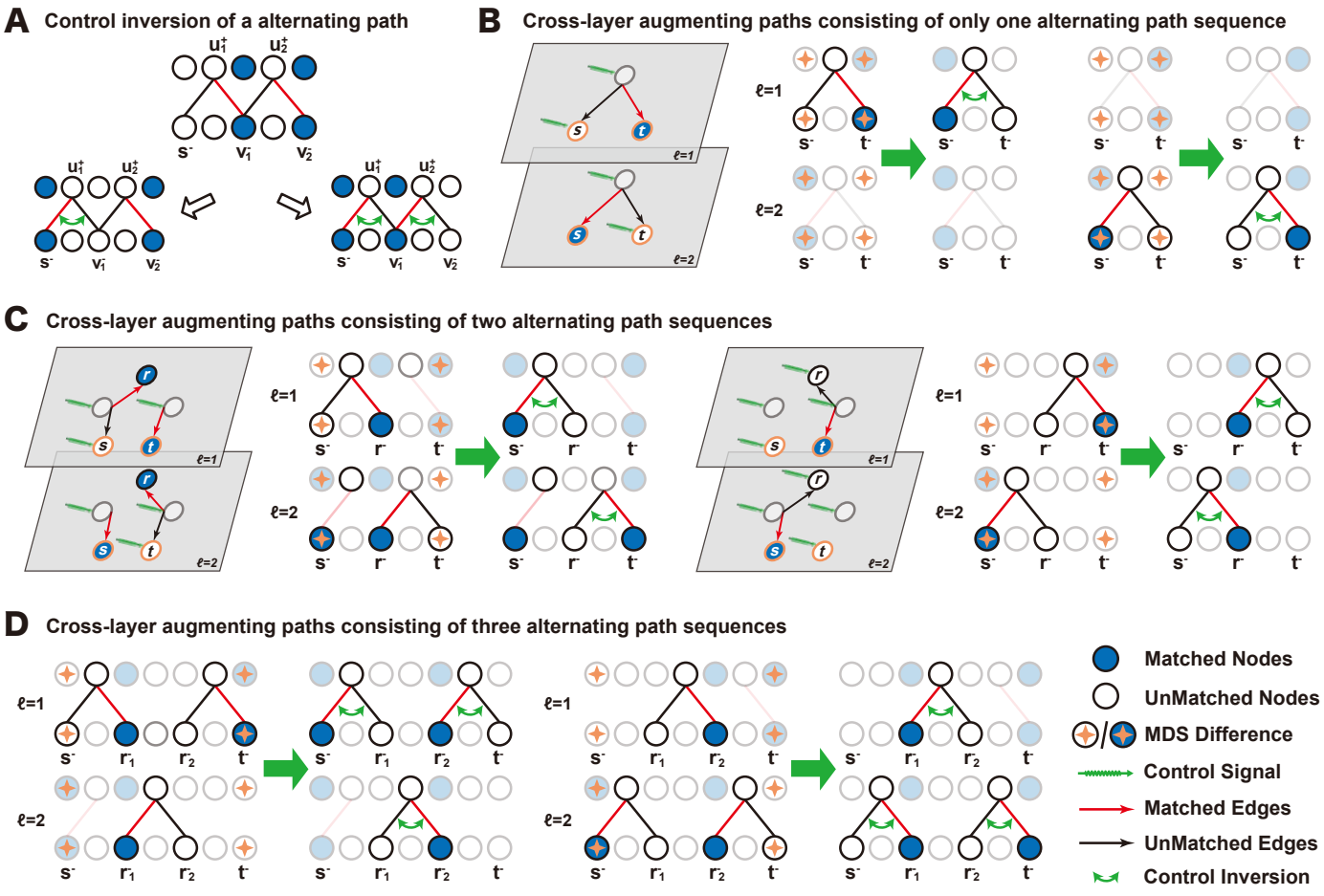

Fig. 1. **Mechanistic Visualization of Cross-Layer Augmenting Paths.** (A) **Atomic Reconfiguration:** Illustration of the **Theorem (Driver Exchange Principle)** in the main text. A symmetric difference operation along an alternating path swaps a driver node (s) with a non-driver node (v_1). (B) **1-Segment CLAP:** A direct connection between an L1-Eliminator and an L2-Eliminator. No relay nodes are involved. This corresponds to a single edge in the meta-graph $\overrightarrow{\mathcal{K}}$. (C) **2-Segment CLAP:** Connection via a single Relay Node (r). As the path traverses $s \xrightarrow{1} r \xrightarrow{2} t$, the relay node r flips its status in both layers (e.g., from consistent non-driver to consistent driver), confirming the internal neutrality property (Lemma 6). (D) **3-Segment CLAP:** Connection via two Relay Nodes (r_1, r_2) with alternating layer segments ($\ell = 1, 2, 1$). In all cases (B-D), the operation results in the removal of one difference driver from Layer 1 (s^-) and one from Layer 2 (t^-), reducing the total Difference Mass Δ by exactly 2.

TABLE 2
Detailed Algebraic Balance and Structural Invariants of Representative CLAP Configurations. This table provides a node-by-node decomposition of the state-difference vector (δ_1, δ_2) and the resulting impact on the objective function for the path structures shown in Fig. 1. **Budget Conservation:** The column-wise sums ($\sum \delta_1 = 0$ and $\sum \delta_2 = 0$) demonstrate that each layer-wise reconfiguration adheres to the strict budget conservation law established in Section 3.1, ensuring that structural controllability is maintained with exactly k_ℓ driver nodes per layer. **Internal Neutrality:** The algebraic signatures for relay nodes (r) confirm that they undergo consistent status flips (e.g., $(+1, +1)$ for $CMS \to CDS$), resulting in a net Δ-impact of 0. This verifies Lemma 6, showing that the path's interior does not dissipate optimization energy. **Endpoint Concentration:** In all cases, the optimization gain ($\sum$ Impact $= -2$) is concentrated at the endpoints $\{s, t\}$, which transition from difference-driver sets to consistent sets. This confirms that the CLAP acts as a length-invariant operator for reducing duplex control redundancy.

Configuration	Node	Role in Path	Signature (δ_1, δ_2)	Δ-Impact	Layer Balance	Total Gain
Case B	s	L1-Source ($DD_1 \to CMS$)	$(-1, 0)$	-1	$\sum \delta_1 = 0$	-2
(1-Segment)	t	L1-Absorber ($DD_2 \to CDS$)	$(+1, 0)$	-1	$\sum \delta_2 = 0$	
Case C	s	L1-Source ($DD_1 \to CMS$)	$(-1, 0)$	-1		
(2-Segments)	r	Relay ($CMS \to CDS$)	$(+1, +1)$	0	$\sum \delta_1 = 0$	-2
	t	L2-Release ($DD_2 \to CMS$)	$(0, -1)$	-1	$\sum \delta_2 = 0$	
Case D	s	L1-Source ($DD_1 \to CMS$)	$(-1, 0)$	-1		
(3-Segments)	r_1	Relay ($CMS \to CDS$)	$(+1, +1)$	0		
	r_2	Relay ($CDS \to CMS$)	$(-1, -1)$	0	$\sum \delta_1 = 0$	-2
	t	L1-Absorber ($DD_2 \to CDS$)	$(+1, 0)$	-1	$\sum \delta_2 = 0$	

3.3 Component-wise Analysis of Objective Function

We now combine the node taxonomy (subsection 3.1) and the meta-graph structure (subsection 3.2) to analyze the total change in the objective function. Let Δ and $\widehat{\Delta}$ be the difference mass of the current state and the comparator state, respectively.

The global change $\Delta_{change} = \widehat{\Delta} - \Delta$ can be decomposed into the sum of contributions from the disjoint connected components of the meta-graph $\mathcal{K}$. Let $\mathcal{C}$ be the set of these components (paths and cycles).

$$\Delta_{change} = \sum_{C \in \mathcal{C}} \text{Gain}(C),$$
$$\text{where Gain}(C) = \sum_{v \in V(C)} \text{Impact}(v).$$

Here, Impact(v) is the Δ-Impact value defined in Table 1. We analyze the gain of each component type.

Lemma 6 (Neutrality of Internal Nodes)**.** *Any node v that is strictly internal to a path component or belongs to a cycle component in $\mathcal{K}$ has a net Δ-Impact of 0.*

Proof: Let v be an internal node (or cycle node). By the construction of $\mathcal{K}$ (Definition 4) and Lemma 5, edges are derived exclusively from path components of the symmetric difference. Since matchings enforce a maximum degree of 1 in the bipartite graph, the projected node v can have at most one incident edge per layer.

v has a degree of 2, and its incident edges must have alternating labels. This implies v is incident to exactly one edge labeled 1 and one edge labeled 2.

Physically, an edge labeled ℓ corresponds to a change in driver status in layer ℓ (i.e., $\delta_\ell(v) \neq 0$). Since v is connected in both layers, its signature $\delta(v) = (\delta_1, \delta_2)$ must have non-zero entries for both components. The possible signatures are restricted to $\{(\pm 1, \pm 1), (\pm 1, \mp 1)\}$. We analyze the impact on Δ for these two cases based on the initial state implied by the signature:

- **Case 1: Consistent Flip** ($\delta_1 = \delta_2 = \pm 1$). This corresponds to the "Neutrals: Consistent Flip" row in Table 1.
  - If $\delta = (+1, +1)$, the node transitions from CMS $\to$ CDS.
  - If $\delta = (-1, -1)$, the node transitions from CDS $\to$ CMS.

  In both sub-cases, the node belongs to a consistent set (CMS or CDS) in both the initial and final states. Its contribution to the difference mass Δ is 0 before and 0 after. Thus, Impact$(v) = 0$.
- **Case 2: Difference Swap** ($\delta_1 = -\delta_2 = \pm 1$). This corresponds to the "Neutrals: Swap" row in Table 1.
  - If $\delta = (-1, +1)$, the node transitions from $\text{DD}_1 \to \text{DD}_2$.
  - If $\delta = (+1, -1)$, the node transitions from $\text{DD}_2 \to \text{DD}_1$.

  In both sub-cases, the node belongs to a difference set in both the initial and final states. Its contribution to Δ is 1 before and 1 after. The net change is $1 - 1 = 0$. Thus, Impact$(v) = 0$.

Since all possible signatures for degree-2 nodes yield zero impact, the lemma holds. □

Corollary 3 (Zero Gain for Cycles)**.** *For any cycle component $C_{cycle} \in \mathcal{C}$, all nodes are internal (degree 2). Therefore, Gain$(C_{cycle}) = 0$. Cycles in the symmetric difference graph represent reconfiguration redundancies that do not affect the optimization objective.*

By Lemma 6, the sum of impacts for all internal nodes is zero. Thus, Gain(C_{path}) = Impact(s) + Impact(t). In the meta-graph $\mathcal{K}$, path endpoints have a degree of 1. A node v with degree 1 is incident to an edge from exactly one layer (say ℓ). This implies $\delta_\ell(v) \neq 0$ and $\delta_{3-\ell}(v) = 0$. Consequently, the signature $\delta(v)$ must be of the form $(\pm 1, 0)$ or $(0, \pm 1)$. Referring to Table 1, nodes with such signatures fall strictly into two categories:

- **Eliminators** (Impact -1): e.g., $(-1, 0)$ or $(+1, 0)$ starting from DD_2.
- **Generators** (Impact $+1$): e.g., $(+1, 0)$ starting from CMS.

Crucially, nodes with Impact 0 (Neutrals) require non-zero δ in both layers (degree 2) and thus cannot be endpoints. Therefore, the possible sums for Impact(s) + Impact(t) are restricted to sums of ± 1:

- $(-1) + (-1) = -2$: Both are Eliminators (Optimization).
- $(-1) + (+1) = 0$: One Eliminator, one Generator (No net change).
- $(+1) + (+1) = +2$: Both are Generators (Regression).

This proves that the gain is even and that Gain $= -2$ requires two Eliminators.

Proposition 3 (Path Gain Characterization)**.** *For a path component $C_{path} \in \mathcal{C}$ with endpoints s and t, the total gain is determined solely by its endpoints. Moreover, the gain must be an even integer:*

$$\textit{Gain}(C_{path}) = \textit{Impact}(s) + \textit{Impact}(t) \in \{-2, 0, 2\}. \tag{7}$$

*A strict reduction in difference mass (Gain $= -2$) occurs if and only if both endpoints are **Eliminators**.*

Corollary 4 (The CLAP Connection)**.** *A path component that contributes Gain $= -2$ corresponds exactly to a structure connecting two Eliminator endpoints. As established in Remark 2 and Corollary 2, any such path connecting a Layer-1 difference node to a Layer-2 difference node (or vise versa) via alternating segments constitutes a CLAP.*

To provide a concrete realization of the aforementioned theory, we evaluate the algebraic balance of the typical CLAP structures visualized in Fig. 1. As summarized in Table 2, this arithmetic breakdown highlights the interplay between budget conservation and optimization gain.

The analysis reveals two fundamental patterns: (i) **Internal Neutrality**: Regardless of the path length, all relay nodes (r_i) exhibit a signature where $\delta_1 = \delta_2$. This confirms Lemma 6, showing that relay status flips (e.g., $CMS \leftrightarrow CDS$) do not alter the difference mass Δ. (ii) **Endpoint Concentration**: The total reduction $\Delta_{change} = -2$ is strictly concentrated at the endpoints $\{s, t\}$. This confirms that the search for an optimal union is mathematically equivalent to identifying these endpoint pairs through label-alternating paths in the meta-graph. The consistency across lengths 1, 2, and 3 demonstrates that the CLAP is an invariant operator for budget-preserving union contraction.

3.4 Proof of the CLAP-Stability Theorem

We now assemble the results from the previous subsections to prove the main theoretical guaranty of our paper: **Theorem (CLAP-or-Optimal)**. Recall that a state is defined as *CLAP-stable* if it contains no Cross-Layer Augmenting Paths.

Theorem 1 (CLAP-or-Optimal)**.** *A state $(\mathcal{M}_1, \mathcal{M}_2) \in \mathbb{M}_1(k_1) \times \mathbb{M}_2(k_2)$ minimizes the UDS size $|U|$ (or equivalently, the difference mass Δ) if and only if it is CLAP-stable.*

Proof: The proof consists of two directions:

Direction 1: Optimal $\implies$ CLAP-stable (Sufficiency) We prove the contrapositive: If a state is *not* CLAP-stable, then it is *not* optimal. Assume there exists at least one CLAP in the current state $(\mathcal{M}_1, \mathcal{M}_2)$. By **Theorem (CLAP Gain Theorem)** in the main text, applying this path yields a valid new state $(\mathcal{M}_1', \mathcal{M}_2')$ with preserved budgets such that:

$$|U(\mathcal{M}_1', \mathcal{M}_2')| = |U(\mathcal{M}_1, \mathcal{M}_2)| - 1. \tag{8}$$

Since a state with a strictly smaller objective value exists, the original state $(\mathcal{M}_1, \mathcal{M}_2)$ cannot be a minimum. Thus, any optimal state must be devoid of CLAPs (i.e., CLAP-stable).

Direction 2: CLAP-stable $\implies$ Optimal (Necessity) We prove this by contradiction. Assume the state $(\mathcal{M}_1, \mathcal{M}_2)$ is CLAP-stable but *not* optimal. Since it is not optimal, there exists some other feasible state $(\widehat{M}_1, \widehat{M}_2)$ with a strictly smaller difference mass: $\widehat{\Delta} < \Delta$. Consequently, the total change is negative:

$$\Delta_{change} = \widehat{\Delta} - \Delta < 0. \tag{9}$$

We construct the layer-labeled meta-graph $\mathcal{K}$ between $(\mathcal{M}_1, \mathcal{M}_2)$ and $(\widehat{M}_1, \widehat{M}_2)$ as defined in subsection 3.2. By the decomposition established in subsection 3.3, the total change is the sum of component gains:

$$\Delta_{change} = \sum_{C \in \mathcal{C}} \text{Gain}(C) < 0. \tag{10}$$

For the sum to be strictly negative, there must exist at least one component $C^\star \in \mathcal{C}$ such that $\text{Gain}(C^\star) < 0$. According to Proposition 3, the gain of any component must be in $\{-2, 0, 2\}$. Therefore, we must have:

$$\text{Gain}(C^\star) = -2. \tag{11}$$

By the Corollary 4, a component with Gain $= -2$ is structurally isomorphic to a CLAP connecting two Eliminator endpoints. This implies that a CLAP exists in the state $(\mathcal{M}_1, \mathcal{M}_2)$, which directly contradicts the assumption that the state is CLAP-stable. Therefore, the assumption that a non-optimal state can be CLAP-stable is false. A CLAP-stable state must be globally optimal. □

Remark 4 (Analogy to Classical Matching Theory). This result can be viewed as a generalization of *Berge's Lemma* [3] to the problem of coordinated duplex control. Just as a matching is maximum if and only if it admits no augmenting paths, a duplex control configuration is union-minimal if and only if it admits no cross-layer augmenting paths. The meta-graph decomposition serves the same role as the symmetric difference argument in standard matching theory, identifying the topological certificate of optimality.

4 Auxiliary Algorithmic Procedures

The main CLAP-S algorithm relies on several subroutines. Their pseudocode is provided in Algorithm 4. These functions handle the low-level graph traversal and matching manipulations required to find and apply CLAPs.

Algorithm — Auxiliary functions

Procedure ALTREACH($\mathcal{S} \mid \mathcal{M}_\ell$)

Purpose. Given a *set of sources* $\mathcal{S} \subseteq V$ and matching $\mathcal{M}_\ell$ in layer ℓ, return the reachable set R together with an origin map $\omega : R \to \mathcal{S}$ such that each $v \in R$ has a witnessed ℓ-alternating path from $\omega(v)$ to v.

$$R = \{\, v \in V \setminus \mathcal{S} \;:\; \exists \text{ an } \ell\text{-alternating path from some } u \in \mathcal{S} \text{ to } v \,\}.$$

Implementation. Run a single *multi-source* BFS in the $\mathcal{M}_\ell$-alternating residual graph, seeding all $u \in \mathcal{S}$ simultaneously and respecting the start-parity for layer ℓ. Propagate a source label so that each discovered node v stores $\omega(v) \in \mathcal{S}$ (the frontier source that first reaches it).

return (R, ω)

Procedure ALTPATH($u, v \mid \mathcal{M}_\ell$)

Purpose. Return a *shortest* $\mathcal{M}_\ell$-alternating path p from u to v under the layer-ℓ move pattern, or $\perp$ if no such path exists. One may realize this with a BFS that records parent pointers; details are omitted here.

return p or $\perp$

Procedure UNROLLSEGMENTS($(u, \ell), v, \text{pred}, \text{predLayer}$)

Purpose. Recover the segment *endpoints* (without witnesses) by backtracking predecessor pointers.

$seg \leftarrow [\,]$; $(x, \eta) \leftarrow (u, \ell)$
while $\text{pred}[x, \eta] \neq \text{nil}$ **do**
  $p \leftarrow \text{pred}[x, \eta]$; $\pi \leftarrow \text{predLayer}[x, \eta]$
  append (p, x, π) to seg; $(x, \eta) \leftarrow (p, \pi)$
end while
$seg \leftarrow \text{reverse}(seg)$
append (u, v, ℓ) to seg ▷ final segment to the target
return seg

Procedure SYMDIFF($\mathcal{M}_\ell, E(p)$)

return $(\mathcal{M}_\ell \setminus E(p)) \cup (E(p) \setminus \mathcal{M}_\ell)$

Procedure APPLYATOMICUPDATE($seg, \mathcal{M}_1, \mathcal{M}_2$)

Purpose. Execute a CLAP given by segment endpoints, updating matchings sequentially.

for (x, y, ℓ) in seg in order **do**
  if $\ell = 1$ **then**
    $p \leftarrow \text{ALTPATH}(x, y \mid \mathcal{M}_1)$
    $\mathcal{M}_1 \leftarrow \text{SYMDIFF}(\mathcal{M}_1, E(p))$
  else
    $p \leftarrow \text{ALTPATH}(x, y \mid \mathcal{M}_2)$
    $\mathcal{M}_2 \leftarrow \text{SYMDIFF}(\mathcal{M}_2, E(p))$
  end if
end for
return $(\mathcal{M}_1, \mathcal{M}_2)$

5 REAL-WORLD NETWORKS

To assess the CLAP-S algorithm's applicability in practical scenarios, we curated a dataset of real-world systems from a multilayer network collection [4]. These systems often comprise more than two layers and may have different node sets in each layer. For our experiments, we constructed duplex networks by selecting two functionally significant layers from each system and considering all the nodes present in any layer of the two layers. This ensures a consistent node set V across the duplex structure. The selected networks span various domains. For instance, genetic networks map biological interactions, from which we constructed duplex layers representing distinct mechanisms such as direct versus physical association interactions from the BioGRID database [5], or positive versus negative genetic interactions from studies like the Yeast Landscape [6]. Similarly, neuronal networks include connectome data, such as the *C. elegans* connectome [7], where layers distinguish between electrical junctions and chemical synapses. In the realm of social networks, we analyzed data capturing online interactions, primarily from Twitter [8], [9], where layers differentiate between user actions such as retweets and mentions. Finally, human relationship networks map interpersonal ties within closed communities [10], [11], [12], with layers representing different social connections, such as friendship, advice, or co-working relationships. The properties of these curated real-world duplex networks are summarized in Table 3.

6 CALIBRATION OF RSU SAMPLING BUDGET

The performance of the stochastic baseline RSU is intrinsically linked to its sampling budget S. To ensure a fair and computationally feasible comparison in subsequent experiments, we conduct a sensitivity analysis on S.

6.1 Sensitivity to Sampling Budget S

We evaluate the optimization performance of RSU-S for $S \in \{20, 50, 100, 200\}$ on ER and BA duplex networks ($N = 1000$). As shown in Fig. 2, RSU exhibits a marginal improvement in ΔN_D as the sampling budget increases. However, this

TABLE 3
Properties of the real-world duplex networks used in experiments. For each system, two representative layers were selected to form a duplex network on their common set of nodes. The node count N represents the size of this common set. The average degree $\langle k \rangle$ is calculated with respect to the nodes present in the given layer.

Network Type	Network Name	Nodes (N)	Nodes in Layer 1	Nodes in Layer 2	Avg. Degree ($\langle k \rangle$)	$\langle k_1 \rangle$	$\langle k_2 \rangle$
Genetic & Neuronal	Arabidopsis	6903	5493	2859	5.29	5.05	3.09
	Celegans	3191	3126	239	3.68	3.56	2.62
	Drosophila	8060	7356	2851	9.20	6.55	9.11
	HumanHIV1	994	758	380	2.62	2.29	2.28
	SacchPomb	2622	971	2402	7.01	3.47	6.25
	Rattus	2593	2035	1017	3.17	2.96	2.15
	YeastLandscape	4455	4422	4432	85.89	30.41	55.99
	CelegansConnectome	275	253	260	19.42	8.15	12.61
Social	Cannes	438513	340349	233735	4.14	2.92	3.52
	MLKing	327660	288738	79070	2.31	2.02	2.20
	MoscowAthletics	88777	74688	46821	4.45	2.81	3.95
	NYClimate	102416	94574	50054	6.75	4.52	5.26
	NBAFinals	747690	690288	321515	4.39	3.06	3.64
	Sanremo	55871	49904	34564	10.95	8.50	5.43
	UCLFinal	676654	590337	293682	3.91	3.00	2.98
	GravitationalWaves	361846	321307	118282	3.40	2.76	2.89
Human Relationship	KrackhardtHighTech-f&a	21	21	21	27.81	18.10	9.71
	KrackhardtHighTech-f&r	21	21	21	11.62	9.71	1.90
	LazegaLawFirm-f&a	71	71	69	41.32	25.13	16.67
	LazegaLawFirm-f&c	71	69	71	47.30	16.67	31.10
	PhysiciansInnovation-f&a	238	215	228	8.29	4.47	4.44
	PhysiciansInnovation-f&d	241	231	228	8.89	4.89	4.44

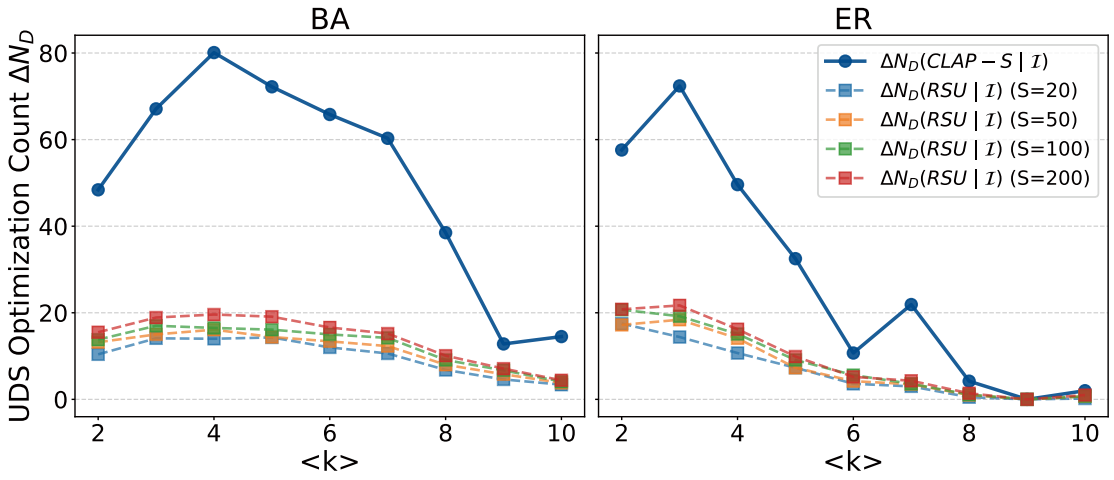

Fig. 2. Sensitivity of RSU optimization performance to the sampling budget S. Dashed lines represent RSU-S for $S \in \{20, 50, 100, 200\}$, while the solid blue line denotes CLAP-S. The results demonstrate that increasing S provides diminishing returns and remains significantly inferior to the deterministic shortest-path search of CLAP-S.

improvement follows a clear law of diminishing returns; the performance gap between $S = 100$ and $S = 200$ is significantly smaller than that between $S = 20$ and $S = 50$.

Crucially, even with a massive budget of $S = 200$ (representing $40,000$ pairwise union comparisons), RSU still fails to reach the optimization levels achieved by CLAP-S. This gap stems from the combinatorial explosion of the MDS space; randomized sampling is statistically unlikely to identify the precise driver nodes required for cross-layer alignment, whereas CLAP-S navigates this space through a deterministic, guided search.

6.2 Standardization of the Baseline

While increasing S slightly improves the baseline quality, it imposes a quadratic increase in computational complexity for union comparisons—$O(S^2 \cdot N)$—plus the linear cost of $2S$ maximum matching executions. In our tests, RSU-200 required over an order of magnitude more time than CLAP-S while remaining sub-optimal.

Based on these observations, we select $S = 20$ as our standard baseline for the remainder of this study. This choice represents a well-calibrated trade-off: it is sufficient to capture the representative performance of undirected sampling while remaining within a practical computational envelope.

7 Control Energy Analysis

While CLAP-S is designed to minimize the structural hardware cost (the number of driver nodes), a critical question remains regarding the dynamical implications of this optimization: Does the removal of redundant driver nodes significantly increase the control effort required to steer the system? To address this, we conducted a comprehensive control energy analysis using the Controllability Gramian framework.

7.1 Experimental Setup

The experiment was conducted on synthetic networks ($N = 100$, ER and BA, 200 instances) to ensure numerical precision and stability. The setup followed these steps:

Dynamics Generation: For each network topology, the non-zero entries of A were drawn from a standard normal distribution $\mathcal{N}(0, 1)$. To ensure system stability (a prerequisite for the infinite-horizon Gramian), we shifted the spectrum of A such that all eigenvalues satisfy $\mathrm{Re}(\lambda_i) < -1$.

Input Configuration: For a given instance, we constructed two input matrices: B_{CLAPS} (using drivers identified by our algorithm) and B_{RSU} (using the best baseline result). Both matrices correspond to the same driver budget constraints k_ℓ.

Energy Metric: We calculated the infinite-horizon Controllability Gramian W by solving the Lyapunov equation:

$$AW + WA^\top + BB^\top = 0. \tag{12}$$

The average control energy required to move the system to a random state is proportional to the trace of the inverse Gramian [13]. We define the logarithmic energy metric as:

$$E = \log_{10}\left(\mathrm{Trace}(W^{-1})\right). \tag{13}$$

7.2 Results and Discussion

The results, summarized in Fig. 3, provide strong evidence that CLAP-S achieves hardware efficiency without compromising physical controllability.

Comparable Energy Magnitude (Fig. 3a): The distribution of control energy for CLAP-S overlaps significantly with that of the random sampling baseline (RSU). The median energy values are nearly identical across both ER and BA topologies. This indicates that the "redundant" drivers removed by CLAP-S are typically peripheral nodes that contribute little to the Gramian's dominant eigenmodes.

Bounded Optimization Penalty (Fig. 3b): The distribution of the energy difference ΔE confirms that the penalty for structural optimization is mild and bounded. The probability density is concentrated in the range $[-0.5, 1.5]$ on the logarithmic scale. This implies that, in most cases, the optimized configuration requires less than one order of magnitude of additional energy compared to a random configuration—a negligible cost, given that Gramian values typically span dozens of orders of magnitude.

Efficient Engineering Trade-off (Fig. 3c): The trade-off analysis reveals a favorable mechanism: significant reductions in the union size (up to 10-12 nodes saved) are correlated with only marginal increases in energy. Crucially, we observe no outliers where small hardware savings lead to a disproportionately large energy increase.

In conclusion, these dynamical simulations confirm that the minimal union sets identified by CLAP-S remain physically feasible for control, validating the algorithm's applicability in realistic engineering scenarios where both hardware costs and actuation energy are concerns.

8 Stability Analysis under Structural Perturbation

A common concern in network control optimization is whether the pursuit of a minimal driver set leads to *brittle* solutions—configurations that are highly sensitive to small topological changes or noise. To address this, we evaluated the *structural stability* of the driver sets identified by CLAP-S compared to the RSU baseline under random edge removal.

8.1 Experimental Setup

We define robustness as the *persistence* of the driver configuration when the network topology is perturbed. The experiment was conducted on ER and BA duplex networks ($N = 100$) using the following procedure:

Reference Identification: For the original network G_0, we identified the driver sets U_0 using both CLAP-S and RSU.

Random Perturbation: We generated perturbed networks G_p by randomly removing a fraction p of edges, with p varying from 0 to 0.5.

Metric: We recalculated the new driver sets U_p on the perturbed graphs and measured the Jaccard Similarity Coefficient $J(p) = \frac{|U_0 \cap U_p|}{|U_0 \cup U_p|}$. A higher $J(p)$ indicates that the optimized solution is stable and reuses a larger proportion of the original actuators despite the noise.

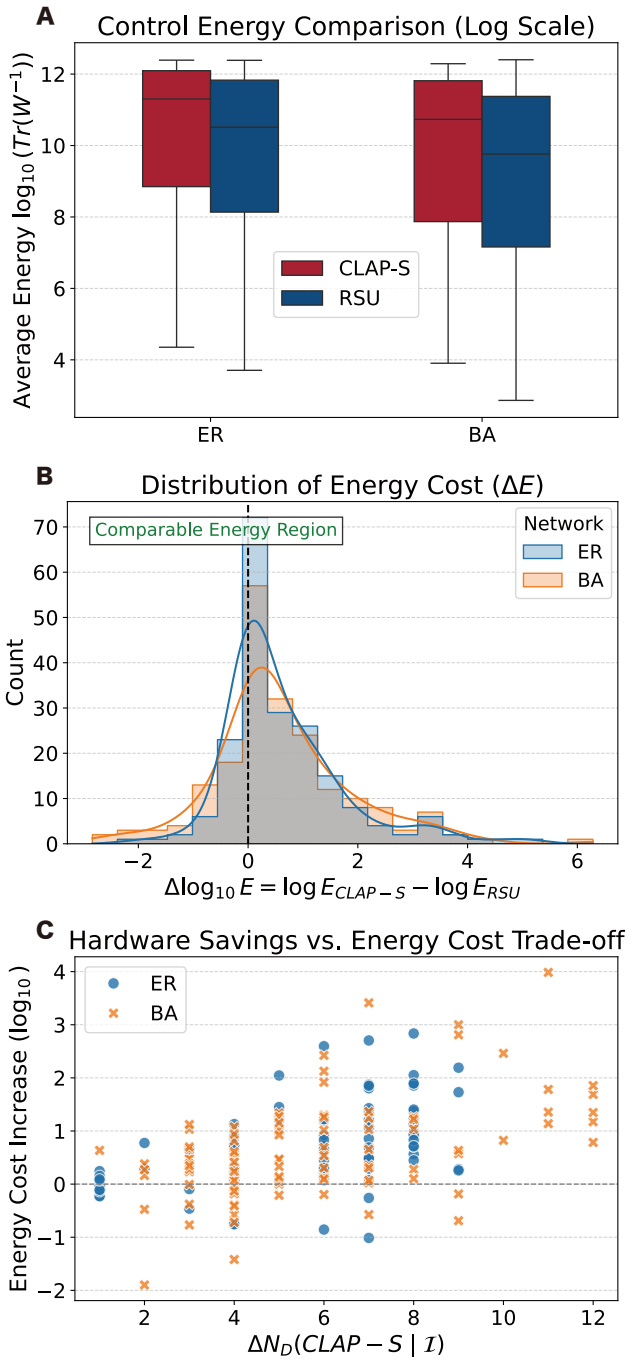

Fig. 3. **Control Energy Validation comparing CLAP-S and RSU.** (a) **Global Distribution:** Boxplots of the average control energy ($\log_{10} \mathrm{Tr}(W^{-1})$) show that the energy requirements for CLAP-S (red) and RSU (blue) are statistically comparable and lie within the same order of magnitude. The variance is dominated by the random weights of A rather than the driver selection strategy. (b) **Penalty Distribution:** The histogram of the energy difference ($\Delta E = E_{\text{CLAPS}} - E_{\text{RSU}}$) is unimodal and centered near zero. The "Comparable Energy Region" indicates that for the vast majority of networks, the optimization incurs a negligible energy penalty ($|\Delta \log_{10} E| < 1$). (c) **Cost-Benefit Trade-off:** A scatter plot of hardware savings (x-axis) versus energy cost increase (y-axis). The absence of data points in the upper-left region confirms that CLAP-S effectively identifies low-cost optimization opportunities without imposing catastrophic energy penalties.

8.2 Results and Discussion

The results are visualized in Fig. 4. We observe two key trends:

1. Superior Stability in Low-Noise Regimes: Contrary to the intuition that optimization induces fragility, CLAP-S (red line) demonstrates *higher* similarity retention than RSU (blue line) for perturbation levels $p < 0.2$. This is particularly evident in BA networks. This suggests that CLAP-S acts as a "structural anchor," converging on a deterministic, compact alignment of drivers that is less prone to random drift than the stochastic selection of RSU.

2. Resilience in Scale-Free Topologies: In BA networks (Right Panel), CLAP-S maintains a significant stability advantage up to $p \approx 0.35$. Since real-world networks often exhibit scale-free properties, this result implies that our method

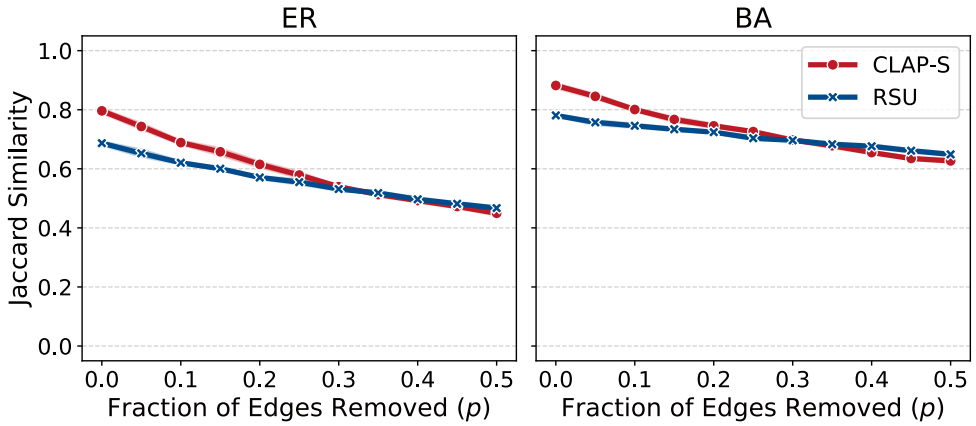

Fig. 4. **Robustness of Driver Sets under Edge Removal.** The plots show the Jaccard similarity between the initial driver set and the set re-calculated after removing a fraction p of edges. **Left:** ER Network. **Right:** BA Network. In the regime of realistic noise ($p < 0.2$), CLAP-S (red) exhibits consistently higher stability than the random sampling baseline (blue). This indicates that CLAP-S locks onto structurally robust configurations, whereas random sampling drifts among equivalent but distinct solutions.

is highly practical for reliable actuator deployment in real systems subject to link uncertainties.

In summary, the minimal union sets found by CLAP-S are not only cost-efficient but also topologically robust, maintaining their validity better than random baselines in the face of structural noise.

9 Mechanism and Scalability of the CLAP-S Algorithm

In this section, we investigate the underlying mechanism and computational efficiency that enable CLAP-S to handle large-scale systems.

9.1 Locality of Multi-layer Control Conflicts.

To understand why the algorithm converges so rapidly despite the combinatorial nature of the search space, we examine the empirical distribution of the CLAP length h (the number of segments in a shortest path). As shown in Fig. 5(a) and (b), the distribution of h for both synthetic and real-world networks is highly concentrated, with the vast majority of paths having a length of $h = 1$ or $h = 2$. The empirical mean remains consistently low ($\bar{h} \approx 1.2$).

This observation reveals a fundamental property of multi-layer structural controllability: the locality of control conflicts. It suggests that redundant driver nodes are not typically separated by long, complex chains of interactions across layers. Instead, most cross-layer redundancies can be resolved through very short alternating sequences of driver exchanges. This locality ensures that the search depth for CLAP-S remains extremely shallow, effectively bypassing the theoretical worst-case complexity $O(|V|^2|E|)$ and explaining the millisecond-level response times observed in practice.

9.2 Computational Scaling and Resource Efficiency.

We further evaluate the scalability of CLAP-S by varying the network size N from 100 to $10,000$ at a fixed density ($\langle k \rangle = 4$). As visualized in Fig. 5(c), the Average Runtime exhibits efficient polynomial scaling. Even for a duplex system with $10,000$ nodes—a scale that often exceeds the practical capacity of standard ILP solvers—CLAP-S completes the optimization in approximately 4.1 seconds.

Most notably, the Algorithmic Peak Memory (Fig. 5(d)) remains exceptionally low, reaching only about 6 MiB at $N = 10,000$. The slight fluctuations in memory usage reflect the dynamic nature of the BFS frontier, which depends more on the local topology and the number of active search branches than on the global node count. This minimal memory footprint is a direct result of our "search-on-demand" strategy: unlike sampling-based methods that must store multiple MDS configurations, CLAP-S only maintains the current matching and a shallow search frontier. This resource efficiency makes CLAP-S uniquely suitable for deployment in resource-constrained environments or as a real-time monitor for massive cyber-physical systems.

10 Driver Occupancy and Structural Rigidity

To further investigate the discrepancy between the optimization landscapes of synthetic and real-world networks, we analyzed the relationship between the **Initial Driver Occupancy Ratio** ($|\mathrm{DD}_1 \cup \mathrm{DD}_2|/N$) and the absolute optimization gain (ΔN_D). The results, visualized in Fig. 6, reveal a fundamental topological divergence.

Synthetic vs. Real-World Divergence.

- **Synthetic Regime (Low Occupancy, High Elasticity):** As shown in Fig. 6(a) and (b), synthetic networks (both ER and BA) typically exhibit a driver occupancy ratio below 50%. In this regime, the network possesses abundant

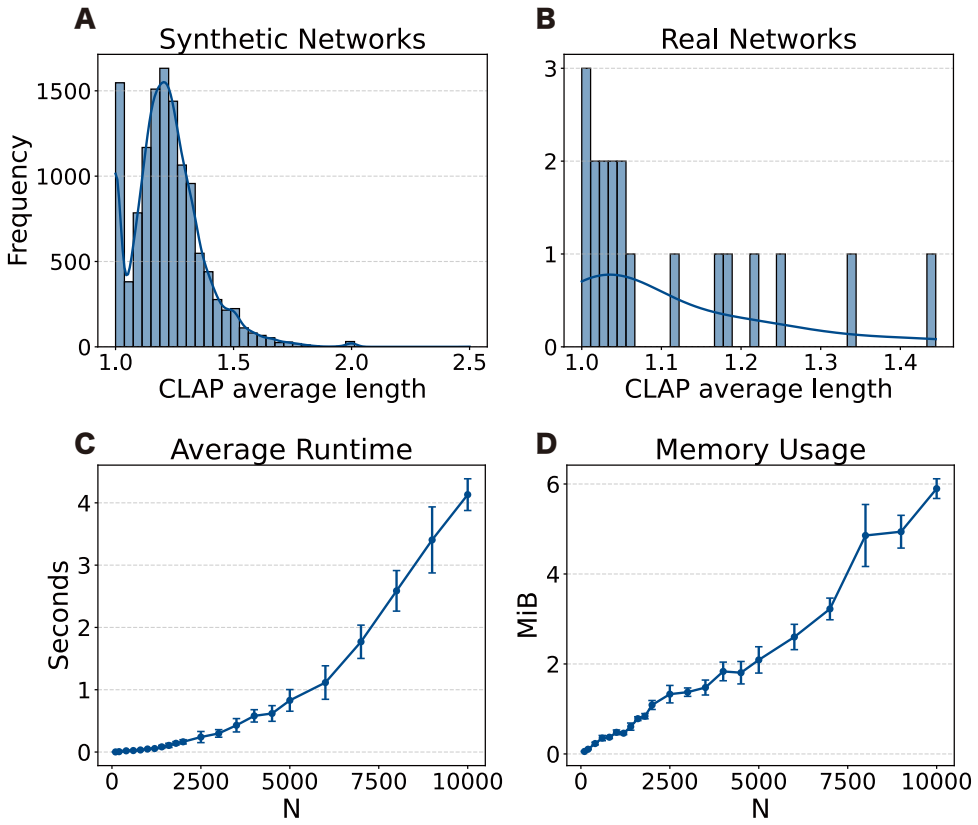

Fig. 5. **Mechanistic and scalability analysis.** (a)-(b) Frequency distribution of the CLAP length h for synthetic and real-world networks, showing strong locality ($\bar{h} \approx 1.2$). (c)-(d) Scaling of average runtime and peak memory usage as a function of network size N (from 100 to $10,000$). Error bars represent standard deviation across 10 independent realizations.

"non-driver" nodes, providing a flexible buffer for reconfiguring matchings. Consequently, the optimization gain scales linearly with the occupancy, maintaining a high conversion efficiency close to the theoretical limit.

- **Real-World Regime (High Occupancy, High Rigidity):** In sharp contrast, Fig. 6(c) demonstrates that most empirical networks cluster in the extremely high occupancy region ($> 80\%$, sometimes approaching 100%). This indicates that in systems like genetic regulation or sparse social hierarchies, nearly every node is structurally compelled to participate in the driver set of at least one layer. This leaves minimal "topological slack" for optimization. The low correlation ($R^2 = 0.12$) and the flattened slope for high-occupancy networks confirm that structural rigidity severely compresses the optimization headroom.

The Sweet Spot for CLAP-S. Combining these observations, we identify the topological conditions under which CLAP-S yields the most significant improvements relative to the system size. The algorithm performs closest to the theoretical conversion ceiling ($k = 0.5$) in networks with moderate driver occupancy (20%–60%) and high matching degeneracy.

- Networks with extremely low occupancy ($\approx 0\%$) have no conflicts to resolve.
- Networks with extremely high occupancy ($> 90\%$, typical of real-world scale-free tails) have no space to maneuver.
- Networks in the intermediate range (typical of dense random graphs or engineered systems) offer the ideal balance of conflict potential and reconfiguration flexibility.

Nevertheless, even for the rigid real-world networks on the far right of the spectrum, CLAP-S remains valuable by extracting the few remaining non-trivial optimizations that stochastic methods cannot find, as evidenced by the consistent power-law recovery rate discussed in the main text.

References

[1] L. Lovász and M. D. Plummer, *Matching Theory*, ser. Annals of Discrete Mathematics. North-Holland, 1986, vol. 29.
[2] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, ser. Algorithms and Combinatorics. Springer, 2003, vol. 24.
[3] C. Berge, "TWO THEOREMS IN GRAPH THEORY," *Proceedings of the National Academy of Sciences*, vol. 43, no. 9, pp. 842–844, Sep. 1957, publisher: Proceedings of the National Academy of Sciences. [Online]. Available: http://dx.doi.org/10.1073/pnas.43.9.842
[4] M. De Domenico, V. Nicosia, A. Arenas, and V. Latora, "Structural reducibility of multilayer networks," *Nature Communications*, vol. 6, no. 1, Apr. 2015, publisher: Springer Science and Business Media LLC. [Online]. Available: http://dx.doi.org/10.1038/ncomms7864
[5] C. Stark, "BioGRID: a general repository for interaction datasets," *Nucleic Acids Research*, vol. 34, no. 90001, pp. D535–D539, Jan. 2006, publisher: Oxford University Press (OUP). [Online]. Available: http://dx.doi.org/10.1093/nar/gkj109

[6] M. Costanzo, A. Baryshnikova, J. Bellay, Y. Kim, E. D. Spear, C. S. Sevier, H. Ding, J. L. Koh, K. Toufighi, S. Mostafavi, and others, "The genetic landscape of a cell," *science*, vol. 327, no. 5964, pp. 425–431, 2010, publisher: American Association for the Advancement of Science.

[7] B. L. Chen, D. H. Hall, and D. B. Chklovskii, "Wiring optimization can relate neuronal structure and function," *Proceedings of the National Academy of Sciences*, vol. 103, no. 12, pp. 4723–4728, Mar. 2006, publisher: Proceedings of the National Academy of Sciences. [Online]. Available: http://dx.doi.org/10.1073/pnas.0506806103

[8] E. Omodei, M. De Domenico, and A. Arenas, "Characterizing interactions in online social networks during exceptional events," *Frontiers in Physics*, vol. 3, Aug. 2015, publisher: Frontiers Media SA. [Online]. Available: http://dx.doi.org/10.3389/fphy.2015.00059

[9] M. De Domenico and E. G. Altmann, "Unraveling the Origin of Social Bursts in Collective Attention," *Scientific Reports*, vol. 10, no. 1, Mar. 2020, publisher: Springer Science and Business Media LLC. [Online]. Available: http://dx.doi.org/10.1038/s41598-020-61523-z

[10] D. Krackhardt, "Cognitive social structures," *Social Networks*, vol. 9, no. 2, pp. 109–134, Jun. 1987, publisher: Elsevier BV. [Online]. Available: http://dx.doi.org/10.1016/0378-8733(87)90009-8

[11] W. Raub, "Emmanuel Lazega: The Collegial Phenomenon. The Social Mechanisms of Cooperation among Peers in a Corporate Law Partnership. Oxford: Oxford University Press, 2001. xi + 346 pp," *European Sociological Review*, vol. 21, no. 2, pp. 183–184, Apr. 2005, publisher: Oxford University Press (OUP). [Online]. Available: http://dx.doi.org/10.1093/esr/jci012

[12] J. Coleman, E. Katz, and H. Menzel, "The Diffusion of an Innovation Among Physicians," *Sociometry*, vol. 20, no. 4, p. 253, Dec. 1957, publisher: JSTOR. [Online]. Available: http://dx.doi.org/10.2307/2785979

[13] F. Pasqualetti, S. Zampieri, and F. Bullo, "Controllability metrics, limitations and algorithms for complex networks," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 40–52, 2014.

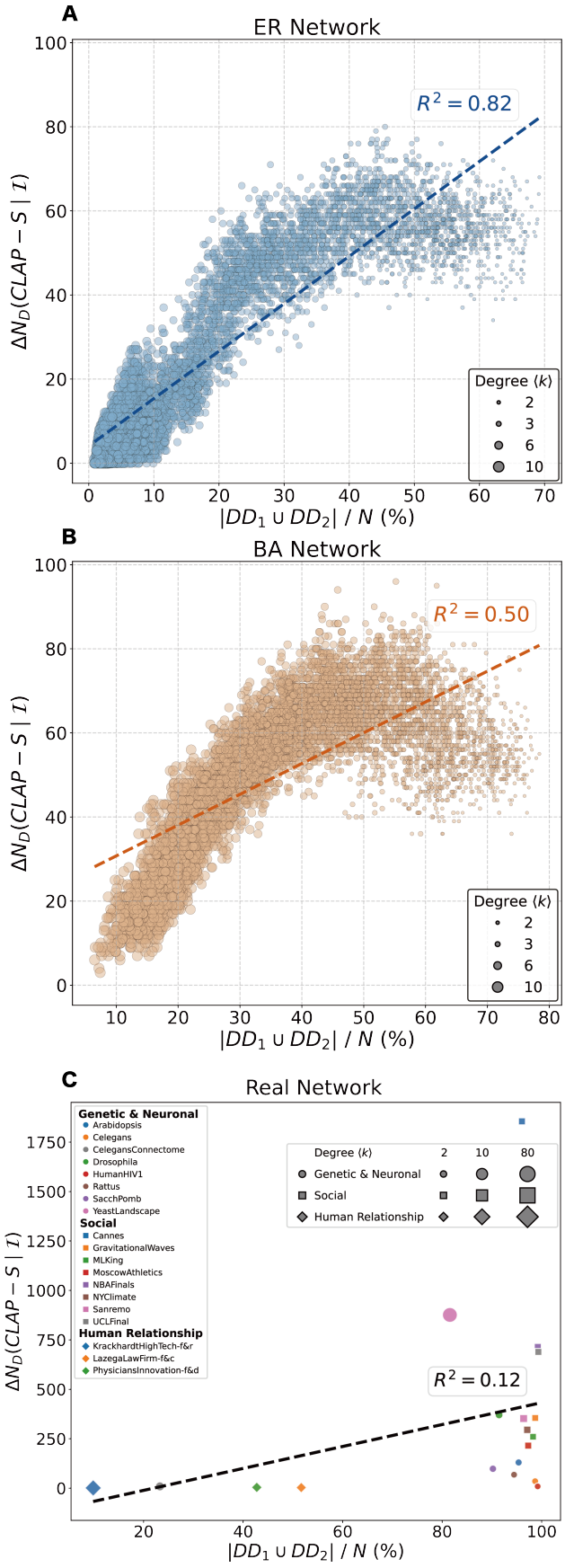

Fig. 6. **Impact of Initial Driver Occupancy on Optimization Potential.** (a-b) Synthetic networks (ER and BA) mostly reside in a low-occupancy regime ($< 60\%$), where optimization gain (ΔN_D) increases linearly with the size of the driver union, tracking the theoretical efficiency frontier. (c) Real-world networks typically exhibit extreme occupancy ($> 80\%$), indicating high structural rigidity. The optimization gain is compressed because the vast majority of nodes are topologically "locked" into driver roles, leaving little room for cross-layer reconfiguration.