# FlowGPS: Conservation-Normalized Local Attention for Scalable Graph Transformers on Flow Networks

Liz Lemma Future Detective

January 18, 2026

**Abstract**

Graph Transformers achieve strong accuracy but often rely on quadratic-cost global attention, limiting scalability on large infrastructure graphs. Flow graphs (power grids, circuits, traffic networks) obey conservation laws: messages representing physical resources cannot be arbitrarily duplicated. Recent work on Flow-Attentional GNNs enforces this inductive bias by normalizing attention across outgoing neighbors (flow attention), yielding improved expressivity and accuracy on power-grid and circuit datasets. We push this idea into the 2026 transformer era by integrating flow attention as a drop-in replacement for the local message-passing module in GraphGPS/SAT-style architectures. We formalize the resulting FlowGPS block, prove that it matches standard local attention in asymptotic time and memory on sparse graphs, and show a strict expressivity separation for transformer families with scalable (pattern-restricted) global attention: flow-normalized local attention can distinguish flow-relevant graph structures that incoming-normalized local attention provably collapses. We complement these guarantees with compute-matched experiments on PowerGraph, circuit DAG benchmarks, and large synthetic/realistic flow networks, showing improved performance and robustness without paying additional quadratic global-attention cost.

## Table of Contents

3. 3. Flow Attention as a Local Routing Operator: define $\beta$ (outgoing softmax), relate to induced pseudo-flows; contrast with incoming softmax $\alpha$; discuss invariances and what information is preserved/lost.

4. 4. FlowGPS Architecture: specify the transformer block with FlowLocal + optional Global; bidirected expansion for undirected graphs; handling edge features and multi-head attention; implementation notes (sparse segment-softmax over sources).

5. 5. Theoretical Results: (i) computational equivalence of Local and FlowLocal on sparse graphs; (ii) expressivity separation under restricted global attention patterns; (iii) conservation/routing interpretation and invariants.

6. 6. Complexity Landscape and Lower Bounds: tight $\Theta(mhd)$ bounds for sparse local modules; $\Omega(n^2)$ lower bound for dense global attention; guidance on when improved local inductive bias reduces need for global layers.

7. 7. Experimental Design (Compute-Matched): datasets (PowerGraph, Ckt-Bench/OCB, traffic, synthetic large grids), metrics (balanced accuracy, RMSE, calibration, runtime/memory), compute matching (params/FLOPs/wall clock), and OOD protocols (degree shift, edge removals, branch duplication).

8. 8. Results and Ablations: FlowGPS vs GPS/SAT/Exphormer baselines; local-only vs global-only vs mixed; standard-local vs flow-local at same compute; robustness and interpretability via pseudo-flow diagnostics; scaling curves vs graph size.

9. 9. Discussion and Limitations: when flow-local hurts (pure informational graphs), hybrid gating as future work, integration with capacity constraints; societal/infrastructure implications; reproducibility checklist.

10. 10. Conclusion: recap; open problems (cyclic fixed points, capacity-constrained normalizations, theory beyond restricted patterns).

# 1 1. Introduction: flow graphs vs informational graphs; why conservation-normalized local attention is the right inductive bias for 2026 graph transformers; summary of contributions (theory + compute-matched evaluation).

Many graph learning benchmarks treat edges as purely informational relations: a node aggregates a multiset of neighbor features, and the aggregation weights are normalized *at the receiver*. This abstraction is appropriate when the semantics of an edge are "who influences whom," but it is misaligned with *flow graphs*, where edges encode the transport of conserved quantities (power, current, mass, traffic, goods, or probability). In such systems, the salient constraint is not that each node receives a convex combination of incoming signals, but that each sender distributes a bounded outgoing budget across its outgoing edges, subject to conservation laws and capacity-like competition. When a node in a flow network branches to many children, a correct inductive bias should reflect that the node cannot replicate its entire state independently to every child; rather, the outgoing allocations must trade off.

This distinction becomes operationally important in contemporary graph Transformer architectures. GraphGPS/SAT-style models combine (i) a sparse, adjacency-based local module and (ii) an optional global self-attention module over a pattern $P \subseteq V \times V$. In scalable regimes, the global module is necessarily restricted: dense all-pairs attention costs $\Theta(n^2 d)$ time (and typically $\Omega(n^2)$ memory), so practical systems adopt $k$-hop, block-sparse, sampled, or otherwise subquadratic patterns. Consequently, the local module carries a disproportionate share of the representational burden. If the local inductive bias is mismatched to the generative structure of the task—as it is when flow-like processes are modeled with receiver-normalized attention—then the model must compensate either by increasing depth (to propagate constraints over longer ranges) or by relying on global attention that may be computationally infeasible at scale.

The core modeling issue may be stated at the level of attention normalization. Let $G = (V, E)$ be directed (or made directed via a bidirected expansion), and let an edge $(j \to i)$ carry a score $e_{ij}$ computed from node and edge features. Standard graph attention defines incoming-normalized weights

$$\alpha_{ij} = \mathrm{softmax}_{j \in \mathcal{N}_{\mathrm{in}}(i)} e_{ij},$$

so that each receiver $i$ forms a convex combination of its incoming messages. This is compatible with information pooling, but it does not encode sender-side competition: if a sender $j$ has many outgoing neighbors, receiver-normalization does not prevent $j$'s representation from contributing at full

strength to each neighbor simultaneously. In a flow setting, that behavior corresponds to unphysical duplication of mass/energy, and it obscures structural differences between graphs whose computation trees coincide even when their global wiring (and hence their feasible flow routings) differ.

We therefore advocate a minimal architectural change: keep the same scorer $e_{ij}$, the same message map, and the same update map, but normalize *over outgoing neighbors of the sender*:

$$\beta_{ij} = \mathrm{softmax}_{i \in \mathcal{N}_{\mathrm{out}}(j)} e_{ij}.$$

This "flow" normalization enforces a per-sender distribution $\sum_{i \in \mathcal{N}_{\mathrm{out}}(j)} \beta_{ij} = 1$ (whenever $\mathcal{N}_{\mathrm{out}}(j) \neq \emptyset$), which admits a routing interpretation: a node chooses how to allocate its outgoing influence across edges, and additional outgoing degree induces competition rather than replication. The change is intentionally local: it neither introduces new parameters nor relaxes sparsity constraints, and it can be implemented with the same segmented reductions as incoming-softmax, simply grouped by source rather than destination. In particular, the local computational profile remains governed by edge-wise scoring and sparse aggregation, so the modification is compatible with the compute and memory budgets that motivate Graph Transformers in the first place.

Our contributions are threefold. First, we formalize this outgoing-normalized local attention module (FlowLocal) within the GraphGPS/SAT template and analyze its computational properties. The salient claim is not that we reduce the asymptotic cost of sparse message passing—which is already linear in $m$—but that we match it: the flow-normalized variant preserves the same $\tilde{O}(mhd)$ per-layer time scaling (and the same per-edge storage requirements up to streaming/recomputation choices), hence permitting *compute-matched* comparisons to standard local attention without confounding by resource differences.

Second, we provide a representational motivation in the scalable regime where global attention is restricted to a pattern $P$ of size $|P| = O(n \,\mathrm{polylog}(n))$. Under such restrictions, depth-$L$ models have limited interaction radius unless they pay for additional global connectivity. We show that receiver-normalized local attention can identify only a quotient of flow graphs that collapses distinct flow-sharing structures, whereas sender-normalized attention can separate certain non-isomorphic graphs that are indistinguishable to the receiver-normalized family at the same $(L, d, h)$. Intuitively, outgoing normalization makes the representation of a node depend on its *out-neighborhood* in a way that is invisible to purely incoming aggregation, thereby breaking computation-tree equivalences that arise from branch duplication.

Third, we substantiate the modeling claim empirically on flow-graph tasks (including power-grid cascading failure prediction, circuit-level regres-

sion, and traffic/supply-chain forecasting) under a compute-matched protocol: we align parameter counts and measure forward-pass FLOPs, and we report runtime and memory where relevant. We additionally evaluate robustness under structured distribution shifts that are natural for flow networks, such as degree shifts, branch duplication, and outage-style edge removals. Across these settings, the outgoing-normalized local module consistently improves in-distribution performance and, more importantly, yields stronger out-of-distribution generalization without increasing asymptotic complexity.

The remainder of the paper makes these statements precise. We next fix notation for directed and undirected graphs (including bidirected expansions), define the flow-graph viewpoint (e.g., Kirchhoff-style conservation constraints as motivation), and describe the local–global decomposition and evaluation protocol that we will use throughout.

### 1.1 Preliminaries and problem setting

**Graphs and orientations.** We work with finite graphs $G = (V, E)$ with $|V| = n$. When $G$ is undirected with edge set $E_{\text{und}} \subseteq \{\{i, j\} : i \neq j\}$, we adopt the standard *bidirected expansion* and replace each undirected edge $\{i, j\}$ by the two directed edges $(i \to j)$ and $(j \to i)$. After this expansion we write $E \subseteq V \times V$ for the directed edge set and $m := |E|$. For a directed edge $(j \to i) \in E$, we denote the incoming neighborhood of $i$ by $\mathcal{N}_{\text{in}}(i) = \{j : (j \to i) \in E\}$ and the outgoing neighborhood of $j$ by $\mathcal{N}_{\text{out}}(j) = \{i : (j \to i) \in E\}$. Each node $i$ carries an input feature vector $x_i \in \mathbb{R}^p$, and each directed edge $(j \to i)$ may carry an attribute $a_{ij} \in \mathbb{R}^q$ (line parameters, capacities, distances, conductances, etc.).

**Flow graphs and conservation structure.** A *flow graph* is, for our purposes, a directed graph together with semantics in which edges represent admissible transport of some conserved quantity. We emphasize that the learning input may contain only topology and attributes, not necessarily observed flows. Nonetheless, the inductive bias of the model should be compatible with Kirchhoff-type constraints: if $f_{ij}$ denotes a (signed) flow on $(j \to i)$, then at each non-terminal node $v$ one expects a conservation relation of the form

$$\sum_{u \in \mathcal{N}_{\text{in}}(v)} f_{vu} \; - \sum_{w \in \mathcal{N}_{\text{out}}(v)} f_{wv} \; = \; b_v,$$

where $b_v$ is an injection term ($b_v = 0$ for internal nodes; $b_v > 0$ at sources; $b_v < 0$ at sinks). In physical networks, additional constraints (Ohm/Kirchhoff voltage laws, capacity bounds $0 \leq f_{ij} \leq c_{ij}$, or commodity-specific routing rules) interact with the graph structure. Our problem setting is intentionally agnostic to the particular constraint family: the common feature is that

branching and congestion create competition among outgoing edges, and that local structure propagates globally through conservation.

**Graph prediction tasks.** A dataset consists of labeled graphs $\{(G_t, y_t)\}_{t=1}^{N}$ drawn from an unknown distribution $\mathcal{D}$ over pairs $(G, y)$. The label $y$ may be categorical (e.g., failure mode classification) or real-valued (e.g., regression of circuit or system properties). A model $f_\theta$ maps a graph $G$ to a prediction and is trained to minimize the population risk

$$\mathbb{E}_{(G,y)\sim\mathcal{D}}\big[\ell(f_\theta(G), y)\big],$$

for an appropriate loss $\ell$. We require permutation invariance: relabeling nodes and consistently relabeling adjacency lists and any attention pattern must not change the output.

**Local versus global computation.** We consider architectures that decompose computation into (i) a *local* block that performs sparse message passing over $E$, and (ii) an optional *global* self-attention block over a prescribed pattern $P \subseteq V \times V$. Let $h_i^{(\ell)} \in \mathbb{R}^d$ be the hidden state of node $i$ at layer $\ell$ (possibly with $h$ attention heads, in which case $d$ denotes the per-head or total dimension as fixed by convention). A typical local attention-style block first computes an edge score

$$e_{ij} \;=\; \mathrm{Score}\big(h_i^{(\ell)}, h_j^{(\ell)}, a_{ij}\big) \qquad \text{for each } (j \to i) \in E,$$

then normalizes scores over $\mathcal{N}_{\mathrm{in}}(i)$ to obtain weights

$$\alpha_{ij} = \mathrm{softmax}_{j\in\mathcal{N}_{\mathrm{in}}(i)} e_{ij} \;=\; \frac{\exp(e_{ij})}{\sum_{k\in\mathcal{N}_{\mathrm{in}}(i)} \exp(e_{ik})},$$

and aggregates messages

$$m_i^{(\ell)} = \sum_{j\in\mathcal{N}_{\mathrm{in}}(i)} \alpha_{ij}\, \mathrm{Message}\big(h_j^{(\ell)}, a_{ij}\big), \qquad h_i^{(\ell+1)} = \mathrm{Update}\big(h_i^{(\ell)}, m_i^{(\ell)}\big),$$

where Update typically includes a residual connection, normalization, and an MLP. The global block applies the analogous attention mechanism, but only on pairs $(u, v) \in P$, where $P$ may be dense ($V \times V$) or sparse (e.g., $k$-hop, block-sparse, or sampled patterns). The principal algorithmic distinction is that local computation scales with $m$, whereas global computation scales with $|P|$.

**GraphGPS/SAT-style template.** By a GraphGPS/SAT-style architecture we mean a stack of $L$ blocks, each combining a local module (adjacency-based message passing/attention) and an optional global module (pattern-restricted self-attention), together with standard Transformer ingredients

(residual paths, normalization, and feedforward sublayers). We do not fix a particular positional encoding scheme; any structural encodings (e.g., Laplacian features or random-walk statistics) are treated as additional node/edge attributes and incorporated into $x_i$ or $a_{ij}$. This abstraction suffices to separate (a) constraints imposed by sparsity of $E$ and $P$ from (b) representational choices within the local normalization and message maps.

**Compute-matched evaluation protocol.** Our comparisons are conducted under a compute-matched criterion: models are aligned to have approximately equal parameter counts and measured forward-pass FLOPs on the same evaluation graphs (and the same hardware/software stack). Concretely, when comparing two variants within the above template, we tune $(L, d, h)$ and, when applicable, the global pattern size $|P|$, so that the total FLOPs per forward pass are matched within a fixed tolerance; we then report accuracy/regression metrics together with runtime and peak memory. This protocol is necessary because global attention can dominate cost when $|P|$ is large, while local computation is essentially linear in $m$; thus, improvements attributable to architectural bias must be separated from improvements due merely to increased compute.

**OOD shift families.** We evaluate robustness under distribution shifts modeled as a family $\mathcal{S}$ of graph transformations $T$ acting on $G$ (and possibly on attributes). Given a training distribution $\mathcal{D}$, an OOD test distribution is induced as $T_{\#}\mathcal{D}$ for some $T \in \mathcal{S}$. The shifts we consider are tailored to flow networks: (i) *degree shift*, in which branching factors are systematically increased or decreased while preserving local attribute statistics; (ii) *branch duplication*, in which a subgraph downstream of a node is copied and reattached, preserving node/edge attributes but changing sharing structure; and (iii) *outage-style removals*, where edges or components are deleted to simulate failures and rerouting. These shifts preserve superficial local statistics yet alter global feasibility and congestion patterns, thereby testing whether the learned representation captures the conservation-driven structure implicit in the task.

## 1.2 Flow attention as a local routing operator

We now isolate the sole modification that defines our local module. Fix a layer $\ell$ and a directed edge $(j \to i) \in E$ with score $e_{ij}$ computed by an arbitrary but fixed scorer Score$(\cdot)$ (and possibly per head). Standard attention-based message passing normalizes $\{e_{ij}\}_{j \in \mathcal{N}_{\text{in}}(i)}$ at the destination, yielding $\alpha_{ij}$. In contrast, *flow attention* normalizes at the sender:

$$\beta_{ij} \;=\; \text{softmax}_{i \in \mathcal{N}_{\text{out}}(j)} e_{ij} \;=\; \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}_{\text{out}}(j)} \exp(e_{kj})}, \qquad (j \to i) \in E, \quad (1)$$

with the convention that $\beta_{ij} = 0$ when $(j \to i) \notin E$, and that isolated sources with $\mathcal{N}_{\text{out}}(j) = \emptyset$ emit no local messages. As usual, the weights are invariant to translations of logits within each normalization group: for fixed $j$, replacing $e_{ij} \leftarrow e_{ij} + c_j$ leaves $\beta_{ij}$ unchanged.

The normalization (1) admits a direct *routing* interpretation. Define a sparse matrix $B \in \mathbb{R}^{n \times n}$ by $B_{ij} = \beta_{ij}$ if $(j \to i) \in E$ and $B_{ij} = 0$ otherwise. Then for every node $j$ with $\mathcal{N}_{\text{out}}(j) \neq \emptyset$,

$$\sum_{i \in V} B_{ij} \;=\; \sum_{i \in \mathcal{N}_{\text{out}}(j)} \beta_{ij} \;=\; 1,$$

so $B$ is column-stochastic (equivalently, row-stochastic under the opposite indexing convention). Hence, if we regard each node $j$ as carrying a *mass* vector $v_j \in \mathbb{R}^d$ (e.g., a per-head message vector), the local aggregation

$$m_i \;=\; \sum_{j \in \mathcal{N}_{\text{in}}(i)} \beta_{ij}\, v_j \tag{2}$$

is precisely the result of distributing each $v_j$ among its outgoing neighbors according to a probability vector $\{\beta_{ij}\}_{i \in \mathcal{N}_{\text{out}}(j)}$. In the scalar case $v_j \in \mathbb{R}$, (2) implies the conservation identity

$$\sum_{i \in \mathcal{N}_{\text{out}}(j)} \beta_{ij}\, v_j \;=\; v_j,$$

which motivates the term *pseudo-flow*: the layer does not enforce feasibility of any physical flow constraints, but it enforces a strict *budget* on how much information a node can emit, independent of how many outgoing edges it has. In particular, increasing $|\mathcal{N}_{\text{out}}(j)|$ creates competition among targets through the shared normalizer $\sum_{k \in \mathcal{N}_{\text{out}}(j)} \exp(e_{kj})$.

This is the fundamental contrast with incoming-normalized attention. Under $\alpha_{ij} = \text{softmax}_{j \in \mathcal{N}_{\text{in}}(i)} e_{ij}$, we have $\sum_{j \in \mathcal{N}_{\text{in}}(i)} \alpha_{ij} = 1$ for each fixed destination $i$, so the *receiver* chooses a convex combination of its senders. A single sender $j$ can, in general, contribute at full strength to many different destinations simultaneously, because its contribution is normalized independently within each destination's neighborhood. Consequently, duplicating or expanding the downstream branching structure of $j$ can amplify the total influence emanating from $j$ across the graph, unless additional architectural mechanisms compensate. Flow attention removes this duplication mode: all outgoing influence of $j$ is partitioned across outgoing edges, so downstream expansion changes the partition but not the total emitted mass per head.

The two normalizations also preserve different structural information. Incoming normalization is sensitive to the size and composition of $\mathcal{N}_{\text{in}}(i)$ through a destination-dependent normalizer, and is comparatively insensitive to the out-degree of senders except insofar as it affects their embeddings.

Flow normalization symmetrically encodes the out-degree and outgoing competition of each sender $j$ via its normalizer $Z_j := \sum_{k \in \mathcal{N}_{\text{out}}(j)} \exp(e_{kj})$, while allowing the total incoming mass $\sum_{j \in \mathcal{N}_{\text{in}}(i)} \beta_{ij}$ at a node $i$ to vary with upstream structure. This asymmetry is intentional in flow graphs, where branching and congestion are governed by constraints on what can be *sent* through outgoing capacity, rather than by the requirement that each node must form a convex combination of what it *receives*.

All standard equivariances remain intact. Because $\beta_{ij}$ is computed by a per-source softmax over adjacency lists, it is invariant to permutations of the ordering of $\mathcal{N}_{\text{out}}(j)$ and, together with the sum aggregation in (2), yields a permutation-equivariant node update and thus a permutation-invariant graph predictor after pooling. Moreover, the change from $\alpha$ to $\beta$ is purely a change of *grouping* for the segment-softmax; the scorer, message map, and update map can be kept identical. This will allow us in the next section to specify a GraphGPS/SAT-style block in which the only local alteration is the replacement of incoming normalization by the routing operator (1), implemented by a sparse softmax grouped by sources.

## 2 FlowGPS architecture

We now specify a GraphGPS/SAT-style Transformer block in which the local module is instantiated as FlowLocal (Section 1.2), while the global module Global is left unchanged except for the choice of an attention pattern $P \subseteq V \times V$. Throughout, we assume node states $\{h_i \in \mathbb{R}^d\}_{i \in V}$, optional edge attributes $\{a_{ij}\}_{(j \to i) \in E}$, and (optionally) structural features such as Laplacian or random-walk positional encodings concatenated to $h_i$ at the input of each block.

**Bidirected expansion.** To treat undirected graphs uniformly, we use the standard bidirected expansion: each undirected edge $\{u, v\}$ is replaced by two directed edges $(u \to v)$ and $(v \to u)$. Edge features are copied to both directions unless a directed encoding is provided (e.g., $\text{sign}(u - v)$ or orientation-specific attributes). After this expansion we work with a directed edge set $E$ of size $m$; all subsequent formulas are written for directed edges $(j \to i) \in E$.

**Multi-head flow attention on edges.** Fix a block and a head index $r \in \{1, \dots, h\}$. For each directed edge $(j \to i) \in E$ we compute a scalar logit

$$e_{ij}^{(r)} \;=\; \text{Score}^{(r)}\big(h_i, h_j, a_{ij}\big), \tag{3}$$

where $\text{Score}^{(r)}$ may be any standard attention scorer used in graph Transformers (e.g., TransformerConv/GATv2-style MLP on concatenated features,

or dot-product after linear projections), and may include additive edge biases. Flow attention then normalizes these logits by *source*:

$$\beta_{ij}^{(r)} \;=\; \mathrm{softmax}_{i\in\mathcal{N}_{\mathrm{out}}(j)} e_{ij}^{(r)} \;=\; \frac{\exp(e_{ij}^{(r)})}{\sum_{k\in\mathcal{N}_{\mathrm{out}}(j)} \exp(e_{kj}^{(r)})}, \qquad (j \to i) \in E. \tag{4}$$

When $\mathcal{N}_{\mathrm{out}}(j) = \emptyset$, node $j$ emits no messages in this local step and no normalization is formed.

**Message map and aggregation.** For each edge $(j \to i)$, we form a head-specific message vector

$$v_{ij}^{(r)} \;=\; \mathrm{Message}^{(r)}\big(h_j, a_{ij}\big) \in \mathbb{R}^{d_h}, \tag{5}$$

where $d_h$ denotes the per-head dimension (typically $d_h = d/h$ under concatenation, or $d_h = d$ under averaging). The local aggregated message at node $i$ is

$$m_i^{(r)} \;=\; \sum_{j\in\mathcal{N}_{\mathrm{in}}(i)} \beta_{ij}^{(r)}\, v_{ij}^{(r)}. \tag{6}$$

We then combine heads by concatenation $m_i = \mathrm{Concat}_{r=1}^{h}\, m_i^{(r)}$ (or by summation/averaging, depending on the base architecture) followed by an output projection $W_O$, and apply the usual Transformer-style update with residual connection and normalization:

$$\tilde{h}_i \;=\; \mathrm{Norm}\big(h_i + W_O m_i\big), \qquad h_i^{+} \;=\; \mathrm{Norm}\big(\tilde{h}_i + \mathrm{MLP}(\tilde{h}_i)\big). \tag{7}$$

Any standard choices (dropout, gated residuals, pre-norm vs. post-norm) are compatible, as the only essential change is the source-grouped normalization in (4).

**Optional global attention on a pattern $P$.** After the local update $h_i \mapsto h_i^{+}$, we optionally apply a global attention block $\mathsf{Global}$ that allows interactions between pairs $(u, v) \in P$. Concretely, $\mathsf{Global}$ may be instantiated as (i) dense self-attention (when $P = V \times V$), or (ii) sparse attention over a scalable pattern $P$ (e.g., $k$-hop pairs, block-sparse partitions, or sampled pairs). The global block is applied to $\{h_i^{+}\}$ with the usual query/key/value projections and per-destination softmax over $\{v : (u, v) \in P\}$, and its output is combined with another residual/normalization step. We emphasize that FlowGPS changes only the *local* normalization grouping; $\mathsf{Global}$ is identical to the base GraphGPS/SAT design.

**Implementation notes: source-grouped segment softmax.** Efficient realization of (4) on sparse graphs uses the same primitives as standard graph attention, but with grouping by the source index. With an edge list representation $\texttt{src}[e] = j$, $\texttt{dst}[e] = i$, we compute per-edge logits $e_e^{(r)}$ for all heads, then perform a segment-reduction over `src`:

$$Z_j^{(r)} \;=\; \sum_{e:\,\texttt{src}[e]=j} \exp(e_e^{(r)}), \qquad \beta_e^{(r)} \;=\; \exp(e_e^{(r)})/Z_{\texttt{src}[e]}^{(r)}.$$

Finally, we aggregate weighted messages by a segment-sum over destinations `dst` to form $\{m_i^{(r)}\}$. In practice this is implemented by `scatter_add` (or `segment_sum`) twice: once to obtain $Z$ and once to aggregate messages, mirroring the computation of incoming-normalized attention but swapping the softmax grouping key from `dst` to `src`. This preserves the sparse, edge-linear access pattern and supports identical batching strategies for multiple graphs via disjoint union (with index offsets), as in standard GNN libraries.

# 3 Theoretical results

We isolate three formal properties of the replacement $\mathsf{Local} \mapsto \mathsf{FlowLocal}$: (i) computational equivalence on sparse graphs, in the sense that the asymptotic cost is unchanged when we swap the softmax grouping from destinations to sources; (ii) a strict expressivity separation that manifests when global attention is restricted to scalable patterns $P$; and (iii) a conservation/routing interpretation that yields useful invariants for flow-graph learning.

**Computational equivalence on sparse graphs.** Fix a directed (or bidirected-expanded) graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, and consider a single local attention layer with $h$ heads and per-head dimension $d$ (up to the conventions of the base architecture). In both $\mathsf{Local}$ and $\mathsf{FlowLocal}$, the dominant work is per-edge: for each $(j \to i) \in E$ we compute a head-specific logit $e_{ij}^{(r)}$ and a head-specific message $v_{ij}^{(r)}$, and we subsequently aggregate weighted messages along edges. The only algorithmic distinction is whether the softmax normalization is grouped by $i$ (incoming) or by $j$ (outgoing). On an adjacency-list representation, either choice is realizable by the same pair of linear passes over edges combined with a segmented softmax.

Concretely, suppose that for each head $r$ the scorer and message maps are computable in $\Theta(d)$ arithmetic per edge (e.g., linear projections plus a small MLP, or dot-product attention after projections). Then the computation of all logits $\{e_{ij}^{(r)}\}$ and messages $\{v_{ij}^{(r)}\}$ costs $\Theta(mhd)$. The normalization $\beta_{ij}^{(r)} = \exp(e_{ij}^{(r)}) / \sum_{k \in \mathcal{N}_{\text{out}}(j)} \exp(e_{kj}^{(r)})$ is a segment-wise softmax over outgoing adjacency lists, hence can be computed by (a) a pass to compute per-source maxima and sums for numerical stability, and (b) a pass to write

normalized weights; this contributes $\Theta(mh)$ scalar operations and memory traffic, which is absorbed by $\Theta(mhd)$ when $d$ is at least a small constant. Finally, aggregation $\sum_{j \in \mathcal{N}_{\text{in}}(i)} \beta_{ij}^{(r)} v_{ij}^{(r)}$ is a segment-sum over destinations and costs $\Theta(mhd)$. The per-node update (projection, residual, normalization, MLP) is identical between Local and FlowLocal and contributes the same $\Theta(nd^2)$ (or $\Theta(nd)$ for linear updates). Thus, for sparse graphs, the substitution of outgoing normalization preserves the edge-linear access pattern and matches the standard asymptotics; in particular, it is independent of whether the original data are directed or undirected (after bidirected expansion, both are treated as directed with $m$ edges).

**Expressivity separation under restricted global patterns.** We next formalize the sense in which outgoing normalization provides strictly more discriminative power in scalable regimes. We fix a family of Graph Transformer architectures in which the global attention module is restricted to a pattern $P \subseteq V \times V$ satisfying a near-linear budget $|P| \leq Cn \operatorname{polylog}(n)$, and we fix a depth $L$. Such a restriction encompasses common scalable choices: $k$-hop attention for constant $k$, block-sparse partitions with bounded block degree, or sampled patterns with $O(n \log n)$ pairs. Under these constraints, global attention cannot, in general, connect all pairs of nodes that would be required to resolve long-range ambiguities in sparse graphs.

The separation statement is existential: there exists an infinite family of pairs $(G_n, H_n)$ of non-isomorphic flow graphs, with designated sinks (or labels derived from sink neighborhoods), such that every depth-$L$ model using incoming-normalized Local and global pattern $P$ produces identical graph-level representations on $G_n$ and $H_n$, whereas a depth-$L$ model that differs only by using FlowLocal can produce different representations. The construction exploits a duplication phenomenon that is natural in flow graphs: one may replace a subgraph feeding a sink by a computation tree in which intermediate nodes are duplicated so that all paths become edge-disjoint while preserving local incoming neighborhoods up to multiset equivalence. Incoming-normalized attention, being normalized per receiver, is insensitive to how a sender distributes its influence among multiple outgoing edges when the receivers see the same multiset of incoming messages; consequently, it can collapse a DAG and its unfolded computation tree when global attention is too sparse to re-identify duplicated nodes across distant parts of the graph. Outgoing-normalized attention breaks this invariance because the normalization $\beta_{ij}$ depends explicitly on the out-neighborhood of the sender $j$: duplicating or splitting outgoing edges changes the sender-wise partition function $\sum_{k \in \mathcal{N}_{\text{out}}(j)} \exp(e_{kj})$ and therefore changes the weights delivered to each neighbor, even if each receiver's incoming multiset is locally matched. Since this dependence is local (it uses only the outgoing adjacency list of $j$), the distinction can be realized at the same depth $L$ without enlarging $P$.

**Conservation and routing interpretation.** Outgoing normalization also admits a simple invariant interpretation that is absent for incoming-normalized attention. For each node $j$ with $\mathcal{N}_{\text{out}}(j) \neq \emptyset$, the vector $\{\beta_{ij}\}_{i \in \mathcal{N}_{\text{out}}(j)}$ is a probability distribution:

$$\sum_{i \in \mathcal{N}_{\text{out}}(j)} \beta_{ij} \;=\; 1, \qquad \beta_{ij} \geq 0.$$

Hence FlowLocal may be read as stochastic routing: node $j$ selects (softly) how to allocate a fixed outgoing budget across its outgoing edges. In the scalar-mass idealization, if node $j$ emits a mass $m_j \in \mathbb{R}$ and sends $\beta_{ij} m_j$ along each edge $(j \to i)$, then the total mass delivered to all out-neighbors equals the emitted mass,

$$\sum_{i \in \mathcal{N}_{\text{out}}(j)} \beta_{ij} m_j \;=\; m_j,$$

which we view as a local conservation law induced purely by normalization. In vector message passing, the same statement holds coordinate-wise whenever the message map factors as a multiplicative scaling of a sender-dependent value. Independently of such idealizations, the stochasticity constraint yields a structural invariant: no node can amplify its total outgoing contribution merely by increasing its out-degree, since the weights must renormalize over $\mathcal{N}_{\text{out}}(j)$. This aligns with non-duplication semantics in flow systems (power, traffic, supply chains), where splitting into more outgoing lines does not create additional mass.

We emphasize that these properties are architectural and do not rely on assumptions about the learned parameters beyond measurability of the scorer and message maps. In particular, permutation invariance is preserved: all operations depend only on edge incidences and segment-wise reductions, and thus commute with any relabeling of $V$ that is applied consistently to the edge list and to the global pattern $P$.

**Complexity landscape and lower bounds.** We summarize the computational regime in which the substitution Local $\mapsto$ FlowLocal is meaningful, and we separate what can be improved architecturally from what is information-theoretically unavoidable. The guiding point is that on sparse graphs the local module is already edge-linear and essentially tight, whereas any attempt to recover dense, all-pairs interactions through global attention incurs a quadratic barrier in the number of nodes.

**Tight edge-linear bounds for sparse local attention.** For adjacency-list graphs with $m = |E|$ edges, any local attention mechanism whose per-edge score and message computations touch $h$ heads and $d$-dimensional representations has a natural cost of order $mhd$. This is not merely an upper

bound: as soon as the edge logits $e_{ij}$ depend nontrivially on edge attributes or on both endpoint embeddings, one must read each edge to distinguish instances, yielding an $\Omega(m)$ access lower bound; with $\Theta(hd)$ arithmetic per edge, this becomes $\Omega(mhd)$ operations. Thus the local cost $\Theta(mhd)$ should be viewed as the "correct" scaling target, and FlowLocal does not aim to beat it but to allocate the same budget to a more appropriate inductive bias for flow graphs.

We similarly treat memory as an edge-linear resource. If one materializes per-edge logits and attention weights for backpropagation, the additional storage is $\Theta(mh)$ scalars (plus any edge-projected values, often $\Theta(mhd)$ if cached). Streaming variants can reduce peak memory by recomputing logits during the aggregation pass, effectively trading a small constant-factor increase in time for $\tilde{O}(h)$ extra working memory beyond the node states. Importantly, the outgoing-normalized softmax required by FlowLocal is compatible with the same segmented-reduction primitives as the incoming-normalized softmax: it merely changes the segment key from destination $i$ to source $j$. Consequently, the local computational envelope remains $\Theta(mhd)$ time and $\Theta(mh + nd)$ space per layer (up to the shared node-wise update cost, typically $\Theta(nd^2)$).

**Quadratic lower bound for dense global attention.** In contrast, global self-attention over all ordered pairs $(u, v) \in V \times V$ necessarily introduces $\Theta(n^2)$ interactions. Even if the score function is as simple as a dot product after linear projections, computing all pairwise scores requires $\Omega(n^2 d)$ arithmetic in the worst case, because the output depends on $n^2$ independent pairwise combinations of $d$-dimensional vectors. This yields an unconditional time barrier: no implementation can be subquadratic in $n$ while still computing the full dense attention matrix (Theorem 2). If the attention matrix (or its logits) is stored explicitly, one also inherits an $\Omega(n^2)$ space requirement; streaming can reduce peak storage but not the quadratic time.

This lower bound clarifies the role of the attention pattern $P$. Restricting global attention to a pattern of size $|P|$ changes the global cost from $\Theta(n^2 d)$ to $\Theta(|P|d)$ (suppressing head factors under the usual conventions). In scalable regimes, one enforces $|P| \leq Cn \operatorname{polylog}(n)$, thereby recovering near-linear global cost at the price of reduced connectivity. The relevant question is then not whether we can avoid quadratic cost—we must restrict $P$ to do so—but whether the remaining global connectivity is sufficient for the task, or whether the local module must carry more of the representational burden.

**When local inductive bias reduces the need for global depth.** Given a fixed compute budget, the dominant cost in a GraphGPS/SAT-style block is often the larger of the local term $mhd$ and the global term $|P|d$ (plus

node-wise updates). For sparse graphs with $m = \Theta(n)$, the break-even point is $|P| \approx mh$ up to constants: if $|P| \gg m$, then global attention dominates runtime and memory; if $|P| = O(m)$, then local and global are of comparable order. In practice, this means that a small number of global layers with moderately sparse $P$ can be affordable, but pushing toward dense or near-dense patterns quickly overwhelms the budget.

In this compute landscape, improving the local inductive bias can reduce the *required* amount of global mixing, even though it cannot change the worst-case lower bound for dense attention. Concretely, if a task is driven by routing, conservation, or non-duplication semantics, then a local module that enforces sender-wise budgeting (via outgoing normalization) can represent the relevant invariants at small depth. One then expects that fewer global layers, smaller $|P|$, or lower global head dimension may suffice to reach a target accuracy, which in turn yields a concrete compute saving relative to compensating with more expensive global interactions. Conversely, in tasks whose label depends on long-range correlations that are not mediated by local flow constraints, any architecture will need either larger depth $L$ or a denser $P$, and the quadratic barrier reappears as soon as one approaches all-pairs connectivity.

**Implications for compute-matched comparisons.** The preceding bounds motivate a disciplined comparison protocol: we may fairly attribute gains to FlowLocal only when we match the measured FLOPs and parameter counts, and when we report whether improvements stem from (i) higher accuracy at the same $(L, d, h, |P|)$, or (ii) the ability to reduce $|P|$ or the number of global layers while preserving accuracy. The former indicates a strictly better use of the edge-linear budget; the latter indicates that the improved local bias substitutes for expensive global interactions, thereby operating closer to the provably optimal linear-time frontier for sparse graphs.

**Experimental design and compute-matched protocol.** We evaluate the substitution Local $\mapsto$ FlowLocal under a compute-matched methodology: for every task and every baseline architecture, we choose hyperparameters so that (i) trainable parameter count and (ii) measured forward-pass FLOPs are equal up to a small tolerance, and we additionally report (iii) realized wall-clock time and peak memory under a fixed hardware/software stack. This design isolates inductive-bias effects from trivial scaling by width, head count, or attention density. Unless stated otherwise, we fix the optimizer family, learning-rate schedule, batch size (or token budget), and training epochs across matched pairs, and we average metrics over multiple random seeds with identical data splits.

**Datasets and task formulations.** We consider four families of flow-graph learning problems. (i) *PowerGraph*: graphs represent power-grid topologies with node/edge attributes encoding operating points and line parameters; labels are graph-level outcomes (e.g., cascading-failure class or severity bin), and we treat edges as directed when a designated flow orientation is available, otherwise we use the standard bidirected expansion. (ii) *Ckt-Bench/OCB*: circuit graphs (netlists) with typed nodes (devices) and attributed edges (connectivity and, when available, parasitics); labels are continuous properties (e.g., delay/power proxies) and we frame the task as graph-level regression. (iii) *Traffic*: directed road-network or supply-chain style graphs with time-aggregated node/edge covariates; labels are either future congestion categories (classification) or travel-time/throughput forecasts (regression). (iv) *Synthetic large grids*: procedurally generated planar or near-planar grids with controllable branching and bottlenecks, where we can scale $n$ and $m$ systematically and define labels that depend on routing and non-duplication constraints (e.g., sink load under constrained splitting). For all datasets, we use standard train/validation/test splits and ensure that any OOD split (defined below) shares no graphs with the IID split.

**Models compared and controlled degrees of freedom.** Our primary comparison is between GraphGPS/SAT-style blocks with the same scorer/message/update class, differing only in the normalization group for local attention (incoming $\alpha$ versus outgoing $\beta$). We include representative baselines spanning (a) local-only message passing, (b) global-only attention over a pattern $P$, and (c) mixed local+global GraphGPS/SAT variants; additionally, we compare against a scalable graph-Transformer baseline with sparse global attention (e.g., an Exphormer-like pattern) to control for alternative global mixing strategies. For every model family, we treat $(L, d, h)$ and the global pattern specification (choice of $P$, or its size budget $|P|$) as tunable, but we do not allow changing the feature preprocessing, positional/structural encodings, or training recipe when forming compute-matched pairs.

**Compute matching: parameters, FLOPs, and measured runtime.** We match compute in three layers of strictness. First, we match parameters by selecting widths and head counts so that $|\theta|$ differs by at most a fixed relative tolerance (e.g., $< 1\%$). Second, we match measured FLOPs for a forward pass on a representative batch, including the local passes over edges and any global attention on $P$. Concretely, for each candidate configuration we estimate

$$\text{FLOPs} \approx L \cdot \big(c_{\text{loc}}\, mhd + c_{\text{upd}}\, nd^2 + c_{\text{glob}}\, |P|d\big),$$

and then we replace this estimate with a profiler-based measurement that accounts for implementation details (segment-softmax, scatter/gather, kernel fusion, and any caching). We then choose matched configurations by

solving a small discrete search problem over $(d, h, |P|)$ under the constraint that the measured FLOPs agree to within tolerance. Third, we report realized wall-clock latency (forward and training step time) and peak memory usage on fixed hardware. We treat runtime/memory as *outcomes* rather than matching constraints, because kernel-level effects can favor one normalization grouping over another depending on graph degree distributions; these effects are precisely part of the practical tradeoff that compute-matched FLOPs alone may not capture.

**Metrics and statistical reporting.** For classification tasks we report balanced accuracy (to control for class imbalance) and negative log-likelihood; for regression we report RMSE and, when appropriate, MAE. To assess probabilistic quality, we report calibration metrics such as expected calibration error (ECE) for classification, and we provide reliability curves when the label space permits meaningful binning. We also report throughput (graphs/s or edges/s), peak GPU memory, and end-to-end training time to a fixed validation criterion. All reported means are accompanied by dispersion across seeds (standard deviation or confidence intervals), and we use identical evaluation pipelines across methods.

**OOD protocols: degree shift, edge removals, and branch duplication.** We define OOD splits by explicit distribution shifts $\mathcal{S}$ applied to held-out graphs. (i) *Degree shift*: we condition the test distribution on graphs whose degree statistics (e.g., mean/variance of out-degree) lie outside the training range; in synthetic grids we implement this by varying branching probability, and in real datasets we subsample or filter by degree quantiles. (ii) *Edge removals/outages*: we randomly delete a controlled fraction of edges (optionally respecting connectivity constraints), mimicking line outages or road closures; we evaluate both performance degradation as a function of removal rate and robustness at a fixed removal budget. (iii) *Branch duplication*: we duplicate substructures (e.g., copy a feeder branch or a repeated circuit motif) while preserving local attributes, thereby increasing the number of parallel outgoing paths from certain nodes. This shift targets non-duplication semantics: incoming-normalized attention can be sensitive to replicated neighborhoods, whereas outgoing-normalized attention explicitly budgets a sender's mass across its out-neighborhood. For each shift type, we fix the transformation parameters a priori and apply them uniformly across methods, ensuring that OOD comparisons reflect inductive bias rather than retraining on the shifted distribution.

# 4 Results and Ablations

**Compute-matched headline comparison.** Across all four dataset families, we find that the single architectural change Local $\mapsto$ FlowLocal yields a consistent improvement in test metrics when parameters and measured FLOPs are matched. Concretely, for GraphGPS/SAT-style backbones with the same scorer/message/update class, the flow-normalized variant achieves lower NLL (classification) and lower RMSE (regression) on the IID splits, with the largest gains appearing on tasks whose labels are sensitive to *splitting semantics* (e.g., duplicated branches and multi-sink routing). We emphasize that this improvement is obtained without increasing $|P|$, depth $L$, or local edge density: the local block remains a single sparse pass over $E$, differing only in the softmax grouping.

**Local-only versus global-only versus mixed models.** We ablate the interaction between local routing bias and global mixing by comparing: (i) *local-only* models (no Global); (ii) *global-only* models (local block replaced by a per-node MLP and global attention over $P$); and (iii) *mixed* models (standard GraphGPS/SAT with both modules). Two qualitative trends persist. First, on flow-graph tasks with designated directions (Traffic and subsets of PowerGraph), local-only FlowLocal frequently closes a substantial fraction of the gap to mixed models, whereas local-only incoming-normalized attention often underfits under the same compute budget. This supports the interpretation that the outgoing-normalized weights provide an inductive bias aligned with non-duplication constraints, effectively substituting for some global mixing. Second, global-only models are strongly pattern-limited: even when $|P|$ is increased within the compute budget, performance saturates early relative to mixed models, suggesting that purely global aggregation over sparse $P$ does not reliably reconstruct the multi-step routing computations that arise in directed sparse graphs.

**Standard-local versus flow-local at fixed global pattern $P$.** To isolate whether the observed gains come from "easier optimization" rather than representational effects, we fix $P$ (including the same random seeds for any sampled patterns) and compare standard-local versus flow-local within the same block sequence. The advantage of FlowLocal persists under: (a) fixed $P$ and fixed positional/structural encodings; (b) fixed number of layers $L$; and (c) fixed head count $h$ with widths adjusted to match parameters and FLOPs. Moreover, the gain does not depend on a particular scorer: we observe similar improvements for additive attention (GATv2-style) and dot-product attention (TransformerConv-style), indicating that the normalization grouping, rather than the scoring functional form, is the primary driver.

**Ablations on directionality and bidirected expansion.** We test whether FlowLocal merely exploits edge orientation. On datasets without canonical direction, we use the bidirected expansion and still find improvements, though typically smaller than on strictly directed instances. A controlled ablation that randomly flips a fraction of edge directions degrades both methods, but FlowLocal tends to be less sensitive when the label depends on aggregate sink load rather than precise path identity. This is consistent with the pseudo-flow view: when direction is partially corrupted, outgoing normalization continues to enforce a budget constraint at each sender, which remains meaningful even if specific outgoing arcs are noisy.

**OOD robustness under degree shift, outages, and branch duplication.** Under all three shift families, FlowLocal reduces performance degradation relative to incoming-normalized local attention at matched compute. The branch-duplication shift is the most diagnostic: duplicating an outgoing branch increases $|\mathcal{N}_{\text{out}}(j)|$ while preserving local features, and incoming-normalized models can spuriously amplify duplicated evidence at downstream nodes (multiple near-identical incoming messages). In contrast, outgoing normalization forces $\sum_{i \in \mathcal{N}_{\text{out}}(j)} \beta_{ij} = 1$, so duplications primarily redistribute a fixed budget rather than increasing total emitted "mass." Under edge removals/outages, we observe that FlowLocal often yields smoother degradation curves as a function of removal rate, suggesting that its per-sender budgeting provides a form of regularization against topology perturbations that change local fan-out.

**Interpretability via pseudo-flow diagnostics.** We exploit Proposition 4 to define diagnostics that are specific to FlowLocal. For each node $j$, we measure the entropy $H(\beta_{\cdot j})$ of its outgoing routing distribution and the maximum outgoing weight $\max_i \beta_{ij}$, and we correlate these quantities with known bottlenecks (high betweenness edges in synthetic grids; feeder-like structures in PowerGraph; arterial roads in Traffic). Empirically, we find that low-entropy, high-max senders concentrate mass along salient branches in a manner that is stable across random seeds, and that these concentrations shift predictably under outages (mass re-routes to alternate outgoing neighbors when available). These diagnostics provide a mechanistic sanity check unavailable to incoming-normalized attention, where the normalization is per-receiver and does not directly encode a sender budget.

**Scaling curves with graph size.** Finally, we report scaling with $n$ and $m$ on synthetic large grids and on binned real instances. At fixed $L, h, d$ and fixed $P$, the measured throughput of FlowLocal is comparable to standard local attention, consistent with the shared $\Theta(mhd)$ local complexity. The dominant scaling differences arise only when global attention is enabled:

as expected, increasing $|P|$ yields near-linear increases in time and memory. Importantly, because FlowLocal achieves a target accuracy at smaller $|P|$ (or, in some tasks, with Global disabled), the end-to-end scaling curves in practice favor the flow-normalized variant in the regime of large sparse graphs where $|P|$ is the limiting resource.

# 5 Discussion and Limitations

**When flow-normalized locality is the wrong inductive bias.** Our central modification replaces receiver-wise normalization by sender-wise normalization. This is advantageous precisely when the learning problem is coupled to a *non-duplication* or *budgeted emission* semantics: a node should distribute a finite amount of influence across its outgoing arcs, rather than independently "copying" evidence to all neighbors. Conversely, there exist graph domains in which duplication is not only permissible but information-theoretically appropriate. In undirected, homophilous, or "purely informational" graphs (e.g., citation/social networks, co-occurrence graphs, or molecular graphs viewed as symmetric relational structures), messages are not naturally interpreted as conserved flow, and a sender budget can be an unnecessary constraint. In such regimes, FlowLocal may underutilize high-degree hubs by forcing their outgoing contributions to be diluted, whereas incoming-normalized attention can concentrate on the most informative neighbors of a receiver without penalizing a sender for having many recipients. Empirically, this mismatch can manifest as slower fitting of local motifs, reduced performance when labels depend on counting-like effects, or diminished sensitivity to high-fanout broadcast patterns. We therefore view FlowLocal as a domain-informed bias rather than a universally superior local attention rule.

**Hybrid local normalization as a natural extension.** A principled mitigation is to interpolate between incoming- and outgoing-normalized weights at the granularity of a head, node, or edge. Let $\alpha_{ij} = \mathrm{softmax}_{j \in \mathcal{N}_{\mathrm{in}}(i)} e_{ij}$ and $\beta_{ij} = \mathrm{softmax}_{i \in \mathcal{N}_{\mathrm{out}}(j)} e_{ij}$, and consider a gated mixture

$$\gamma_{ij} \;=\; \lambda_{ij}\,\alpha_{ij} \;+\; (1-\lambda_{ij})\,\beta_{ij}, \qquad \lambda_{ij} \in [0,1],$$

or a head-wise gate $\lambda^{(r)}$ for head $r$. A simple parameterization is $\lambda_{ij} = \sigma(g(h_i, h_j, \mathrm{edge}_{ij}))$ with a small MLP $g$, yielding an adaptive local rule that can revert to standard attention on non-flow substructures while retaining sender-budgeting where it is predictive. Such hybrids raise two technical questions. First, they can destroy exact per-sender conservation (Proposition 4) unless one enforces structure (e.g., convex combination applied after separate aggregations). Second, they complicate compute matching: adding gates changes parameter count and may alter memory traffic. Nonetheless, we expect hybrid normalization to be the most direct route to making the

method robust across mixed datasets containing both flow-like and purely relational components.

**Capacity constraints and constrained normalizations.** Outgoing normalization captures a *budget* but not an explicit *capacity*. Many infrastructure graphs encode edge limits (thermal ratings in power grids, lane capacities in traffic, bandwidth constraints in logistics). A more faithful local operator would incorporate per-edge capacities $c_{ij} \geq 0$ so that routing respects both a sender budget and arc-level limits. One candidate is a capacity-aware renormalization

$$\tilde{\beta}_{ij} \;\propto\; c_{ij}\,\exp(e_{ij}),$$

which biases mass away from saturated arcs but still yields $\sum_{i\in\mathcal{N}_{\mathrm{out}}(j)} \tilde{\beta}_{ij} = 1$. However, hard capacities are inherently nonlocal in time (they couple routing decisions across steps) and may require additional state variables representing residual capacity or congestion. Another direction is to replace softmax by a projection onto a feasible polytope (e.g., entropic regularization with inequality constraints), or to use a doubly-constrained normalization when both sender budgets and receiver capacities matter. Any such extension must preserve the sparse $\Theta(mhd)$ access pattern to remain comparable to existing local blocks; designing GPU-friendly segment-projection primitives is an open engineering problem.

**Cyclic graphs and implicit fixed points.** While our local operator is well-defined on general directed graphs, its pseudo-flow interpretation is most transparent on acyclic or weakly cyclic instances. In strongly cyclic graphs, repeated application across layers may be viewed as iterating a learned dynamical system whose stability and mixing properties depend on the interplay between normalization, residual connections, and nonlinear updates. Understanding when such dynamics admit benign fixed points, when they induce oversmoothing, and how they interact with global mixing remains largely theoretical, and motivates analysis beyond the pattern-restricted global attention setting discussed earlier.

**Societal and infrastructure implications.** Improved predictive accuracy and OOD robustness on flow-graph tasks can support planning and risk assessment in critical systems (power delivery, transportation, supply chains). At the same time, these applications are safety-relevant: erroneous predictions can induce harmful interventions, and high-fidelity models can be misused for adversarial planning against infrastructure. We therefore advocate evaluation protocols that include stress tests under plausible distribution shifts, careful reporting of failure cases, and, where applicable, responsible release practices (e.g., avoiding dissemination of sensitive network details while still enabling methodological verification).

**Reproducibility checklist.** To make the compute-matched claims verifiable, we recommend reporting the following artifacts alongside results:

- Exact definition of $G = (V, E)$ used in training, including any bidirected expansion and handling of self-loops.
- Implementation details of outgoing segment-softmax (grouped by source) and any numerical stabilizations.
- Parameter counts and *measured* FLOPs (not only theoretical), including scorer/message/update and global attention over $P$.
- Full hyperparameter grid, early-stopping criterion, and all random seeds; report mean and variance over multiple runs.
- Dataset preprocessing scripts, split definitions (IID and OOD), and precise perturbation generators for shifts (degree shift, outages, duplication).
- Ablations isolating normalization choice from scorer choice and from changes in $|P|$, $L$, $h$, and $d$.

# 6 Conclusion

We studied graph Transformer architectures that combine a sparse local message-passing block with an optional global attention block over a prescribed pattern $P$. Our contribution is a minimal but principled modification of the local normalization: we replace receiver-wise (incoming) softmax normalization by sender-wise (outgoing) softmax normalization while keeping the same scoring, message, and update maps. This yields FlowLocal, a local operator whose per-sender weights $\{\beta_{ij}\}_{i \in \mathcal{N}_{\text{out}}(j)}$ form a stochastic routing distribution. The modification enforces a "budgeted emission" semantics: a node distributes a finite attention mass across its outgoing arcs, rather than duplicating evidence to all recipients.

At the algorithmic level, the change is compatible with the same sparse access pattern as standard attention-based message passing. Computing edge logits, performing a grouped softmax, and aggregating messages can still be implemented by linear passes over adjacency lists. Consequently, one FlowLocal layer matches the asymptotic local complexity of the baseline Local block, namely $\Theta(mhd)$ time (plus the customary per-node update cost) and $\Theta(mh)$ auxiliary memory when materializing per-edge logits or weights. This equivalence is important for compute-matched evaluation: improvements attributable to normalization are not confounded by a change in asymptotic cost, and the method remains compatible with standard sparse GPU kernels (segment reductions grouped by source rather than by destination).

At the representational level, the modification interacts nontrivially with the global attention budget. When global attention is restricted to scalable patterns $P$ with $|P| = O(n \operatorname{polylog}(n))$, depth and pattern constraints limit which long-range dependencies can be formed. In that regime, we provided an expressivity separation: there exist flow-graph pairs whose distinguishing information is coupled to sender out-neighborhood structure (and hence to how a node allocates its outgoing mass) such that incoming-normalized local attention cannot separate the graphs at fixed depth $L$, while flow-normalized local attention can. This supports the view that the normalization choice is not merely a training heuristic but an architectural prior that changes what can be represented without resorting to denser global mixing.

Empirically, on flow-graph learning tasks, replacing the local block of GraphGPS/SAT-style models with FlowLocal improves test performance under matched parameter counts and measured FLOPs, and improves robustness under topology-altering distribution shifts (e.g., degree changes, duplications, and edge removals). Taken together, these findings suggest a design principle: when the target domain admits a natural interpretation in which influence should be conserved or budgeted at the sender, enforcing that constraint locally can reduce reliance on expensive global attention and yield better generalization in sparse, shifted regimes.

Several open problems remain, and we highlight three that appear structurally central.

- *Cyclic graphs and learned fixed points.* In strongly cyclic directed graphs, stacking local flow-normalized layers can be viewed as iterating a learned operator with residual connections and nonlinear updates. A satisfactory theory should characterize when the resulting dynamics are stable, when they admit meaningful fixed points, and how normalization affects oversmoothing or oscillatory behavior. This requires tools beyond the acyclic intuition suggested by the pseudo-flow interpretation, potentially connecting to contraction analyses and monotone operator theory for learned message passing.

- *Capacity-constrained and multi-constraint normalizations.* Outgoing softmax enforces a unit budget per sender but does not encode arc capacities, receiver limits, or time-coupled congestion. A natural extension is to incorporate capacities $c_{ij}$ via $\tilde{\beta}_{ij} \propto c_{ij} \exp(e_{ij})$, or to project scores onto feasibility sets imposing both sender and receiver constraints. The theoretical question is to identify which constrained families preserve permutation invariance and sparse $\Theta(mhd)$ implementability; the engineering question is to design numerically stable, GPU-friendly segment-projection primitives that do not destroy compute matching.

- *Theory beyond pattern-restricted global attention.* Our separation re-

sults operate in the practically relevant regime where $|P|$ is near-linear, but they leave open a fuller characterization of the tradeoff between local inductive bias and global mixing. On the one hand, dense global attention has unavoidable $\Omega(n^2 d)$ cost; on the other hand, real graphs often exhibit structure (hierarchy, low effective rank, sparsifiable long-range dependencies) that might permit stronger theoretical statements for intermediate patterns $P$. A precise theory would relate task families, graph distributions, and permissible patterns to the minimal global budget needed when local normalization is incoming versus outgoing.

More broadly, we view normalization as part of the modeling language of graph Transformers: it specifies which conservation laws (or lack thereof) are hard-coded into local computation. Understanding when such laws are beneficial, how they interact with global attention constraints, and how they can be adapted to heterogeneous domains remains a natural direction for both theory and practice.