TEMPORAL HYPERGRAPH COARSENING WITH LOCAL REPAIR AND STATIC-RECOMPUTATION GUARANTEES

Anonymous authors
Paper under double-blind review

ABSTRACT

Large hypergraphs are increasingly used to model higher-order machine learning data, including group recommendation sessions, temporal hyperlink events, biological co-activation sets, and evolving tag or topic groups. Existing hypergraph coarsening methods reduce hypergraph size, but they are usually applied to a fixed snapshot; when events arrive over time, the standard alternative is to recompute the coarsening from scratch. We study temporal hypergraph coarsening as an incremental maintenance problem. We propose THGC, a framework that maintains a coarsened hypergraph under timestamped event streams through a Temporal Update Engine and a Coarsening Strategy Interface. The repair step is driven by TAPS, a Temporally-Aware Proximity Score that combines recency-weighted Personalized PageRank with a local conductance signal so that repair decisions account for both temporal interaction history and cut structure. The maintained object remains a hypergraph, with coarse hyperedges and incidence structure; graph projections are used only as local scoring surrogates. We give stability bounds relating temporal score drift, coarsening error, and the deviation from full static recomputation. Experimental results on temporal hypergraph benchmarks demonstrate significant speedups in update time while preserving the quality of the coarsened representation.

1 INTRODUCTION

Hypergraphs represent relations among sets of entities rather than only pairs (Berge, 1973). This makes them a natural representation for machine learning problems where observations are intrinsically group-valued: a recommendation session links a user, multiple items, and context; a temporal hyperlink prediction event links all participants in a higher-order interaction; a biological measurement may record a set of co-activated genes or brain regions; and a question-answering platform records evolving tag sets over time (Benson et al., 2018; Antelmi et al., 2023). In these settings the hypergraph is not a fixed artifact. New group events arrive, old interactions expire under a windowed view, and downstream pipelines need a compact representation that can be updated without rebuilding everything after each timestamp.

Hypergraph coarsening addresses the size problem by merging fine vertices into coarse supernodes while preserving structural information needed by later algorithms (Aghdaei & Feng, 2022; Aghdaei et al., 2021; Sajadinia et al., 2026). Static coarsening is useful, but it leaves a practical gap for event streams. Full static recomputation after every batch is simple and accurate, but it repeats nearly all work even when only a local region changed. Periodic recomputation reduces cost, but it introduces a choice of resynchronization interval and does not itself explain which local changes should alter the coarse mapping. Dynamic graph maintenance is also not a direct substitute, because the maintained state here is a hypergraph coarsening: the mapping from fine vertices to coarse nodes, the induced coarse hyperedges, and the incidence structure all have to remain meaningful as higher-order events evolve.

We formulate temporal hypergraph coarsening as an incremental maintenance problem. Given a timestamped stream of node and hyperedge updates, the goal is to maintain a coarsened hypergraph that tracks the result of full static recomputation while spending work only near the affected region whenever locality makes this justified. This is the central thesis of the paper: **THGC replaces repeated global reconstruction with localized repair and controlled resynchronization while**

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

explicitly tracking a moving static-recomputation reference. TAPS is the temporally aware repair policy instantiated within this framework. THGC exposes quality-efficiency control knobs, including a coupling threshold, a per-round merge cap, and a recomputation interval, and it performs constrained local repair under these controls.

The main design question is how to decide which coarse nodes should be reconsidered after an event. We propose **TAPS** (Temporally-Aware Proximity Score), a coupling score that blends a recency-weighted **Personalized PageRank** (PPR) proximity term (Brin & Page, 1998; Andersen et al., 2006; Zhang et al., 2016) with a local conductance term (Zhou et al., 2007; Shi & Malik, 2000). The contribution is not that a linear combination of two familiar signals is surprising. Rather, the point is operational: in a temporal maintenance pipeline, the diffusion term captures repeated temporal interaction, while the conductance term discourages repairs that damage local cut structure. Our experiments therefore evaluate not only conductance, but also deviation from full static recomputation, assignment churn, downstream metrics, memory, and failure regimes under bursty updates.

THGC maintains a genuine coarsened hypergraph. We use a clique-expansion adjacency only to score local candidate pairs for TAPS, following standard hypergraph learning practice (Zhou et al., 2007). The state that THGC updates is still the hypergraph incidence representation: fine vertices, fine hyperedges, coarse nodes, coarse hyperedges, mapping entries, and incidence entries. This distinction is important because projection-based scores can lose some simultaneous higher-order information; THGC mitigates this by preserving and evaluating the induced coarse hypergraph rather than replacing the task with graph coarsening.

Our contributions are:

- **Incremental formulation.** We formulate temporal hypergraph coarsening as maintaining the coarse hypergraph itself under timestamped structural updates, rather than repeatedly coarsening independent snapshots.
- **Local maintenance architecture.** THGC combines dirty-region localization, a pluggable **Coarsening Strategy Interface** (CSI), and controlled global resynchronization, allowing static coarseners to initialize or define references while temporal updates are processed locally.
- **Temporally aware repair.** We instantiate THGC with TAPS, which couples recency-sensitive diffusion proximity with current cut structure to rank local repair candidates.
- **Stability characterization.** We bound score drift induced by temporal updates and coarse-space approximation and show that merge eligibility is preserved whenever candidate scores remain outside the induced threshold-margin band.
- **Empirical regime map.** Across natural, long-horizon, burst, sliding-window, memory, and downstream evaluations, we characterize when local repair tracks full recomputation efficiently and when global resynchronization becomes preferable.

The empirical picture is intentionally scoped. Local repair is most useful when streams induce real but localized churn: static recomputations change materially across time, yet the affected region is small enough that repairing it is cheaper than rebuilding the whole coarsening. Under dense bursts or high turnover, the dirty region expands and recomputation can become the right choice. We report both regimes.

2 Related Work

Hypergraph coarsening compresses a hypergraph by merging vertices into supernodes while preserving structural information for downstream computation (Aghdaei & Feng, 2022; Aghdaei et al., 2021; Sajadinia et al., 2026). It is related to hypergraph partitioning, but partitioners such as hMETIS and KaHyPar use multilevel coarsening mainly as an intermediate step toward balanced cuts (Karypis & Kumar, 1998; Akhremtsev et al., 2017; Çatalyürek et al., 2023). THGC instead treats the coarsened hypergraph itself as the maintained object.

Temporal graph neural systems such as TGN maintain representations from timestamped pairwise events (Rossi et al., 2020; Zhou et al., 2022; Wang et al., 2023), while higher-order link prediction

studies interactions involving more than two nodes (Benson et al., 2018). THGC asks a complementary systems question: how should the combinatorial compression evolve? Dynamic graph coarsening and sparsification provide useful perturbation tools (Loukas, 2019; Kumar et al., 2023), but contracting graph edges only updates pairwise adjacency. In THGC, local repair must update the fine-to-coarse map, induced coarse hyperedges, and incidence structure. Thus graph projections can rank local candidates, but they do not define the maintained state.

Relative to static and periodic recomputation, THGC is a hybrid maintenance system. Static recomputation is the quality reference but repeats global work; periodic recomputation is a practical engineering baseline; local graph repair does not preserve hyperedges. THGC uses localized repair when temporal locality holds and controlled resynchronization when it does not.

3 Preliminaries and Problem Setup

We use *temporal* to mean hypergraphs with timestamps on hyperedges or snapshots (distinct from the broader *dynamic* notion, which need not carry timestamps). Updates arrive in chronological order, and our coupling metric incorporates recency via exponential decay.

Temporal hypergraph. We study a sequence $\{\mathcal{H}_t\}_{t=1}^{T}$ of weighted hypergraphs $\mathcal{H}_t = (V_t, \mathcal{E}_t, w_t)$, where timestamped events can insert or delete nodes or hyperedges, change weights, or expire old hyperedges under a sliding window. For $S \subseteq V_t$, let $\mathrm{vol}_t(S)$ denote weighted degree volume and $\mathrm{cut}_t(S)$ the total weight of hyperedges crossing S and $V_t \setminus S$.

Coarsening Map and Induced Partition. At each time t, a coarsening map (or partition map) $\varphi_t : V_t \to \mathcal{C}_t$ assigns each node to a coarse "supernode" (cluster) in the coarse node set $\mathcal{C}_t = \{C_1^t, \ldots, C_{k_t}^t\}$, with $C_i^t := \{v \in V_t : \varphi_t(v) = i\}$ and $\bigsqcup_{i=1}^{k_t} C_i^t = V_t$. We control the coarsening level via a target ratio $r \in (0, 1]$, typically enforcing $k_t = |\mathcal{C}_t| \approx r\,|V_t|$ for all t (or a dataset-specific schedule r_t).

Normalized conductance for hypergraphs. Given the k_t-way partition $\mathcal{P}_t = \{C_1^t, \ldots, C_{k_t}^t\}$ of V_t, we quantify the inter-cluster boundary via normalized conductance (Zhou et al., 2007) computed on the original hypergraph:

$$\mathrm{cond}_t(\mathcal{P}_t) \;=\; \frac{1}{k_t} \sum_{i=1}^{k_t} \frac{\mathrm{cut}_t(C_i^t)}{\min\{\mathrm{vol}_t(C_i^t)\,,\; \mathrm{vol}_t(V_t \setminus C_i^t)\}}\,. \tag{1}$$

Conductance is denoted $\phi_t(\mathcal{P}_t)$. Lower values indicate tighter intra-cluster connectivity with fewer cross-cluster hyperedges; it is one structural metric, not the sole definition of coarsening quality.

Maintenance objective. Given $\{\mathcal{H}_t\}_{t=1}^{T}$ and a target ratio r, our goal is to produce $\{\varphi_t\}_{t=1}^{T}$ such that the induced coarsened hypergraph remains close to a full static recomputation while the transition from φ_{t-1} to φ_t is cheaper than rebuilding from scratch. We therefore report multiple quantities: per-slice hypergraph conductance, update time, pair-disagreement from the static recomputation reference, static-vs-static assignment churn, downstream task quality, and state size. Conductance is one structural metric, not the sole definition of coarsening quality.

Static recomputation reference. Let $B_t(\mathcal{H}_t)$ denote the selected static coarsening procedure applied from scratch to snapshot $\mathcal{H}_t$ under the same base method and target-ratio configuration used by the incremental run. THGC maintains an incremental mapping $\widetilde{B}_t$ from $\widetilde{B}_{t-1}$ after processing the stream update. Static-reference disagreement metrics compare $\widetilde{B}_t$ to $B_t(\mathcal{H}_t)$; when different base algorithms are used, the corresponding B_t is always named in the experiment.

Pair-disagreement metric. For two mappings P, Q on the same node set V with $n = |V|$, we use

$$d_{\mathrm{pair}}(P, Q) = \frac{1}{\binom{n}{2}} \sum_{u<v} \mathbf{1}[\mathbf{1}\{P(u) = P(v)\} \neq \mathbf{1}\{Q(u) = Q(v)\}]\,. \tag{2}$$

This metric is invariant to arbitrary cluster labels because it compares pairwise co-membership rather than cluster IDs. Assignment churn and pair disagreement measure different effects: churn counts node-level assignment changes after best matching, while pair disagreement counts the fraction of node pairs whose co-membership relation changes. A substantial number of boundary-node reassignments can therefore coexist with small pair disagreement when most pairwise cluster relations are preserved.

Event streams and snapshots. A continuous-time stream with timestamps can be evaluated either after every event, after minibatches of events, or under a sliding window (Rossi et al., 2020; Zhou et al., 2022). The experiments specify which construction is used. We use "update" for a processed event or minibatch, "snapshot" for the current hypergraph after those updates, and "slice" for the reporting point at which metrics are recorded.

4 TEMPORALLY AWARE PROXIMITY SCORE (TAPS)

Static coupling metrics such as effective resistance score a fixed snapshot (Aghdaei & Feng, 2022). TAPS instead scores local repair candidates using both recent temporal interaction and the current cut structure. The score is used only to rank candidate repairs; it does not replace the maintained hypergraph by a graph.

4.1 TEMPORALLY-WEIGHTED HYPERGRAPH

Accumulated temporal weight, $W_t(e)$ for a hyperedge e, that is involved in the interaction is defined as:

$$W_t(e) = w_t(e) + e^{-\lambda(t-t_{prev})} \cdot W_{t_{prev}}(e) \tag{3}$$

where t_{prev} is the timestamp of the last update for e. Here $w_t(e)$ denotes the event contribution or weight increment observed at time t, not the entire accumulated history of e. If e is new, $W_t(e) = w_t(e)$. Using $W_t(e)$, TAPS forms a clique-expansion scoring surrogate (Zhou et al., 2007):

$$A_{ij}(t) = \sum_{e \in \mathcal{E}_t | v_i, v_j \in e} \frac{W_t(e)}{|e| - 1} \tag{4}$$

with degree matrix $D(t)$ and row-stochastic transition matrix $P(t) = D(t)^{-1}A(t)$. The maintained state remains the hypergraph incidence structure; the projection is used only for local proximity scores.

4.2 DEFINING THE TAPS METRIC

TAPS combines two components. The **random walk component** is inspired by **Personalized PageRank** (PPR) (Zhang et al., 2016) and provides a temporally-aware view of node proximity by simulating a random walk over the time-weighted projection. The PPR vector $\pi_s^{(t)}$ for source node s is the stationary distribution of a random walk that teleports back to s with probability $1 - \gamma$:

$$\pi_s^{(t)} = \gamma P(t)^T \pi_s^{(t)} + (1 - \gamma) h_s \tag{5}$$

where h_s is a personalization vector. The component is defined symmetrically as $S_{ppr}(u, v, t) = \pi_u^{(t)}[v] + \pi_v^{(t)}[u]$.

The **conductance component** ensures that supernodes form dense communities and focuses on the local quality of clusters (Zhou et al., 2007). The candidate cluster $C \subseteq \mathcal{V}$ is the current supernode (or proposed merged supernode) containing both u and v; during an incremental update, C is taken to be the 1-hop coarse neighbourhood of the dirty region including u and v. The conductance of C is computed on the current snapshot $\mathcal{H}_t$ using Eq. equation 1:

$$S_{cut}(u, v, t) = 1 - \phi_t(C) \tag{6}$$

The TAPS score for a pair of nodes (u, v) at time t is a weighted coupling score with mixing parameter $\alpha \in [0, 1]$:

$$TAPS(u, v, t) = \alpha \cdot S_{ppr}(u, v, t) + (1 - \alpha) \cdot S_{cut}(u, v, t) \tag{7}$$

When $\alpha = 1$, TAPS reduces to the PPR term; when $\alpha = 0$, it reduces to the conductance term. The coefficient α interpolates contributions in their native score scales; it is not an exact equal-contribution coefficient unless the terms are additionally normalized. The two terms encode complementary inductive biases: temporally repeated interaction through diffusion proximity and preservation of local cut structure. In practice, an α sweep can appear flat on conductance when the residual threshold or score distribution leaves the candidate ranking unchanged. We therefore evaluate the coupling

Algorithm 1 THGC+TAPS

Require: H_0, updates $\{\Delta_i\}$, threshold τ, interval k, merge cap K
Ensure: H_{coarse}, mapping
(H_{coarse}, mapping) ← `CSI.initialize_from_baseline`(H_0); last_recompute ← 0
for each update Δ_i **do**
  Apply Δ_i to H_0; dirty ← `CSI.perform_incremental_update`(Δ_i)
  if `time`(Δ_i) − last_recompute $\geq k$ **then**
    `CSI.recompute_stability`(dirty); `CSI.split/merge_unstable`(); last_recompute ← `time`(Δ_i)
  end if
  $\mathcal{Q}$ ← 1-hop pairs in dirty; score by TAPS $s(u, v)$
  Greedily contract up to K disjoint pairs with $s(u, v) \geq \tau$; update mapping and incidence entries
  Log runtime, conductance, cluster sizes
end for
return H_{coarse}, mapping

by conductance, deviation from static recomputation, and downstream metrics rather than using conductance alone.

TAPS can also drive stand-alone static coarsening; the next section embeds it in the THGC framework.

5 Temporal Hypergraph Coarsening (THGC) Framework

THGC separates update management from coarsening logic. A temporal update engine maintains the current timestamped hypergraph snapshot and passes each change to a Coarsening Strategy Interface (CSI), which can initialize from existing static coarseners or from TAPS-based greedy coarsening (Aghdaei & Feng, 2022; Aghdaei et al., 2021; Sajadinia et al., 2026). The architecture diagram is given in Appendix B; the main operations are summarized in Algorithm 1.

For a changed hyperedge e, THGC marks the coarse nodes containing vertices of e and their one-hop coarse neighbours:

$$\mathcal{D} = \{\varphi(v) : v \in e\} \cup \bigcup_{w \in e} \mathcal{N}_{\text{coarse}}(\varphi(w)).$$

Changed hyperedges invalidate scores for incident clusters; the one-hop expansion covers candidate merges whose local cut or projected diffusion neighbourhood changes through an adjacent coarse boundary. Given a symmetric coupling score $s(\cdot, \cdot)$, threshold τ, merge cap K, and resynchronization interval k, THGC greedily contracts up to K disjoint local candidate pairs with $s(u, v) \geq \tau$, updates mapping and incidence entries, and periodically resynchronizes when locality no longer justifies purely local repair.

Let ΔE be changed hyperedges, Q the dirty candidate-pair set, W the recent window, and $C^\star$ the maximum recent co-occurrence count over evaluated pairs. TAPS-based local repair costs

$$O\left(\sum_{e \in \Delta E} |e|^2\right) + O\big(|Q|(C^\star \log C^\star + W\bar{b})\big) + O(K \log |Q|),$$

where $\bar{b}$ is average changed-hyperedge size. Thus work scales with the dirty region rather than the full hypergraph when updates are local; the detailed derivation is in Appendix I.2.

6 Theoretical Guarantees

Our guarantees rest on five standard assumptions (stated formally in Appendix C): (A1) the transition matrix $P(t)$ is row-stochastic with bounded per-step changes under the temporal accumulator; (A2) PPR uses teleportation $\gamma \in (0, 1)$ in resolvent form; (A3) TAPS is symmetric and bounded in $[0, 2]$; (A4) the lifted coarse Laplacian satisfies the restricted spectral approximation on the cluster-constant subspace $\mathcal{U}_t$ with error ε_t; and (A5) the random-walk operator deviation on $\mathcal{U}_t$ is bounded

by $c_{\mathrm{op}}\,\varepsilon_t$ (Chung, 1997; Loukas, 2019; Stewart & guang Sun, 1990). A1–A2 formalize temporal locality: one update does not globally perturb transition mass. A3 is intrinsic to the score definition. A4–A5 formalize structural fidelity of the maintained coarse space. Together, these assumptions describe the operating regime targeted by THGC: updates are localized and the current partition remains a meaningful approximation of the fine structure. Failure of temporal locality is exactly the regime probed by burst experiments, while degradation of coarse-space fidelity motivates controlled resynchronization.

6.1 Mathematical Guarantees

Theorem 1 (Temporal stability of TAPS)**.** *Fix (u, v) and let C be any admissible cluster at times t and $t{+}1$ such that $u, v \in C$. Then*

$$\begin{aligned}\big|\,\mathrm{TAPS}(u,v,t{+}1) - \mathrm{TAPS}(u,v,t)\,\big| \;\le\\ \alpha\cdot\frac{2\gamma}{1-\gamma}\,\|P(t{+}1)-P(t)\|_1 \;+\\ (1-\alpha)\cdot\big|\phi_{t+1}(C)-\phi_t(C)\big|,\end{aligned}$$

and $\|P(t{+}1) - P(t)\|_1$ is bounded in terms of the decayed temporal weight increments and degree changes specified by the update rule in the paper.

Interpretation. Sparse updates cause only proportionally small changes to $P(t)$ and $\phi_t(C)$, so coupling scores outside the dirty region remain valid without recomputation. Dense updates produce larger deviations, which correctly triggers periodic global recomputation at interval k.

Theorem 2 (Combined preservation under coarsening and time)**.** *Let $\mathrm{TAPS}^{\diamond}(u, v, t)$ denote the TAPS value computed on the coarse graph at time t and lifted back to the fine graph via the coarsening matrix Π_t. Assume the PPR personalization vectors used in S_{ppr} are cluster-constant at time t (i.e., $h_u, h_v \in \mathcal{U}_t = \mathrm{im}(\Pi_t^{\top})$), or equivalently are projected to $\mathcal{U}_t$ via Π_t. Under Assumptions 1–5, for any (u, v) contained in a union of clusters at time t,*

$$\begin{aligned}\big|\,\mathrm{TAPS}^{\diamond}(u,v,t) - \mathrm{TAPS}(u,v,t)\,\big| \;\le\\ \alpha\cdot\frac{2\gamma}{1-\gamma}\,c_{\mathrm{op}}\,\varepsilon_t \;+\; (1-\alpha)\cdot\varepsilon_t\end{aligned}$$

Combining this with Theorem 1 by the triangle inequality yields the end to end slice-to-slice bound

$$\begin{aligned}\big|\,\mathrm{TAPS}^{\diamond}(u,v,t{+}1) - \mathrm{TAPS}(u,v,t)\,\big| \;\le\\ \alpha\cdot\frac{2\gamma}{1-\gamma}\,c_{\mathrm{op}}\,\varepsilon_{t+1} \;+\; (1-\alpha)\cdot\varepsilon_{t+1}\\ +\alpha\cdot\frac{2\gamma}{1-\gamma}\,\|P(t{+}1)-P(t)\|_1 \;+\\ (1-\alpha)\cdot\big|\phi_{t+1}(C)-\phi_t(C)\big|\end{aligned}$$

These inequalities show that TAPS deviations due to coarsening and temporal updates are jointly controlled by the restricted spectral error ε_t and the transition/volume increments arising from the temporal accumulator (Loukas, 2019; Zhang et al., 2016).

Interpretation. The guarantees decompose the approximation incurred by temporal maintenance into two quantities: temporal transition drift and restricted coarse-space distortion. Under these conditions, TAPS scores remain stable. The restricted spectral approximation is a condition on the maintained coarse space rather than a guarantee produced by every possible CSI implementation; it identifies the regime in which local repair can safely reuse previous coarse structure.

Corollary 1 (Threshold-margin stability of merge eligibility)**.** *Let $q = (u, v)$ be a candidate pair considered by Algorithm 1. Let $s_t(q)$ denote the static-reference TAPS score and $\widetilde{s}_t(q)$ the maintained score used by THGC. Suppose the preceding bounds imply $|\widetilde{s}_t(q) - s_t(q)| \le \eta_t$ for all q in the local candidate set. For threshold τ, define the ambiguity band*

$$\mathcal{A}_t(\eta_t) = \{q : |s_t(q) - \tau| \le \eta_t\}.$$

Then for every $q \notin \mathcal{A}_t(\eta_t)$,

$$\mathbf{1}\{\widetilde{s}_t(q) \geq \tau\} = \mathbf{1}\{s_t(q) \geq \tau\}.$$

Equivalently, if $\widetilde{\mathcal{M}}_t$ and $\mathcal{M}_t^\star$ are the maintained and static-reference threshold-eligible candidate sets, then

$$\widetilde{\mathcal{M}}_t \triangle \mathcal{M}_t^\star \subseteq \mathcal{A}_t(\eta_t).$$

If two candidate scores differ by more than $2\eta_t$, their pairwise ordering is also preserved. Thus the theory directly links score stability to Algorithm 1: only candidates near the decision boundary can change merge eligibility under incremental maintenance.

An additional fixed-slice spectral comparability result for two valid coarse representations is stated and proved in Appendix H.6. The main text focuses on the temporal score stability that governs local repair decisions.

7 Experimental Setup

We evaluate THGC on temporal hypergraph datasets from the open-source collection of Benson et al. (Benson et al., 2018): NDC-classes, contact-high-school, tags-ask-ubuntu, and tags-math-sx. Dataset counts and domains are reported in Appendix Table 2. Additional long-horizon and repeated-seed analyses focus on the two datasets with the cleanest update-stream and downstream-task instrumentation, NDC-classes and tags-math-sx.

For each stream, we initialise the coarsening on an initial cumulative snapshot and process later timestamped events as updates. We compare incremental THGC to full static recomputation and to periodic recomputation schedules. Unless otherwise stated, quality is measured on matched reporting horizons and, where possible, matched compression. The main structural metrics are hypergraph conductance, assignment churn under consecutive static recomputations, and pair-disagreement between the incremental mapping and the static recomputation reference. We also report downstream node classification or hyperlink prediction where labels or held-out hyperedges are available, and we track process-level resident set size (RSS) together with exact state counters such as mapping and incidence entries.

We use HyperEF, HyperSF, SHyPar, heavy-edge repair, PPR-only coupling, conductance-only coupling, and periodic recomputation as comparison points, depending on the experiment (Aghdaei & Feng, 2022; Aghdaei et al., 2021; Sajadinia et al., 2026; Andersen et al., 2006; Zhang et al., 2016; Zhou et al., 2007). THGC uses a local dirty region induced by changed hyperedges and neighbouring coarse nodes. The threshold τ controls which candidate repairs are accepted, the merge cap K bounds local work per round, and the recomputation interval k controls resynchronization. All timings were produced on an Apple M3 laptop. Downstream neural runs use Apple Metal Performance Shaders (MPS) where available. When a method name carries the suffix CSI (e.g., HyperEFCSI, HyperSFCSI, SHyParCSI), it denotes the lightweight CSI-compatible structural variant defined in Appendix J; it is not an incremental execution of the full static algorithm.

8 Results

8.1 Q1: Does the static target actually change?

The central experimental question is not whether THGC produces a low-conductance partition in isolation. It is whether THGC can follow a static coarsener whose own solution changes as the stream evolves. Static-vs-static churn provides the reference scale for this question.

Conductance remains useful, but it is not sufficient by itself. The evaluation therefore asks whether the stream actually changes the static coarsening and whether THGC follows that changing reference. On NDC-classes, consecutive full static recomputations every 200 updates change 27.5% of assignments on average under best matching, with mean pair-disagreement 0.0028. On tags-math-sx, the corresponding assignment churn is 37.6%, with mean pair-disagreement 0.0072. Thus flat conductance columns do not imply that the stream is stationary; many structurally different coarsenings can have nearly identical conductance.

The long-horizon trajectory confirms that THGC tracks the static reference at the same scale as static-vs-static variability. After 750 updates, THGC's final pair-disagreement from static recomputation is 0.00335 on NDC-classes and 0.00795 on tags-math-sx. The corresponding static-vs-static pair-disagreements over the same horizon are 0.00340 and 0.00832. Static-vs-static disagreement provides an empirical scale for how much the reference itself moves over the same horizon; THGC-vs-static disagreement being of comparable magnitude indicates that maintenance error does not overwhelm the intrinsic motion of the reference partition. The assignment churn over that horizon is large: 31.4% on NDC-classes and 40.9% on tags-math-sx.

8.2 Q2: Can incremental maintenance track that target efficiently?

Local TAPS-based repair is substantially faster than rebuilding a static coarsening at every slice, but we treat this as meaningful only after matching the relevant quality target. The fair engineering comparator is therefore the cheapest periodic-recomputation cadence that matches the desired fidelity, while recomputation at every update is reported as an upper-cost reference. Full timing and conductance trajectories for the main temporal runs are moved to Appendix B; the compact validation tables below report the quantities needed to interpret those plots.

8.3 Q3: Where does local maintenance stop being appropriate?

The merge cap K is a work budget, not a guarantee that quality will be unchanged. In mild streams it may be inactive, which is why identical outcomes across several K values should not be interpreted as robustness. In the bulk-update setting, however, the cap binds: for NDC-classes, $K = 1, 2, 5$ suppresses 70, 48, and 19 candidate merges respectively over 50 bulk rounds; for tags-math-sx, $K = 1, 2$ suppresses 11 and 4 merges. This validates the intended role of K as a latency and locality control under sufficiently active update regimes.

The burst experiments identify the boundary where local repair stops being the right tool. At low burst density, only a small dirty region is touched and THGC is much faster than static recomputation. As density rises, most fine nodes are touched in a single burst and the dirty region approaches the whole graph. The density sweep identifies a crossover in which the computational advantage of local repair progressively collapses as the dirty region saturates. This provides a design implication for adaptive resynchronization: dirty-region size can be treated as a runtime signal for deciding when local maintenance should give way to global reconstruction. This is consistent with the theory: the bounds are meaningful when updates are locally sparse and become loose when temporal perturbations are dense.

8.4 Sliding windows, memory, and task-level preservation

Sliding-window expiry tests the deletion-heavy regime that append-only streams miss. In deletion-heavy sliding windows, frequent expiration can erase the computational advantage of local maintenance. THGC's resynchronization mechanism restores exact agreement at reporting points in the completed run, but the resulting runtime can exceed static rebuilding. This experiment therefore delineates a regime in which incremental maintenance should not be used aggressively.

Memory measurements combine process-level RSS with exact state counters. RSS is useful for end to end cost, while counts of coarse nodes, coarse hyperedges, mapping entries, and incidence entries verify that THGC is not hiding an accumulating per-update structure. Downstream experiments are reported as supporting evidence rather than the sole quality claim. Where seeded repeats are available, we report mean and standard deviation; where a downstream split is under-resolved, we use it only to check that coarsening has not destroyed task-relevant information.

8.5 Structural dynamics and repeated-seed validation

We include additional validation experiments that ask four questions: whether the streams induce real structural change, whether incremental repair tracks full static recomputation over long horizons, whether the local-repair knobs bind in nontrivial regimes, and whether the TAPS coupling is stable under repeated segments rather than a single favourable seed.

Table 1: Structural-change and tracking evidence. Static churn is the fraction of coarse-node assignments that changes between consecutive full static recomputations every 200 updates, after best matching. Pair-disagreement is reported on node pairs and is directly comparable to THGC-vs-static deviation. The 4000-update touched fractions measure how much of the fine and coarse graph the stream exercises.

Dataset	Static churn	Static pair-disagree	THGC-vs-static, 750	Static-vs-static, 750	Fine/coarse touched, 4000
NDC-classes	0.275	0.0028	0.00335	0.00340	0.748 / 0.772
tags-math-sx	0.376	0.0072	0.00795	0.00832	0.720 / 0.923

Table 1 separates two phenomena that average conductance can conflate. The streams are not stationary: consecutive static recomputations change 27.5% of assignments on NDC-classes and 37.6% on tags-math-sx, and by 4000 updates they have touched most fine nodes and coarse regions. At the same time, the final THGC-vs-static pair-disagreement after 750 updates is essentially the same scale as the static-vs-static disagreement over that horizon. This supports the intended interpretation of THGC as tracking a moving static reference in localized regimes, not merely exploiting an unevolving benchmark. Full summary tables for the static-churn, stream-dynamics, periodic-recompute, burst, merge-cap, sliding-window, deviation-trajectory, term-scale, and seeded-coupling runs are reported in Appendix A.

Full TAPS temporal runs at target ratio $r = 0.2$ over 20 slices are summarized in Appendix Table 10: on NDC-classes and tags-math-sx, the maintained states remain meaningfully compressed at the final slice while reporting average update times of 0.973s and 2.051s, respectively. These runs control compression at initialization and then allow local temporal contractions to become part of the maintained state; compression-matched comparisons are reported separately where required.

We also repeated the coupling comparison over six independent stream segments, three regimes (natural stream, sliding window, and bulk arrivals), and three downstream split seeds; the full repeated-seed table is in Appendix Table 12. The endpoint comparisons behave as expected for different objectives: PPR-only is strongest on several hyperlink-prediction settings, while conductance-oriented variants improve particular structural criteria. Hyperlink prediction directly rewards temporal interaction proximity, whereas conductance rewards cluster boundary quality. TAPS is designed for structural maintenance rather than as a task-specific hyperlink predictor, and the endpoint comparisons make this distinction explicit. Term-scale diagnostics further show that PPR and cut terms can induce different candidate rankings; low or negative rank correlation in some settings means the components are not merely rescalings of the same signal.

9 Conclusion and Limitations

THGC changes temporal hypergraph coarsening from a sequence of independent static problems into a state-maintenance problem. The central result is that repeated reconstruction is unnecessary when structural change remains localized: previous coarse structure can be reused, repaired, and periodically resynchronized while remaining close to the moving static reference. The burst and sliding-window studies also show where this principle ceases to be advantageous, providing a concrete boundary between local maintenance and global reconstruction.

Dense or bursty updates can make the dirty region global, at which point static recomputation or a short resynchronization interval is preferable. High-density hypergraphs can reduce the benefit of locality because one changed hyperedge may touch many candidate pairs. TAPS uses projection-based scoring, which can discard some simultaneous higher-order information even though the maintained object remains a hypergraph. The method also depends on the threshold τ, merge cap K, residual tolerance, target compression ratio, and recomputation interval k. Finally, downstream labels are not available for every benchmark, so the strongest evidence combines structural fidelity, memory, runtime, and downstream metrics where they can be measured.

Future work includes adaptive resynchronization policies, downstream-aware coupling objectives, stronger theory for deletion-heavy streams, and larger temporal hypergraph benchmarks with labels or held-out hyperlink tasks.

Reproducibility Statement. An implementation of THGC and the experiment runners is provided at `https://github.com/HyperCoarse/THGC`. The main text specifies the temporal hypergraph coarsening objective, the TAPS scoring rule, the local-repair and resynchronization protocol, the comparison baselines, and the evaluation metrics. The appendix provides the complete assumptions and proofs, dataset and benchmark specifications, full validation tables, memory and runtime audits, ablations, and supporting figures. The released repository contains the corresponding source code and scripts; datasets are obtained from their cited public sources and are not redistributed.

REFERENCES

Ali Aghdaei and Zhuo Feng. HyperEF: Spectral hypergraph coarsening by effective-resistance clustering. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2022. doi: 10.1145/3508352.3549438.

Ali Aghdaei, Zhiqiang Zhao, and Zhuo Feng. HyperSF: Spectral hypergraph coarsening via flow-based local clustering. In *Proceedings of the 40th IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2021. doi: 10.1109/ICCAD51958.2021.9643555.

Yaroslav Akhremtsev, Tobias Heuer, Peter Sanders, and Sebastian Schlag. Engineering a direct k-way hypergraph partitioning algorithm. In *Proceedings of the Nineteenth Workshop on Algorithm Engineering and Experiments (ALENEX)*, pp. 28–42. SIAM, 2017. doi: 10.1137/1.9781611974768.3.

Reid Andersen, Fan Chung, and Kevin Lang. Local graph partitioning using pagerank vectors. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, Washington, DC, USA, 2006. IEEE Computer Society.

Alessia Antelmi, Gennaro Cordasco, Mirko Polato, Vittorio Scarano, Carmine Spagnuolo, and Dingqi Yang. A survey on hypergraph representation learning. *ACM Computing Surveys*, 2023. doi: 10.1145/3605776.

Austin R. Benson, Rediet Abebe, Michael T. Schaub, Ali Jadbabaie, and Jon Kleinberg. Simplicial closure and higher-order link prediction. *Proceedings of the National Academy of Sciences*, 2018. ISSN 0027-8424. doi: 10.1073/pnas.1800683115.

C. Berge. *Graphs and Hypergraphs*. North-Holland, Amsterdam, 1973.

Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International Conference on World Wide Web*, Amsterdam, The Netherlands, 1998. Elsevier Science Publishers B. V.

Ümit Çatalyürek, Karen Devine, Marcelo Faraj, Lars Gottesbüren, Tobias Heuer, Henning Meyerhenke, Peter Sanders, Sebastian Schlag, Christian Schulz, Daniel Seemaier, and Dorothea Wagner. More recent advances in (hyper)graph partitioning. *ACM Computing Surveys*, 55(12):1–38, 2023. doi: 10.1145/3571808.

Fan R. K. Chung. *Spectral Graph Theory*. Number 92 in CBMS Regional Conference Series in Mathematics. American Mathematical Society, 1997.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 3 edition, 2009.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3558–3565, 2019. doi: 10.1609/aaai.v33i01.33013558.

Ilse C. F. Ipsen and Rebecca S. Wills. Mathematical properties and analysis of google's pagerank. Technical Report CRSC-TR07-19, North Carolina State University, Department of Mathematics, 2007. Available as a technical report.

G. Karypis and V. Kumar. hmetis: A hypergraph partitioning package. Technical report, University of Minnesota, 1998.

Manoj Kumar, Anurag Sharma, Shashwat Saxena, and Sandeep Kumar. Featured graph coarsening with similarity guarantees. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Andreas Loukas. Graph reduction with spectral and cut guarantees. *Journal of Machine Learning Research*, 20(116):1–42, 2019.

E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein. Temporal graph networks for deep learning on dynamic graphs. arXiv preprint arXiv:2006.10637, 2020.

Hamed Sajadinia, Ali Aghdaei, and Zhuo Feng. Shypar: A spectral coarsening approach to hypergraph partitioning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 45 (2):673–685, 2026. doi: 10.1109/TCAD.2025.3586885.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 2000.

G. W. Stewart and Ji guang Sun. *Matrix Perturbation Theory*. Academic Press, 1990.

Chunyang Wang, Desen Sun, and Yuebin Bai. PiPAD: Pipelined and parallel dynamic GNN training on GPUs. In *Proceedings of the 28th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming (PPoPP)*, 2023.

Hongyang Zhang, Peter Lofgren, and Ashish Goel. Approximate personalized pagerank on dynamic graphs. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 1315–1324. ACM, 2016. doi: 10.1145/2939672.2939804.

Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In Bernhard Schölkopf, John C. Platt, and Thomas Hoffman (eds.), *Advances in Neural Information Processing Systems 19 (NIPS)*. MIT Press, 2007.

Hongkuan Zhou, Da Zheng, Israt Nisa, Vasileios Ioannidis, Xiang Song, and George Karypis. Tgl: A general framework for temporal gnn training on billion-scale graphs. *Proceedings of the VLDB Endowment*, 15(8):1572–1580, 2022. doi: 10.14778/3529337.3529342.

10 LLM Usage Disclosure

In this work, generative AI tools were used only as writing assistants for textual refinement, organization, and readability improvement, and for generating and revising figure and diagram drafts. They were later used to identify possible errors or inconsistencies in the manuscript that could have been missed during manual review. All suggested improvements were evaluated by the authors and implemented manually. Generative AI tools were not used to generate benchmark data, produce scientific results, or make final scientific decisions. No language model is an author. The authors take complete responsibility for the final contents of this paper, including all text, claims, figures, and other artifacts produced with AI assistance.

A Additional Validation Tables

This appendix gives the full summary tables behind the compact validation results in Section 8. The per-update trajectories and per-seed logs are provided as CSV artifacts; here we report the complete row sets from the summary files used for the paper tables.

Table 2: Temporal hypergraph datasets used in experiments. Nodes and hyperedges are reported for the processed benchmark files.

Dataset	Nodes	Hyperedges	Domain
NDC-classes-full	1,161	1,049	Drug-class interactions
contact-high-school	327	7,818	Physical proximity contacts
tags-ask-ubuntu	3,029	147,222	Q&A tag co-occurrences
tags-math-sx	1,629	822,059	Q&A tag co-occurrences

Table 3: Static-recomputation churn under temporal order and shuffled controls. Churn is assignment churn after best matching. Pair-disagree is the node-pair disagreement between consecutive static recomputations. ARI denotes adjusted Rand index and NMI denotes normalized mutual information.

Dataset	Arm	Churn	Pair-disagree	ARI	NMI	Final churn vs. first	New nodes
NDC-classes	temporal	0.275	0.0028	0.540	0.918	0.342	74
NDC-classes	shuffled	0.284	0.0029	0.521	0.915	0.350	74
tags-math-sx	temporal	0.376	0.0072	0.415	0.849	0.463	59
tags-math-sx	shuffled	0.364	0.0070	0.420	0.851	0.460	59

Table 4: Stream-dynamics summary. Fine touched and coarse touched are fractions of initial fine nodes and initial coarse nodes touched by updates over each horizon. Degree growth and new-edge share are reported at the 90th percentile among touched nodes.

Dataset	Horizon	Fine touched	Coarse touched	New nodes	Degree growth p90	New-edge share p90
NDC-classes	200	0.152	0.245	0	0.250	0.200
NDC-classes	750	0.374	0.485	14	0.500	0.333
NDC-classes	4000	0.748	0.772	74	1.667	0.625
tags-math-sx	200	0.201	0.497	3	0.200	0.167
tags-math-sx	750	0.382	0.699	10	0.387	0.279
tags-math-sx	4000	0.720	0.923	59	1.000	0.500

Table 5: Periodic recomputation schedules over 200 updates. Accuracy is reported where the downstream label split is available.

Dataset	Method	Recomps.	Time (s)	Mean cond.	Final cond.	Coarse nodes	Peak RSS (MB)	Accuracy
NDC-classes	THGC	0	0.65	0.5717	0.5701	170	292.3	0.514
NDC-classes	$k = 1$	200	150.7	0.5666	0.5764	161	325.1	0.485
NDC-classes	$k = 2$	100	82.76	0.5666	0.5764	161	252.4	0.485
NDC-classes	$k = 5$	40	32.10	0.5666	0.5764	161	252.2	0.485
NDC-classes	$k = 10$	20	15.61	0.5666	0.5764	161	252.0	0.485
NDC-classes	$k = 20$	10	8.52	0.5668	0.5764	161	252.0	0.485
tags-math-sx	THGC	0	0.47	0.7966	0.7941	176	471.3	0.000
tags-math-sx	$k = 1$	200	148.5	0.7957	0.7952	144	399.0	–
tags-math-sx	$k = 2$	100	68.21	0.7956	0.7952	144	195.6	–
tags-math-sx	$k = 5$	40	29.43	0.7957	0.7952	144	196.0	–
tags-math-sx	$k = 10$	20	14.62	0.7954	0.7952	144	194.9	–
tags-math-sx	$k = 20$	10	9.26	0.7957	0.7952	144	194.1	–

Table 6: Bursty-update density sweep. Speedup compares local repair with full static recomputation for the burst snapshot.

Dataset	Density	Events	Batch dirty frac.	THGC time	Static time	Speedup	Coarse nodes	Conductance
NDC-classes	0.01	132	0.578	0.02	0.92	49.85	252	0.566
NDC-classes	0.05	663	0.780	0.11	1.04	9.40	320	0.534
NDC-classes	0.10	1318	0.800	0.23	1.40	6.04	308	0.496
NDC-classes	0.20	2654	0.803	0.56	2.05	3.63	287	0.481
NDC-classes	0.30	3985	0.803	0.78	1.75	2.26	275	0.461
NDC-classes	0.40	5315	0.803	1.26	2.55	2.02	279	0.456
NDC-classes	0.50	6639	0.803	1.38	2.44	1.77	264	0.440
tags-math-sx	0.01	88	0.295	0.01	0.76	86.19	205	0.793
tags-math-sx	0.05	417	0.465	0.04	0.68	16.42	349	0.773
tags-math-sx	0.10	832	0.552	0.08	0.71	8.45	393	0.749
tags-math-sx	0.20	1645	0.596	0.15	0.72	4.91	397	0.720
tags-math-sx	0.30	2452	0.604	0.21	0.75	3.50	383	0.707
tags-math-sx	0.40	3253	0.607	0.33	0.84	2.55	360	0.686
tags-math-sx	0.50	4041	0.607	0.45	0.97	2.17	336	0.661

Table 7: Merge-cap activation under bulk updates. Suppressed merges are candidate merges prevented by the cap.

Dataset	Cap	Merges	Merges/round	Capped rounds	Suppressed	Mean update (s)	Coarse nodes	Deviation
NDC-classes	unlimited	117	2.34	0	0	0.047	122	0.0375
NDC-classes	1	50	1.00	33	70	0.027	189	0.0105
NDC-classes	2	76	1.52	27	48	0.053	163	0.0129
NDC-classes	3	100	2.00	19	28	0.059	139	0.0308
NDC-classes	5	113	2.26	16	19	0.055	126	0.0228
NDC-classes	20	117	2.34	0	0	0.076	122	0.0375
tags-math-sx	unlimited	35	0.70	0	0	0.078	108	0.0167
tags-math-sx	1	28	0.56	10	11	0.065	115	0.0152
tags-math-sx	2	34	0.68	4	4	0.077	109	0.0178
tags-math-sx	3	35	0.70	0	0	0.078	108	0.0167
tags-math-sx	5	35	0.70	0	0	0.081	108	0.0167
tags-math-sx	20	35	0.70	0	0	0.082	108	0.0167

Table 8: Sliding-window expiry with matched final coarse-node counts.

Dataset	Updates	Resyncs	Turnover	THGC+resync (s)	Static (s)	Final THGC/static nodes	Final deviation	Static churn
NDC-classes	300	6	0.021	26.27	19.42	162 / 162	0.000	0.208
tags-math-sx	300	6	0.021	68.95	6.90	142 / 142	0.000	0.320

Table 9: Long-horizon exact deviation trajectory over 750 updates. Static-vs-static is measured over the same 750-update horizon.

Dataset	Mean deviation	Max deviation	Final deviation	Slope / 1k	Static-vs-static	Static assignment churn
NDC-classes	0.0030	0.0037	0.00335	0.0023	0.00340	0.314
tags-math-sx	0.0073	0.0085	0.00795	0.0025	0.00832	0.409

Table 10: Full TAPS temporal run at target ratio $r = 0.2$ over 20 slices. Rows aggregate four repeated initialization labels; all use the same matched-compression initializer followed by full TAPS temporal repair. Update time is averaged over slices t02–t20.

Dataset	Rows	Conductance	Coarse ratio, first $\rightarrow$ final	Update time	Mapping-change rate
NDC-classes	80	0.1780 ± 0.0227	$0.200 \rightarrow 0.085$	0.973 ± 1.467s	0.225
tags-math-sx	80	0.9142 ± 0.0026	$0.200 \rightarrow 0.190$	2.051 ± 3.020s	0.069

Table 11: TAPS term-scale diagnostics for 1000 sampled candidate pairs at $\alpha = 0.5$. Spearman columns compare the ranking induced by the blended score with each component.

Dataset	PPR nonzero	Cut nonzero	PPR share	Spearman blend/cut	Spearman blend/PPR	Spearman PPR/cut
NDC-classes	0.216	1.000	0.344	0.985	0.469	0.369
tags-math-sx	0.375	1.000	0.213	0.740	0.411	-0.190

Table 12: Seeded coupling comparison. Entries are mean ± standard deviation across six stream segments and three downstream split seeds. AUROC denotes area under the receiver operating characteristic curve for hyperlink prediction.

Dataset	Regime	Variant	Conductance	Deviation	AUROC	Shock AUROC
NDC-classes	bulk	PPR-only	0.334 ± 0.026	0.0090 ± 0.0003	0.855 ± 0.175	0.891 ± 0.125
NDC-classes	bulk	conductance-only	0.245 ± 0.026	0.0269 ± 0.0046	0.763 ± 0.185	0.503 ± 0.007
NDC-classes	bulk	heavy-edge	0.263 ± 0.020	0.0292 ± 0.0055	0.779 ± 0.188	0.498 ± 0.000
NDC-classes	bulk	TAPS, $\alpha = 0.5$	0.253 ± 0.018	0.0254 ± 0.0051	0.792 ± 0.197	0.499 ± 0.004
NDC-classes	stream	PPR-only	0.332 ± 0.025	0.0146 ± 0.0140	0.947 ± 0.010	0.947 ± 0.010
NDC-classes	stream	conductance-only	0.309 ± 0.026	0.0083 ± 0.0017	0.891 ± 0.017	0.808 ± 0.096
NDC-classes	stream	heavy-edge	0.317 ± 0.035	0.0085 ± 0.0040	0.916 ± 0.045	0.721 ± 0.175
NDC-classes	stream	TAPS, $\alpha = 0.5$	0.307 ± 0.023	0.0096 ± 0.0023	0.887 ± 0.038	0.768 ± 0.131
NDC-classes	window	PPR-only	0.330 ± 0.025	0.0076 ± 0.0018	0.949 ± 0.009	0.947 ± 0.015
NDC-classes	window	conductance-only	0.292 ± 0.029	0.0088 ± 0.0011	0.888 ± 0.026	0.874 ± 0.030
NDC-classes	window	heavy-edge	0.280 ± 0.020	0.0125 ± 0.0025	0.899 ± 0.034	0.846 ± 0.031
NDC-classes	window	TAPS, $\alpha = 0.5$	0.298 ± 0.017	0.0087 ± 0.0037	0.885 ± 0.044	0.872 ± 0.034
tags-math-sx	bulk	PPR-only	0.847 ± 0.005	0.0111 ± 0.0003	0.644 ± 0.059	0.650 ± 0.051
tags-math-sx	bulk	conductance-only	0.843 ± 0.005	0.0162 ± 0.0020	0.633 ± 0.060	0.504 ± 0.007
tags-math-sx	bulk	heavy-edge	0.835 ± 0.008	0.0169 ± 0.0025	0.635 ± 0.069	0.673 ± 0.016
tags-math-sx	bulk	TAPS, $\alpha = 0.5$	0.840 ± 0.005	0.0184 ± 0.0035	0.615 ± 0.061	0.498 ± 0.000
tags-math-sx	stream	PPR-only	0.846 ± 0.005	0.0075 ± 0.0007	0.771 ± 0.007	0.761 ± 0.006
tags-math-sx	stream	conductance-only	0.884 ± 0.006	0.1596 ± 0.0577	0.518 ± 0.020	0.515 ± 0.021
tags-math-sx	stream	heavy-edge	0.830 ± 0.007	0.0212 ± 0.0043	0.741 ± 0.015	0.734 ± 0.013
tags-math-sx	stream	TAPS, $\alpha = 0.5$	0.876 ± 0.008	0.1238 ± 0.0544	0.524 ± 0.017	0.522 ± 0.015
tags-math-sx	window	PPR-only	0.839 ± 0.005	0.0091 ± 0.0008	0.768 ± 0.007	0.759 ± 0.007
tags-math-sx	window	conductance-only	0.878 ± 0.015	0.1028 ± 0.0644	0.534 ± 0.036	0.527 ± 0.021
tags-math-sx	window	heavy-edge	0.825 ± 0.009	0.0258 ± 0.0043	0.729 ± 0.013	0.726 ± 0.014
tags-math-sx	window	TAPS, $\alpha = 0.5$	0.874 ± 0.014	0.0907 ± 0.0677	0.553 ± 0.047	0.545 ± 0.043

B Main-Text Supporting Figures

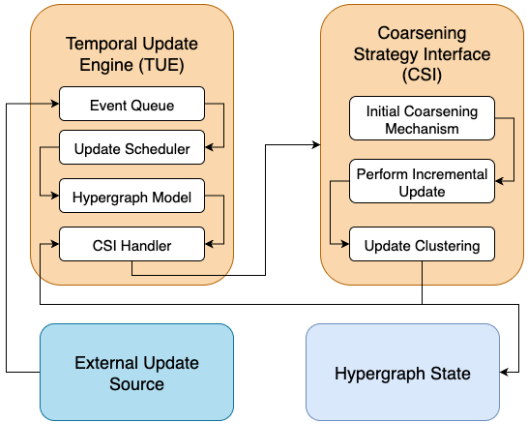


Figure 1: Overall architecture of THGC.

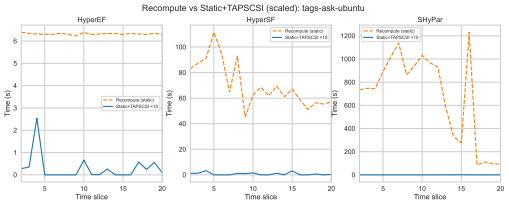


Figure 2: Per-slice update time on tags-ask-ubuntu. THGC values are scaled for visibility.

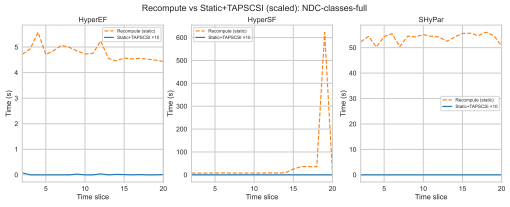


Figure 3: Per-slice update time on NDC-classes-full. THGC values are scaled for visibility.

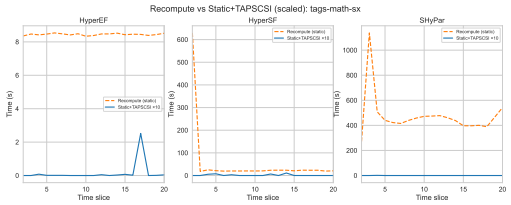


Figure 4: Per-slice update time on tags-math-sx. THGC values are scaled for visibility.

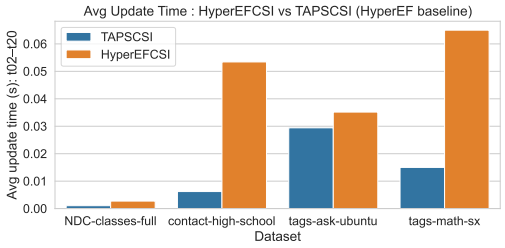


Figure 5: Average update time across datasets for static recomputation and local TAPS-based repair.

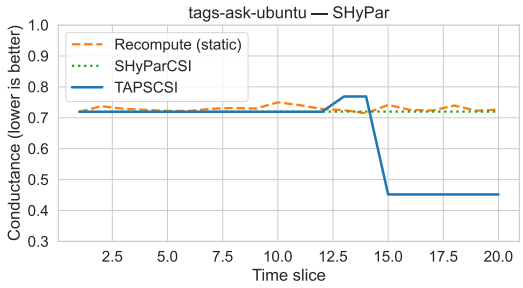


Figure 6: Conductance over time for Ask-Ubuntu under SHyPar recomputation and local CSI variants.

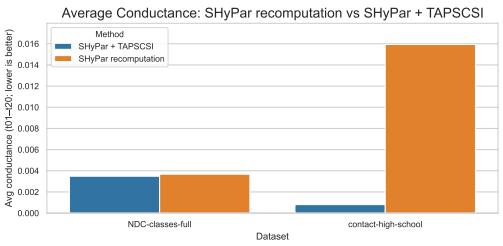


Figure 7: Average conductance over reporting slices for SHyPar recomputation and SHyPar with local TAPS-based repair.

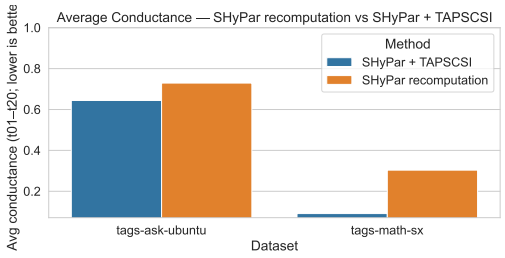


Figure 8: Average conductance over reporting slices for SHyPar recomputation and SHyPar with local TAPS-based repair.

C Assumptions

Assumption 1 (Temporal random-walk model)**.** *Each $P(t)$ is row-stochastic and well-defined with $\delta_{\min}(t) := \min_i D_{ii}(t) > 0$, and temporal accumulation of hyperedge weights uses the exponentially decayed update rule specified in the paper, inducing bounded per-step changes $\|A(t{+}1) - A(t)\|_1$ and $\|D(t{+}1) - D(t)\|_1$.*

This holds because we form $A(t)$ via exponentially decayed hyperedge accumulation and set $P(t) = D(t)^{-1}A(t)$, which is row-stochastic once isolated nodes are removed/regularized, ensuring $\delta_{\min}(t) > 0$. The accumulator bounds per-step $\|\Delta A\|_1, \|\Delta D\|_1$, yielding the perturbation bound on $\|P(t{+}1) - P(t)\|_1$ used in Lemma A.2.

Assumption 2 (PPR and diffusion operators). *Personalized PageRank uses teleportation $\gamma \in (0,1)$ with the resolvent form $\pi_s(t) = (1-\gamma)\sum_{k\geq 0}\gamma^k(P(t)^\top)^k h_s = (1-\gamma)(I-\gamma P(t)^\top)^{-1}h_s$ (Brin & Page, 1998; Ipsen & Wills, 2007).*

We adopt seeded PageRank with teleportation $\gamma \in (0,1)$, giving the standard resolvent/series form. This directly yields the sensitivity constant $\gamma/(1-\gamma)$ used in Lemma A.1 and Theorem 1.

Assumption 3 (TAPS definition). *$S_{\mathrm{ppr}}(u,v,t) = \pi_u^t[v] + \pi_v^t[u] \in [0,2]$ and $S_{\mathrm{cut}}(u,v,t) = 1 - \phi_t(C) \in [0,1]$ for a set $C \ni u,v$, so $\mathrm{TAPS}(u,v,t) \in [0,2]$ and is symmetric in (u,v).*

Symmetry follows from the symmetric definition of S_{ppr} and the use of any admissible $C \ni \{u,v\}$.

Assumption 4 (Restricted spectral approximation for coarsening). *Let $\mathcal{U}_t = \mathrm{im}(\Pi_t^\top)$ be the subspace of cluster-constant signals, where Π_t denotes the degree-aware prolongation/cluster-indicator/coarsening matrix, and suppose the lifted coarse Laplacian $\widetilde{L}(t)$ satisfies the slice-wise restricted approximation $\forall x \in \mathcal{U}_t$ (Chung, 1997; Loukas, 2019):*

$$(1-\varepsilon_t)\, x^\top L(t)\, x \leq x^\top \widetilde{L}(t)\, x \leq (1+\varepsilon_t)\, x^\top L(t)\, x$$

for some $\varepsilon_t \geq 0$ depending on intra-cluster neighbourhood similarity and cluster boundary conductance in $G(t)$ (Loukas, 2019).

Assumption 5 (Operator perturbation induced by restricted approximation). *There exists a normalization constant $c_{\mathrm{op}} > 0$ such that the random-walk operators on $\mathcal{U}_t$ satisfy $\|P_{\mathrm{rw}}(t) - \widetilde{P}_{\mathrm{rw}}(t)\| \leq c_{\mathrm{op}}\,\varepsilon_t$ (Loukas, 2019).*

D Downstream Task Evaluation

Beyond conductance, we evaluate whether the coarsened hypergraph preserves structural information relevant to downstream learning tasks. On NDC-classes-full, we compute structural node features from each full snapshot and from the corresponding THGC-coarsened snapshot, aggregate fine-node features to coarse nodes by the maintained mapping, train a lightweight softmax node classifier with fixed random seeds, and lift coarse predictions back to fine nodes through the same mapping for comparison. This is a task-level preservation test inspired by Hypergraph Neural Network evaluation (Feng et al., 2019), not an additional claim that a deep message-passing architecture is required for THGC.

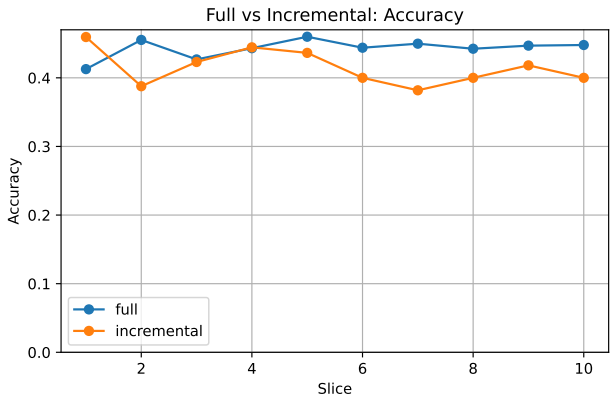


Figure 9: Per-slice node classification accuracy: full hypergraph vs. THGC-coarsened hypergraph (NDC-classes-full, t01–t10). The coarsened snapshots retain most task-level accuracy across evaluated slices.

As shown in Figure 9, the THGC-coarsened snapshots retain 93.9% mean accuracy relative to the full snapshots across the evaluated slices, with median preservation of 92.0%. Running the classifier on the coarsened representation is also cheaper because the computational graph is proportionally smaller than the original (Feng et al., 2019).

E MEMORY USAGE AND SCALABILITY

We evaluate memory efficiency and scalability along two axes: the number of temporal slices processed and the size of the underlying hypergraph via subsampling.

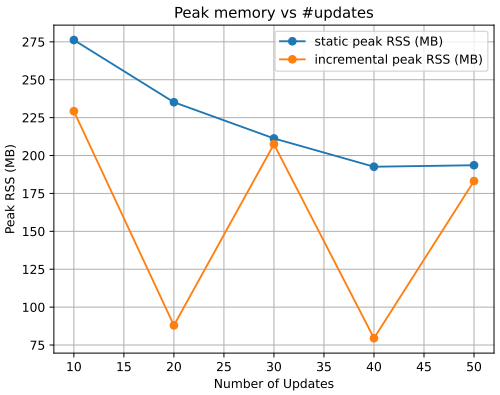


Figure 10: Peak RSS (MB) vs. number of slices: THGC vs. static recomputation. Incremental consistently uses less memory.

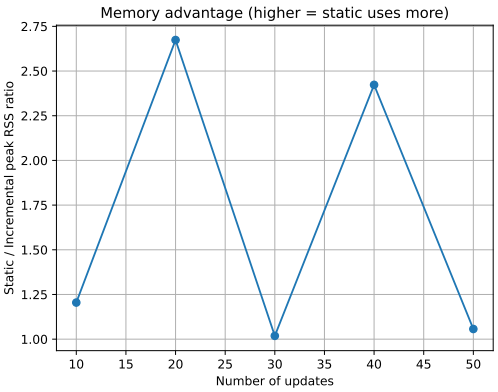


Figure 11: Static/incremental peak RSS ratio vs. number of slices. The ratio exceeds 1 and grows with more slices.

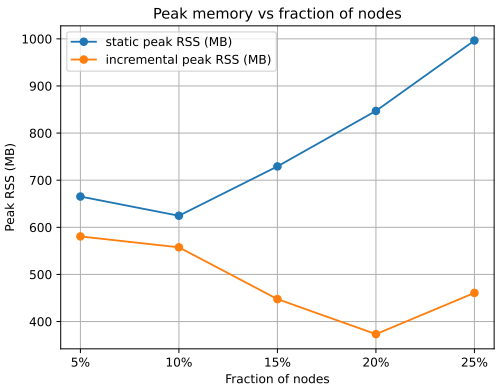


Figure 12: Peak RSS (MB) vs. hypergraph size (subsampling fraction). THGC memory grows sub-linearly.

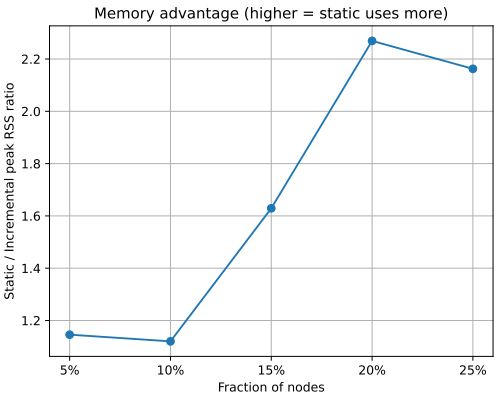


Figure 13: Static/incremental peak RSS ratio vs. graph size fraction. The ratio increases with graph size.

THGC's peak RSS is consistently lower than static recomputation across all dataset sizes and temporal horizons. The static/incremental ratio grows with both the number of slices and the graph size, showing THGC's memory advantage becomes more pronounced at scale. Static total processing time grows linearly with the number of slices, while THGC's total time stays near-constant due to the amortized cost of local incremental updates.

F HYPERPARAMETER SENSITIVITY

THGC uses three main hyperparameters: the mixing factor α (balance between PPR and conductance), the coupling threshold τ, and the recomputation interval k. These correspond to the standard trade-off between diffusion proximity, cut quality, and update frequency in local graph/hypergraph maintenance (Andersen et al., 2006; Zhang et al., 2016; Zhou et al., 2007). We performed a

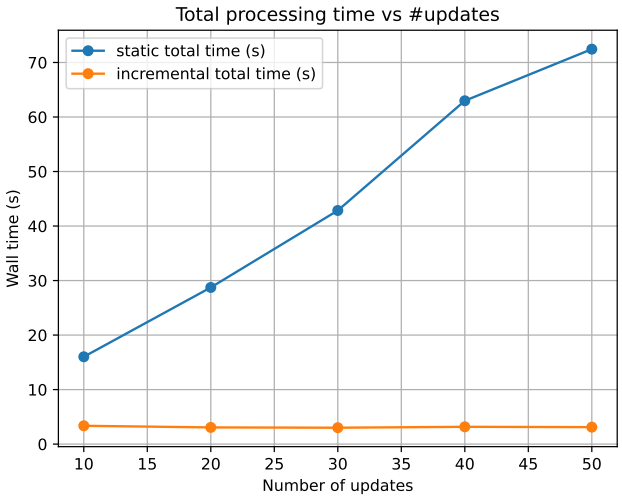


Figure 14: Total processing time vs. number of slices: static recomputation grows linearly while THGC remains near-constant due to local incremental updates.

comprehensive grid search over these parameters, recording mean update time and final conductance for each configuration.

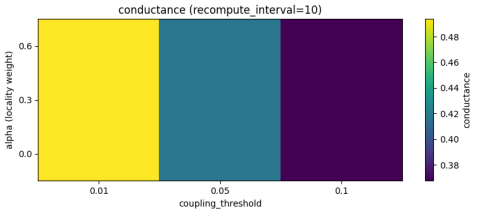


Figure 15: Conductance across hyperparameter values (α, τ, k). Conductance is relatively insensitive over the tested range.

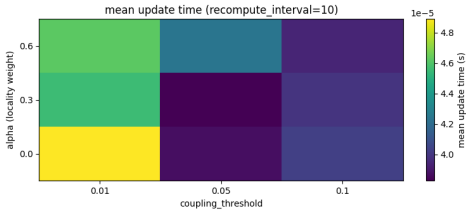


Figure 16: Mean update time across hyperparameter values. Timing varies with k but conductance quality remains stable.

Conductance is relatively insensitive across the tested (α, τ, k) region. This indicates that coarse cut quality is not highly sensitive to these parameters, although other fidelity metrics may distinguish configurations even when conductance is flat. The recomputation interval k primarily affects update time, enabling users to tune the latency/resynchronization trade-off.

G Full TAPS vs. TAPS-Lite Structural Variant

In our implementation, **full TAPS** is the method described in the main paper, combining PPR and conductance (Andersen et al., 2006; Zhang et al., 2016; Zhou et al., 2007). **TAPS-Lite** is a stripped structural baseline used for fast comparison, described in Appendix K.

Full TAPS can improve conductance by using the temporally-aware PPR component together with local cut information. TAPS-Lite is a faster structural alternative when update latency is critical and a quality trade-off is acceptable. All main-text experiments use full TAPS unless explicitly labelled otherwise.

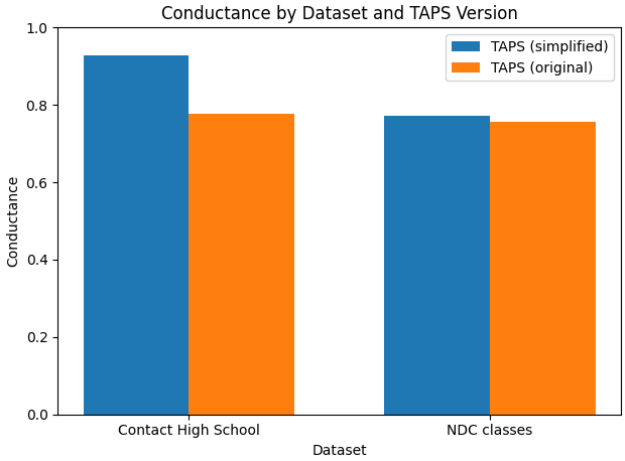


Figure 17: Full TAPS (PPR plus conductance) versus TAPS-Lite on average conductance and update time across datasets. Full TAPS improves structural quality at a modest increase in update time, while TAPS-Lite is faster.

H Proofs of Theoretical Results

H.1 Lemma A.1 (PPR perturbation bound).

For any two row-stochastic transitions P_1, P_2 and $\gamma \in (0, 1)$,

$$\|\pi_s(P_2) - \pi_s(P_1)\|_1 \;\le\; \frac{\gamma}{1-\gamma}\,\|P_2 - P_1\|_1,$$

and consequently $\left|S^{(2)}_{\mathrm{ppr}}(u,v) - S^{(1)}_{\mathrm{ppr}}(u,v)\right| \le \frac{2\gamma}{1-\gamma}\,\|P_2 - P_1\|_1$ (Zhang et al., 2016).

Proof. Let $\pi_s(P) = (1-\gamma)\,(I - \gamma P^\top)^{-1} h_s$, where h_s is the personalization vector, and note that P row-stochastic implies $P^\top$ column-stochastic. Since $\|\gamma P^\top\|_1 \le \gamma < 1$,

$$(I - \gamma P^\top)^{-1} = \sum_{k=0}^{\infty} (\gamma P^\top)^k, \quad \left\|(I - \gamma P^\top)^{-1}\right\|_1 \le \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma}.$$

Write $R(P) = (I - \gamma P^\top)^{-1}$. The resolvent identity yields (Stewart & guang Sun, 1990)

$$R(P_2) - R(P_1) = R(P_2)\,\gamma(P_2^\top - P_1^\top)\,R(P_1).$$

Thus,

$\|\pi_s(P_2) - \pi_s(P_1)\|_1 = (1-\gamma)\,\|(R(P_2) - R(P_1))h_s\|_1 \le (1-\gamma)\,\|R(P_2)\|_1\,\gamma\,\|P_2 - P_1\|_1\,\|R(P_1)\|_1\,\|h_s\|_1.$

Using $\|R(P_i)\|_1 \le (1-\gamma)^{-1}$ and $\|h_s\|_1 = 1$ gives

$$\|\pi_s(P_2) - \pi_s(P_1)\|_1 \le \frac{\gamma}{1-\gamma}\,\|P_2 - P_1\|_1.$$

The symmetric score satisfies $S_{\mathrm{ppr}}(u,v) = \pi_u(P)[v] + \pi_v(P)[u]$, so the triangle inequality adds a factor 2. □

H.2 Lemma A.2 (Transition perturbation from temporal updates).

Let $\Delta A = A(t{+}1) - A(t)$ and $\Delta D = D(t{+}1) - D(t)$ and assume $\delta_{\min}(t) = \min_i D_{ii}(t) > 0$ and $\delta_{\min}(t{+}1) > 0$. Then with $P = D^{-1}A$,

$$\|P(t{+}1) - P(t)\|_1 \;\le\; \frac{\|\Delta A\|_1}{\delta_{\min}(t{+}1)} \;+\; \frac{\|A(t)\|_1\,\|\Delta D\|_1}{\delta_{\min}(t)\delta_{\min}(t{+}1)},$$

and under the accumulator, both $\|\Delta A\|_1$ and $\|\Delta D\|_1$ are bounded by sums of decayed per-hyperedge increments over one time step.

Proof. Using $P(t) = D(t)^{-1}A(t)$ and $A(t{+}1) = A(t) + \Delta A$,

$$P(t{+}1) - P(t) = D(t{+}1)^{-1}\Delta A + \big(D(t{+}1)^{-1} - D(t)^{-1}\big)A(t).$$

The first term satisfies $\|D(t{+}1)^{-1}\Delta A\|_1 \le \delta_{\min}(t{+}1)^{-1}\|\Delta A\|_1$. For the second, the exact inverse identity for invertible diagonal matrices gives (Stewart & guang Sun, 1990)

$$D(t{+}1)^{-1} - D(t)^{-1} = -D(t{+}1)^{-1}\Delta D\, D(t)^{-1}.$$

Therefore

$$\|\big(D(t{+}1)^{-1} - D(t)^{-1}\big)A(t)\|_1 \le \frac{\|A(t)\|_1\,\|\Delta D\|_1}{\delta_{\min}(t)\delta_{\min}(t{+}1)}.$$

Combining the two terms yields the stated bound. Under exponential accumulation, one-step changes aggregate decayed increments of updated hyperedges, bounding $\|\Delta A\|_1$ and $\|\Delta D\|_1$. □

H.3 Lemma A.3 (From restricted spectral approximation to operator proximity).

Under Assumption 4, there exists $c_{\mathrm{op}} > 0$ depending on normalization such that $\|P_{\mathrm{rw}}(t) - \widetilde{P}_{\mathrm{rw}}(t)\|_{\mathcal{U}_t \to \mathcal{U}_t} \le c_{\mathrm{op}}\,\varepsilon_t$, and cut values for unions of clusters are preserved within $(1 \pm \varepsilon_t)$.

Proof sketch. Loukas (2019) shows that restricted quadratic-form control yields spectral/cut guarantees and stability of low-frequency subspaces; random-walk operator proximity follows by translating between $L_{\mathrm{rw}} = I - P_{\mathrm{rw}}$ and quadratic-form distortion on $\mathcal{U}_t$ (Chung, 1997).

Proof. Restricted quadratic-form control on $\mathcal{U}_t = \mathrm{im}(\Pi_t)$ reads

$$(1 - \varepsilon_t)\, x^\top L\, x \le x^\top \widetilde{L}\, x \le (1 + \varepsilon_t)\, x^\top L\, x, \quad \forall x \in \mathcal{U}_t.$$

This yields multiplicative cut preservation on unions of clusters via the equivalence between restricted Rayleigh quotients and cut capacities for cluster-constant vectors (Chung, 1997; Loukas, 2019). Using $L_{\mathrm{rw}} = I - P_{\mathrm{rw}}$ and the degree-aware normalization,

$$\|P_{\mathrm{rw}} - \widetilde{P}_{\mathrm{rw}}\|_{\mathcal{U}_t \to \mathcal{U}_t} = \|L_{\mathrm{rw}} - \widetilde{L}_{\mathrm{rw}}\|_{\mathcal{U}_t \to \mathcal{U}_t} \le c_{\mathrm{op}}\varepsilon_t,$$

for a constant c_{op} depending only on the chosen normalization, completing the claim. □

H.4 Proof of Theorem 1.

Apply Lemma A.1 with $P_1 = P(t)$ and $P_2 = P(t{+}1)$ to bound the PPR term, and write $\big|S_{\mathrm{cut}}(u, v, t{+}1) - S_{\mathrm{cut}}(u, v, t)\big| = \big|\phi_{t+1}(C) - \phi_t(C)\big|$ for any fixed $C \ni u, v$, yielding the stated inequality (Zhang et al., 2016).

Proof. Lemma A.1 with $P_1 = P(t), P_2 = P(t{+}1)$ gives

$$\big|S_{\mathrm{ppr}}(u, v, t{+}1) - S_{\mathrm{ppr}}(u, v, t)\big| \le \frac{2\gamma}{1 - \gamma}\,\|P(t{+}1) - P(t)\|_1.$$

For the cut term and any admissible $C \ni u, v$,

$$\big|S_{\mathrm{cut}}(u, v, t{+}1) - S_{\mathrm{cut}}(u, v, t)\big| = \big|\phi_{t+1}(C) - \phi_t(C)\big|.$$

Combining the mixture $\mathrm{TAPS} = \alpha\, S_{\mathrm{ppr}} + (1{-}\alpha)\, S_{\mathrm{cut}}$ and Lemma A.2 to bound $\|P(t{+}1) - P(t)\|_1$ via $\Delta A, \Delta D$ yields the stated inequality. □

H.5 Proof of Theorem 2.

For the coarse–original slice-wise bound, use Lemma A.3 to control $\|P_{\mathrm{rw}}(t) - \widetilde{P}_{\mathrm{rw}}(t)\|$ by $c_{\mathrm{op}}\varepsilon_t$ on $\mathcal{U}_t$ and transfer to PPR via Lemma A.1, while the cut part follows from multiplicative cut preservation on unions of clusters, giving the first inequality (Loukas, 2019). For the end to end bound from t to $t{+}1$, add and subtract $\mathrm{TAPS}(u, v, t{+}1)$ and apply Theorem 1 and the slice-wise coarse–original bound at time $t{+}1$, then combine by the triangle inequality to obtain the stated inequality Loukas (2019).

Proof. Slice-wise coarse-original at time t: Lemma A.3 implies $\|P_{\mathrm{rw}}(t) - \widetilde{P}_{\mathrm{rw}}(t)\|_{\mathcal{U}_t \to \mathcal{U}_t} \le c_{\mathrm{op}}\varepsilon_t$. Applying Lemma A.1 within $\mathcal{U}_t$,

$$\left|S^{\mathrm{fine}}_{\mathrm{ppr}}(u,v,t) - S^{\mathrm{coarse}\uparrow}_{\mathrm{ppr}}(u,v,t)\right| \le \frac{2\gamma}{1-\gamma}\, c_{\mathrm{op}}\varepsilon_t,$$

while multiplicative cut preservation on unions of clusters gives

$$\left|S^{\mathrm{fine}}_{\mathrm{cut}}(u,v,t) - S^{\mathrm{coarse}\uparrow}_{\mathrm{cut}}(u,v,t)\right| \le \varepsilon_t\, \phi^{\mathrm{fine}}_t(C).$$

Combining under the TAPS mixture yields the slice-wise bound. For the end to end bound across $t \to t{+}1$, add and subtract $\mathrm{TAPS}^{\mathrm{coarse}\uparrow}(u,v,t{+}1)$, apply Theorem 1 at $t{+}1$ and the slice-wise coarse-original bound at $t{+}1$, and finish with the triangle inequality; constants aggregate as claimed. □

H.6 Additional spectral comparability guarantee

Theorem 3 (Mutual spectral comparability of two valid coarse representations)**.** *Fix a time slice t and let $L(t)$ be the fine combinatorial Laplacian of the time-weighted clique-expansion graph constructed from the temporal hypergraph at slice t (Chung, 1997). Let $\tilde{L}^B(t)$ and $\tilde{L}^H(t)$ be the lifted coarse Laplacians induced by the baseline partition and the THGC partition, respectively. If restricted spectral approximation holds on the corresponding cluster-constant subspaces with errors $\varepsilon_B(t), \varepsilon_H(t) \in [0,1)$, then on their intersection the two lifted forms are mutually comparable up to the factors*

$$\frac{1-\varepsilon_H(t)}{1+\varepsilon_B(t)} \quad \textit{and} \quad \frac{1+\varepsilon_H(t)}{1-\varepsilon_B(t)},$$

and their normalized operator deviation is at most $\varepsilon_B(t) + \varepsilon_H(t)$.

Assumption 6 (Restricted spectral approximation)**.** *For $\star \in \{B, H\}$, there exists $\varepsilon_\star(t) \in [0,1)$ such that*

$$(1-\varepsilon_\star)x^\top L(t)x \le x^\top \tilde{L}^\star(t)x \le (1+\varepsilon_\star)x^\top L(t)x, \quad \forall x \in U_t^\star.$$

Lemma 1 (Two-sided comparison via restricted forms)**.** *For all $x \in U_t := U_t^B \cap U_t^H$,*

$$\frac{1-\varepsilon_H}{1+\varepsilon_B}\, x^\top \tilde{L}^B x \;\le\; x^\top \tilde{L}^H x \;\le\; \frac{1+\varepsilon_H}{1-\varepsilon_B}\, x^\top \tilde{L}^B x,$$

where for brevity $\varepsilon_\star = \varepsilon_\star(t)$, $L = L(t)$, and $\tilde{L}^\star = \tilde{L}^\star(t)$.

Proof. Upper bound: For $x \in U_t \subseteq U_t^H$,

$$x^\top \tilde{L}^H x \le (1+\varepsilon_H)\, x^\top L x.$$

Also $x \in U_t \subseteq U_t^B$ implies $x^\top L x \le \frac{1}{1-\varepsilon_B}\, x^\top \tilde{L}^B x$. Combining yields

$$x^\top \tilde{L}^H x \le \frac{1+\varepsilon_H}{1-\varepsilon_B}\, x^\top \tilde{L}^B x.$$

Lower bound: For $x \in U_t \subseteq U_t^H$,

$$x^\top \tilde{L}^H x \ge (1-\varepsilon_H)\, x^\top L x,$$

and for $x \in U_t \subseteq U_t^B$, $x^\top L x \ge \frac{1}{1+\varepsilon_B}\, x^\top \tilde{L}^B x$. Combining yields

$$x^\top \tilde{L}^H x \ge \frac{1-\varepsilon_H}{1+\varepsilon_B}\, x^\top \tilde{L}^B x.$$

□

Lemma 2 (Additive and normalized bounds)**.** *For all $x \in U_t$,*

$$\left| x^\top (\tilde{L}^H - \tilde{L}^B) x \right| \le (\varepsilon_B + \varepsilon_H)\, x^\top L x,$$

and

$$\left\| L^{-\frac{1}{2}} (\tilde{L}^H - \tilde{L}^B) L^{-\frac{1}{2}} \right\|_{U_t \to U_t} \le \varepsilon_B + \varepsilon_H.$$

Proof. Triangle inequality on quadratic forms gives

$$\begin{aligned}\left|x^\top(\tilde{L}^H - \tilde{L}^B)x\right| &\le \left|x^\top(\tilde{L}^H - L)x\right| + \left|x^\top(L - \tilde{L}^B)x\right| \\ &\le \varepsilon_H\, x^\top L x + \varepsilon_B\, x^\top L x = (\varepsilon_B + \varepsilon_H)\, x^\top L x,\end{aligned}$$

using the restricted bounds for $x \in U_t \subseteq U_t^H \cap U_t^B$. The operator inequality follows by taking the supremum of the Rayleigh quotient over $x \in U_t \setminus \{0\}$:

$$\left\|L^{-\frac{1}{2}}(\tilde{L}^H - \tilde{L}^B)L^{-\frac{1}{2}}\right\|_{U_t \to U_t} = \sup_{x \in U_t \setminus \{0\}} \frac{\left|x^\top(\tilde{L}^H - \tilde{L}^B)x\right|}{x^\top L x} \le \varepsilon_B + \varepsilon_H.$$

□

Proof of Theorem. Apply the two-sided comparison lemma to obtain the multiplicative bounds, noting that

$$\frac{1-\varepsilon_H}{1+\varepsilon_B} \le \frac{x^\top \tilde{L}^H x}{x^\top \tilde{L}^B x} \le \frac{1+\varepsilon_H}{1-\varepsilon_B}, \quad \forall x \in U_t.$$

The additive and normalized operator bounds follow from the previous lemma. □

Corollary (random-walk operator closeness). Let $\tilde{P}^\star(t)$ be the coarse random-walk operator induced by $\tilde{L}^\star(t)$ under standard normalization, and assume the usual Laplacian-to-random-walk transfer bound on $U_t^\star$ with constant $c_{\mathrm{op}} > 0$. Then

$$\left\|\tilde{P}^H(t) - \tilde{P}^B(t)\right\|_{U_t \to U_t} \le c_{\mathrm{op}}\left(\varepsilon_B(t) + \varepsilon_H(t)\right).$$

Sketch. Combine the restricted approximation for each partition with the transfer bound to control $\|\tilde{P}^\star(t) - P_{\mathrm{rw}}(t)\|$ and use the triangle inequality.

I Derivation of Time Complexities

I.1 Notation and assumptions

We use the same notation as the main text: ΔE is the batch of changed hyperedges, M is the number of dirty coarse nodes, W is the recent accumulation window, $\bar{b}$ is the average changed-hyperedge size, C_{uv} is the pairwise co-occurrence count for (u, v), $C = \max C_{uv}$ over all evaluated candidate pairs, and $\mathcal{Q}$ is the local candidate-pair set. Candidate pairs are restricted to the 1-hop coarse neighborhood of dirty regions, and contractions per round are greedily selected to be disjoint with a per-round cap K. We assume event handling via a min-heap queue in the TUE (Cormen et al., 2009), and local caches or maps to maintain recent co-occurrence statistics for TAPS on candidate pairs.

I.2 Pairwise TAPS complexity

We derive the complexity for computing the TAPS score $S(u, v, t) = \alpha\, S_{\mathrm{ppr}}(u, v, t) + (1 - \alpha)\, S_{\mathrm{cut}}(u, v, t)$ for a single candidate pair (u, v). Write the cost as $T_{\mathrm{pair}}(u, v) = T_{\mathrm{ppr}}(u, v) + T_{\mathrm{cut}}(u, v)$.

Random-walk/PPR component. In our incremental setting, we use cached, sparsity-controlled local neighborhoods together with an index keyed by co-occurrences of (u, v) within the recent window W; evaluating the symmetric PPR-based score reduces to merging up to C_{uv} contributions with a heap or balanced map, giving $T_{\mathrm{ppr}}(u, v) = O(C_{uv} \log C_{uv})$. The $\log C_{uv}$ factor comes from insert/find on heap or tree-backed associative structures when accumulating per-neighbor contributions across temporal co-occurrences, following the local-update philosophy of dynamic PPR maintenance (Zhang et al., 2016).

Cut component. We compute the local conductance-based term using the windowed accumulator and clique-expansion adjacency; in the bounded neighborhood regime the work scales with updated incident mass in the window and the average hyperedge size $\bar{b}$, giving $T_{\mathrm{cut}}(u, v) = O(W \cdot \bar{b})$. Only hyperedges in the recent window touching the candidate cluster preimage contribute, each inducing $O(\bar{b})$ pairwise updates under clique expansion.

Combined per-pair cost. Summing the above yields

$$T_{\mathrm{pair}}(u, v) \;=\; O\big(C_{uv} \log C_{uv} \;+\; W \cdot \bar{b}\big).$$

I.3 PER-UPDATE COMPLEXITY

A temporal batch ΔE affects a localized dirty region at the coarse level with M dirty coarse nodes and an induced set of candidate pairs $\mathcal{Q}$ taken from the 1-hop coarse neighborhoods around these nodes. We decompose the total update cost into: structural maintenance, candidate generation and scoring, and greedy contraction.

(i) Structural maintenance. Applying the fine-level modification and updating temporal accumulators for adjacency/degree under clique expansion costs

$$T_{\text{struct}} = O\left(\sum_{e \in \Delta E} |e|^2\right),$$

which becomes $O(|\Delta E|\bar{b}^2)$ under bounded changed-hyperedge size. Dirty-region identification at the coarse level is linear in touched incidences and is absorbed into the local terms below.

(ii) Candidate generation and scoring. Restricting to 1-hop neighborhoods around M dirty coarse nodes yields $|\mathcal{Q}| \le O(M^2)$ candidate pairs in the worst case; scoring each pair with TAPS costs $O(C \log C + W \cdot \bar{b})$ by the per-pair bound above, hence

$$T_{\text{cand+score}} \;=\; O\big(|\mathcal{Q}|\,(C \log C + W \cdot \bar{b})\big).$$

In practice, C is the maximum recent co-occurrence over all evaluated pairs and is bounded under windowing and locality.

(iii) Greedy contraction with cap K. Selecting up to K disjoint pairs via a max-heap costs $O(|\mathcal{Q}|) + O(K \log |\mathcal{Q}|)$. Updating incidence entries and the mapping per contracted pair is linear in local degree and folded into the local maintenance term.

Putting it together. Summing the components gives

$$T_{\text{update}} = O\left(\sum_{e \in \Delta E} |e|^2\right) + O\big(|\mathcal{Q}|\,(C \log C + W \cdot \bar{b})\big) + O(K \log |\mathcal{Q}|).$$

Since $|\mathcal{Q}| \le O(M^2)$, the cost is quadratic in dirty-region size up to local scoring factors. For bounded recent-window statistics and bounded hyperedge size, THGC scales with the affected region rather than with the full hypergraph.

I.4 EVENT QUEUE AND BATCHING OVERHEADS

Insertion and extraction from the TUE priority queue cost $O(\log n_{\text{evt}})$ per event using a min-heap (Cormen et al., 2009), which is negligible compared to $T_{\text{cand+score}}$ for nontrivial dirty regions. When processing minibatches of B events, structural and accumulator updates can be fused so that the structural term remains $O(\sum_{e \in \Delta E} |e|^2)$ over the batch, while candidate generation remains localized to the union of dirty regions, preserving the dirty-region-driven dominant term (Zhou et al., 2022).

J LIGHTWEIGHT CSI-COMPATIBLE STRUCTURAL VARIANTS

The suffix "CSI" denotes a lightweight CSI-compatible structural variant defined in this appendix. It is not an incremental execution of the full static algorithm, and it skips the original method's complete construction/refinement pipeline unless explicitly stated. These variants are used to isolate the effect of CSI-compatible local maintenance from the cost of rerunning full static coarseners.

J.1 HYPERSF-DERIVED STRUCTURAL CSI

Metric. Uses the global conductance of the common SHyPar mapping $M^\star : V \to \{1, \dots, k\}$ on snapshot $H = (V, E)$:

$$\phi(M^\star) = \frac{\sum_{e \in E} w(e)\,\mathbf{1}[e \text{ spans } \ge 2 \text{ clusters}]}{\min\big(\text{vol}(S), \text{vol}(V \setminus S)\big)}, \quad \text{vol}(X) = \sum_{e:\, e \cap X \neq \emptyset} w(e)\,|e \cap X|.$$

(Internal similarity scores from original HyperSF are not recomputed, only this evaluation metric.)

Simplification vs. HyperSF. Original HyperSF may (i) build multi-level representations, (ii) compute structural feature similarities (e.g., degree / co-incidence / edge-size features), (iii) perform iterative fusion of clusters. The HyperSF-derived structural CSI skips all construction and refinement; it *only* reuses $M^\star$ and performs a single pass to aggregate volumes and boundary counts. The pseudocode is given in Algorithm 2:

Algorithm 2 HyperSF-derived structural CSI evaluation

Require: Hypergraph $H = (V, E, w)$, common mapping $M^\star$
Initialize: $\texttt{cut} = 0$; for each cluster c, set $\texttt{vol}[c] = 0$
for each hyperedge e with weight $w(e)$ **do**
  Let $C_e = \{M^\star(v) \mid v \in e\}$
  for each $v \in e$ **do**
    $\texttt{vol}[M^\star(v)] \mathrel{+}= w(e)$
  end for
  if $|C_e| > 1$ **then**
    $\texttt{cut} \mathrel{+}= w(e)$
  end if
end for
Compute $\mathrm{vol}(S)$ and $\mathrm{vol}(V \setminus S)$ induced by best bipartition (routine reuses existing conductance helper).
Return $\phi(M^\star) = \texttt{cut} / \min(\mathrm{vol}(S), \mathrm{vol}(V \setminus S))$

J.2 HyperEF-derived structural CSI

Metric. Same global conductance $\phi(M^\star)$ (no separate energy minimization pass).

Simplification vs. HyperEF. Original HyperEF typically: (i) constructs flow / expansion-based objective, (ii) may perform iterative local improvements or multi-level contractions. The HyperEF-derived structural CSI discards flow construction and refinement; it directly tallies boundary hyperedges and volumes for $M^\star$. The pseudocode is given in Algorithm 3:

Algorithm 3 HyperEF-derived structural CSI evaluation

Require: $H = (V, E, w)$, mapping $M^\star$
Initialize cluster volume array and $\texttt{boundary} = 0$
for each hyperedge e **do**
  $S_e = \{M^\star(v) \mid v \in e\}$
  for each $v \in e$ **do**
    $\texttt{vol}[M^\star(v)] \mathrel{+}= w(e)$
  end for
  if $|S_e| > 1$ **then**
    $\texttt{boundary} \mathrel{+}= w(e)$
  end if
end for
Compute $\phi(M^\star)$ exactly as in the HyperSF-derived structural CSI (shared helper).
Return $\phi(M^\star)$

J.3 SHyPar-derived structural CSI

Metric. Same conductance $\phi(M^\star)$; $M^\star$ is the SHyPar partition closest to target ratio r (minimizing $|k - r|V||$ over available part counts).

Simplification vs. SHyPar. Original SHyPar may include: (i) multi-level coarsening, (ii) repeated V-cycle / FM-like refinements. The SHyPar-derived structural CSI omits any re-application of refinement and only evaluates conductance. The pseudocode is given in Algorithm 4:

Algorithm 4 SHyPar-derived structural CSI evaluation

Require: $H = (V, E, w)$, candidate SHyPar partitions $\{\mathcal{P}_p\}$, target ratio r
1: $k_{\text{target}} \leftarrow r|V|$
2: Select $M^\star$ from $\{\mathcal{P}_p\}$ minimizing $|p - k_{\text{target}}|$
3: Evaluate $\phi(M^\star)$ using the same counting procedure (reuse HyperSF-derived structural CSI steps 1–7)
4: Return $M^\star$, $\phi(M^\star)$

K TAPS-LITE STRUCTURAL BASELINE

In our implementation, **full TAPS** is the PPR-plus-conductance method described in the main paper (Andersen et al., 2006; Zhang et al., 2016; Zhou et al., 2007), while **TAPS-Lite** is a stripped structural baseline for fast comparison. The goal is to provide a lightweight, deterministic score for selecting and ranking coarse node representatives without running the full PPR / diffusion refinements.

Retained Core Signal. We keep only a static structural score per node, combining degree-like and participation statistics:

$$s(v) \;=\; \alpha\, \deg(v) \;+\; (1-\alpha)\, \frac{1}{|E(v)|} \sum_{e \ni v} \frac{1}{|e|},$$

where $\deg(v)$ is incident hyperedge count and the second term rewards participation in smaller hyperedges (default $\alpha = 0.7$).

Coarse Representative Selection. Given target ratio r, we keep the top $k = \lfloor r|V| \rfloor$ nodes by $s(v)$ as coarse representatives. Every non-representative node attaches to the representative with which it shares the most hyperedges; ties broken by highest $s(\cdot)$, then by node ID.

Mapping Output. Produces a node $\to$ cluster ID map: (1) representatives form initial singleton clusters in descending s order; (2) non-representatives inherit their representative's ID; (3) isolated nodes become singletons if needed to reach k, otherwise attach to the highest-score representative.

What is removed vs. full TAPS: no iterative PPR/diffusion relaxation; no multi-round seed expansion; no locality-aware re-scoring; purely deterministic ordering.

Complexity. Single pass over hyperedges for degree counts, inverse-size sums, and co-occurrence counts, giving $O\big(\sum_{e \in E} |e| + n \log n\big)$.

The pseudocode is given in Algorithm 5:

L COMPARISON OF PERFORMANCE AMONG VARIANTS OF THGC

This appendix reports conductance comparisons among THGC variants and CSI-enhanced baselines. These figures should be read together with the main structural and downstream metrics: conductance is useful for diagnosing cut quality, but it is not by itself a complete validation of temporal coarsening. The plots below show cases where TAPS gives favourable conductance while preserving the efficiency advantages of local repair.

M EXPERIMENTAL REPRODUCIBILITY

Code & Artifact. Public repo: `https://github.com/HyperCoarse/THGC`.

Environment. Python 3.8+ (tested to 3.13). Recommended isolated install:

```
python3 -m venv .venv
. .venv/bin/activate
pip install --upgrade pip
pip install .
```

(Optional: some legacy baselines require Julia 1.6+; not needed for core results.)

Data. Experiments use temporal hypergraph snapshots in `.hgr`. If snapshots are not provided, generate from raw:

Algorithm 5 TAPS-Lite structural baseline

Require: Hypergraph $H = (V, E)$, target ratio r, mixing parameter α
For each v: $\texttt{deg}[v] \leftarrow 0$, $\texttt{smallSum}[v] \leftarrow 0$
For each unordered node pair co-occurrence counter: empty hash (sparse)
for each hyperedge e **do**
  $w \leftarrow 1/|e|$
  for each $v \in e$ **do**
    $\texttt{deg}[v]{+}{=}1$; $\texttt{smallSum}[v]{+}{=}w$
  end for
  for each unordered pair $(u, v) \subset e$ **do**
    $\texttt{shared}[u, v]{+}{=}1$
  end for
end for
for each $v \in V$ **do**
  $s(v) \leftarrow \alpha\,\texttt{deg}[v] + (1 - \alpha)\,\texttt{smallSum}[v]/\texttt{deg}[v]$ if $\texttt{deg}[v] > 0$, else 0
end for
$k \leftarrow \lfloor r|V| \rfloor$; $\texttt{reps} \leftarrow$ top k nodes by $s(v)$ (ties: smaller ID)
Assign cluster IDs to reps in descending s
for each non-rep node u **do**
  Find rep $r^\star$ maximizing $\texttt{shared}[u, r]$ (tie: higher $s(r)$ then smaller ID)
  $M[u] \leftarrow M[r^\star]$
end for
Handle isolated nodes: attach to best rep or remain singleton if required to reach k
return mapping M

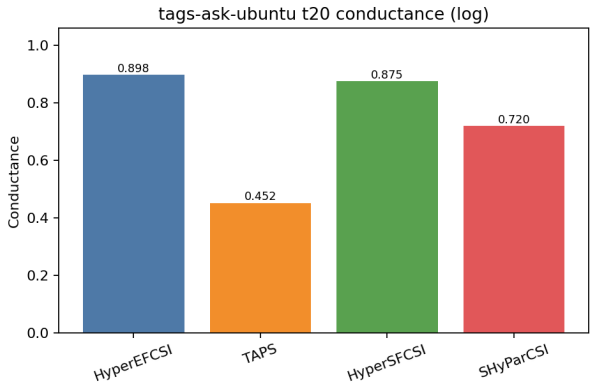


Figure 18: tags-ask-ubuntu (t20): Conductance across THGC variants and baselines on log-scale. Lower is better; bars are annotated with exact values as read from logs. The plot shows TAPS achieving substantially lower conductance than HyperEFCSI and HyperSFCSI, with SHyParCSI higher than TAPS but below HyperEFCSI on this dataset.

```
python Data/processing.py
```

Determinism. Fix seeds and hash for repeatability:

```
export PYTHONHASHSEED=0
# In Python:
import random, numpy as np
random.seed(0); np.random.seed(0)
```

Quick run. End-to-end demo (prints metrics and writes results):

```
python demo_thgc.py --dataset contact-high-school --runs 3 \
  --output Results/THGC/demo
```

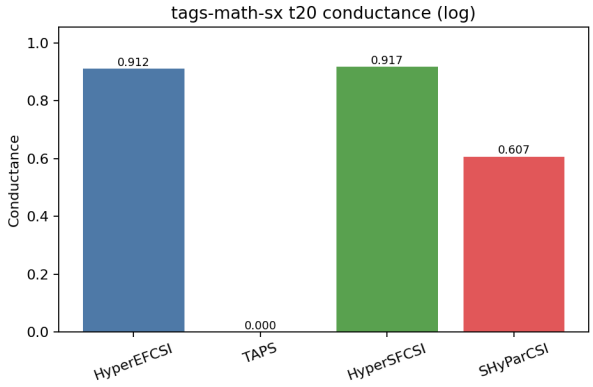


Figure 19: tags-math-sx (t20): Conductance on log-scale. TAPS attains the lowest conductance among compared methods, while HyperEFCSI and HyperSFCSI are notably higher; SHyParCSI is intermediate between the CSI baselines and TAPS. Lower is better.

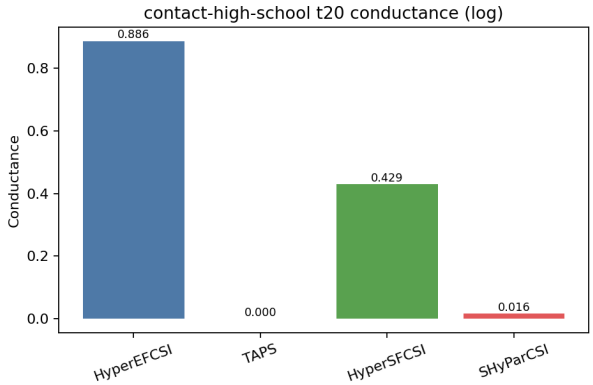


Figure 20: contact-high-school (t20): Conductance on log-scale. TAPS attains the smallest conductance by a wide margin; HyperSFCSI is the next best, with HyperEFCSI much larger. Lower is better; value labels are shown above each bar.

M.1 Scripts (concise usage map)

- `scripts/benchmark_taps_static.py`: One-shot (static) TAPS benchmarks across datasets/ratios.
- `scripts/benchmark_shypar_static.py`: Static SHyPar baselines (when available).
- `scripts/run_taps_full_temporal.py`: Full temporal TAPS over snapshot sequences (timing/quality logs).
- `scripts/run_taps_advanced_all.py`: full TAPS (PPR) hyperparameter sweeps; best-config CSV.
- `scripts/tune_taps_static.py`: Grid/random search for TAPS static parameters.
- `scripts/compare_taps_variants.py`: Compare TAPS variants on identical snapshots (summary CSV).
- `scripts/check_monotonic_conductance.py`: Sanity-check conductance monotonicity.
- `scripts/check_baselines_and_conductance.py`: Validate baseline outputs and conductance.

- `scripts/check_thgc_coarsening_ratios.py`: Verify requested vs. achieved coarsening ratios.
- `scripts/check_thgc_mapping_uniqueness.py`: Ensure one-to-one mappings (no clashes).
- `scripts/compute_old_methods_conductance.py`: Recompute/normalize legacy baseline conductance.
- `scripts/compute_one_old_conductance.py`: Single-case conductance recomputation.
- `scripts/run_old_then_thgc.py`: Run a baseline, then THGC for end to end comparison.
- `scripts/run_massive_experiments.py`: Batch runs over datasets/ratios/method grids; resume/overwrite.
- `scripts/plot_results_for_paper.py`: Regenerate paper figures/tables from `Results/` CSVs.

Typical invocations.

```
# Static TAPS benchmark
python scripts/benchmark_taps_static.py --datasets DAWN --ratios 0.6 \
  --out Results/TAPS_static

# Full temporal TAPS run
python scripts/run_taps_full_temporal.py --dataset contact-high-school \
  --out Results/TAPS_temporal

# Paper plots from aggregated CSVs
python scripts/plot_results_for_paper.py --in Results --out paper/plots
```

Outputs. Scripts write CSV/JSON and artifacts under `Results/` with per-dataset/method subfolders.

N Discussion of Constants and Regimes

The factor $\frac{\gamma}{1-\gamma}$ is the standard PageRank resolvent sensitivity constant, and the additional factor 2 in Theorem 1 comes from symmetrization $S_{\mathrm{ppr}}(u, v, t) = \pi_u[v] + \pi_v[u]$. The constant c_{op} relates restricted quadratic-form error to random-walk operator perturbation under standard normalizations, and can be bounded uniformly for a fixed normalization convention Loukas (2019). As $\alpha \uparrow 1$ the bound focuses on diffusion stability, and as $\alpha \downarrow 0$ it reduces to conductance variation, while larger decay in the temporal accumulator and smaller per-step updates tighten $\|P(t+1) - P(t)\|_1$.

O Elaborated Results

O.1 Average Update Time

We report mean per-snapshot update time (seconds) for each method and dataset under baseline forcing; lower is better. In all tables, **bold** indicates the best result in the row, and underline indicates the second-best result. Ties are shown with the same emphasis where applicable. Results of comparison of update times is reported in Tables 13, 14 and 15.

O.2 Average Conductance Comparison

We report average conductance (lower is better). Consistent with the timing tables, **bold** marks the best per-row value and underline the second-best. Results of comparison of conductance is reported in Tables 16, 17 and 18.

Table 13: Average Update Time (HyperEF Baseline Forcing)

DATASET	HYPEREF	HYPEREFCSI	TAPS
contact-high-school	5.211	0.06	**0.01**
NDC-classes-full	4.698	0.00	**0.00**
tags-ask-ubuntu	5.696	0.03	**0.02**
tags-math-sx	7.897	0.06	**0.01**

Table 14: Average Update Time (HyperSF Baseline Forcing)

DATASET	HYPERSF	HYPERSFCSI	TAPS
contact-high-school	9.639	0.01	**0.01**
NDC-classes-full	43.656	0.00	**0.00**
tags-ask-ubuntu	68.751	**0.03**	0.07
tags-math-sx	51.721	0.08	**0.07**

Table 16: Average Conductance (HyperEF Baseline Forcing)

DATASET	HYPEREFCSI	TAPS
contact-high-school	**0.89**	**0.89**
NDC-classes-full	**0.44**	0.46
tags-ask-ubuntu	**0.89**	**0.89**
tags-math-sx	**0.91**	**0.91**

Table 17: Average Conductance (HyperSF Baseline Forcing)

DATASET	HYPERSFCSI	TAPS
contact-high-school	0.429	**0.390**
NDC-classes-full	**0.124**	0.142
tags-ask-ubuntu	**0.88**	**0.88**
tags-math-sx	**0.92**	**0.92**

Table 18: Average Conductance (SHyPar Baseline Forcing)

DATASET	SHYPARCSI	TAPS
contact-high-school	0.02	**0.00**
NDC-classes-full	**0.00**	0.05
tags-ask-ubuntu	0.72	**0.69**
tags-math-sx	0.61	**0.26**

Result Summary. Across update-time tables, local THGC variants are substantially cheaper than repeated static recomputation. Conductance results are mixed across forcing choices, so we use them as structural diagnostics rather than as the sole basis for the claim. The central claim is supported by the combined evidence: local repair is fast, tracks static recomputation closely under sparse-to-moderate churn, exposes failure boundaries under bursts, and preserves downstream signal where such tasks are available.

Table 15: Average Update Time (SHyPar Baseline Forcing)

DATASET	SHYPAR	SHYPARCSI	TAPS
contact-high-school	21.301	≈ 0	≈ 0
NDC-classes-full	54.311	≈ 0	≈ 0
tags-ask-ubuntu	744.848	≈ 0	≈ 0
tags-math-sx	464.048	≈ 0	≈ 0