ARITHMETIC GROWTH CONTROLS RECURRENCE COMPLEXITY IN UNIFORM GNNS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study how the arithmetic gate basis inside a shared recurrent GNN cell affects the recurrence cost of graph-level color refinement. One-round graph colors form $2^{\Theta(n)}$ classes, while every fixed round $k \geq 2$ refines $2^{\Theta(n^2)}$ classes. For a fixed rational ReLU-MLP (continuous piecewise-affine, CPWA), output-coordinate numerator and denominator bit lengths grow by only $O(\log n)$ per recurrence. This gives an $\Omega(n^2/\log n)$ lower bound for fixed $k \geq 2$, even with a recurrent global sum. More generally, consider a fixed-dimensional cell and readout with a polynomial-growth envelope, polynomial-magnitude initialization, and inverse-polynomial separation between distinct class outputs. For fixed $k \geq 2$, such a model needs $\Omega(\log n)$ recurrences. We meet this bound with one fixed rational arithmetic cell over affine maps, scalar ReLU, and exact binary multiplication gates. Given n and k, it computes graph-level CR_k in $2(k+1)\lceil \log_2(2n) \rceil + 4k + 4$ recurrences using only neighbor sums. Its integer outputs are unit-separated, and its states have $n^{O(k+1)}$ bits. For output radius $M(n)$ and separation $\delta(n)$, every fixed-dimensional representation still satisfies $\log(M(n)+1) + \log(1/\delta(n)) = \Omega(n^2)$. Thus adding exact multiplication reduces fixed-k recurrence complexity to $\Theta(\log n)$ without removing this output cost. The same cell computes stable color refinement in $O(n \log n)$ recurrences, while locality gives an $\Omega(n)$ lower bound. Exact circuit certificates and independent color-refinement implementations check the construction. The results do not address trainability or fixed-word execution.

1 INTRODUCTION

Message-passing graph neural networks are commonly compared through the Weisfeiler–Leman (WL), or color refinement, hierarchy (Xu et al., 2019; Morris et al., 2019; 2023). This familiar equivalence is usually nonuniform: finite-depth simulations may choose a different network for each maximum graph size. A recent uniform result removes this dependence: one recurrent Sum-GNN with rational parameters and a ReLU-MLP update can compute every color-refinement-invariant algorithm (Rosenbluth & Grohe, 2026). Recurrence therefore restores the missing expressivity, but the construction has a large polynomial overhead, and its tight complexity is open.

We ask how the arithmetic gate basis available in a shared cell affects its number of recurrences. This resource differs from the depth of a size-dependent combine network. Across the comparison, the state dimension, parameters, aggregation rule, and robust graph-output contract are fixed; the gate basis changes the induced state-growth rate. The resulting recurrence rates differ despite identical qualitative WL expressivity.

Class counts for the first two refinement rounds provide the lower bound. The first records the degree multiset. The second records, for every vertex, its degree and the multiset of its neighbors' degrees—the neighborhood degree list (NDL) studied in graph realization (Barrus & Donovan, 2018). We prove that their graph-level class counts jump from $2^{\Theta(n)}$ to $2^{\Theta(n^2)}$. The quadratic lower bound is constructive: two copies of a connected antiregular graph carry a dense regular bipartite adjacency matrix, and every second-round invariant has only a $2^{O(n)}$ fiber. Since later colors retain their second-round ancestors, the quadratic demand holds for every fixed $k \geq 2$.

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.

Table 1: Uniform recurrence bounds for graph-level color refinement. The fixed-k rows use unit or inverse-polynomial output separation. The logarithmic gaps in the two CPWA rows remain open.

Cell gate basis	Target	Recurrences
fixed CPWA (ReLU MLP)	CR_1	$\Omega(n/\log n)$ to $O(n)$
fixed CPWA (ReLU MLP)	fixed CR_k, $k \geq 2$	$\Omega(n^2/\log n)$ to $O_k(n^2)$
CPWA + exact multiplication	fixed CR_k, $k \geq 2$	$\Theta(\log n)$
same arithmetic cell	stable CR	$\Omega(n)$ to $O(n \log n)$

For a fixed rational CPWA cell realized by a ReLU MLP, denominators grow by a fixed factor per recurrence and aggregation grows magnitudes by at most $O(n)$. A T-round output therefore has only $O(T \log n)$ bits. The same range bound works for arbitrary real weights under inverse-polynomial output separation. This yields $\Omega(n^2/\log n)$ recurrences for fixed $k \geq 2$. A fixed cell with polynomial growth of degree greater than one can square its bit length, but packing still requires $\Omega(\log n)$ recurrences.

The entropy theorem also yields a lower bound on output precision. For output radius $M(n)$ and separation $\delta(n)$, any fixed-dimensional representation of graph-level CR_k classes, $k \geq 2$, satisfies $\log(M(n)+1) + \log(1/\delta(n)) = \Omega(n^2)$; a unit-separated integer output needs $\Omega(n^2)$ bits in some coordinate. Exact binary multiplication reduces the recurrences needed to produce the exact output but not this lower bound.

To attain the logarithmic bound, we use a local geometric power-sum code whose length doubles in one recurrence. After each refinement round, the packed neighbor sum uniquely determines the complete multiset of old node colors; a final identical block packs the graph-level color histogram. All clocks, phase selectors, and arithmetic are compiled into one 22-dimensional branch-free arithmetic cell with 124 affine, scalar ReLU, multiplication, input, and constant gates in total. The cell is independent of both n and k, receives them as ordinary input features, and never uses recurrent global sum.

Specifically:

- We prove the $2^{\Theta(n)}$-to-$2^{\Theta(n^2)}$ graph-color transition. It yields nearly quadratic uniform CPWA/ReLU-MLP lower bounds and an architecture-independent quadratic output-precision floor, including for arbitrary real parameters under a prescribed final-output margin. More generally, any fixed-width cell with a polynomial-growth envelope needs $\Omega(\log n)$ recurrences for fixed-k graph-level color refinement.
- We construct one deterministic multiplication-augmented arithmetic cell that computes every requested CR_k in $O((k+1)\log n)$ recurrences with unit output separation and $n^{O(k+1)}$-bit states. For fixed $k \geq 2$ this matches the lower bound; setting $k = n$ gives the stable-CR interval in Table 1.
- We provide a machine-readable fixed-cell DAG, an independent nested-descriptor CR implementation, and exact-integer trajectories that check the phase logic and cross-graph canonical invariant.

Within the exact uniform model and its fixed robust output contract, the arithmetic gate basis and its induced state-growth rate can trade against recurrence count.

2 Setting and closest prior work

Color refinement. All graphs are finite, simple, and initially unfeatured. Define cross-graph canonical color objects by

$$c_0^G(v) = \bot,$$
$$c_{t+1}^G(v) = \left(c_t^G(v), \left\{\!\left\{c_t^G(w) : w \in N_G(v)\right\}\!\right\}\right).$$

The graph color after t rounds is $\mathrm{CR}_t(G) = \left\{\!\left\{c_t^G(v) : v \in V(G)\right\}\!\right\}$. Using the recursive object, rather than graph-local integer labels, makes equality across graphs unambiguous. Stable color

refinement denotes the eventual graph-level equivalence relation. Our convention starts from c_0, so every round index counts applications of the displayed update.

Uniform recurrent Sum-GNNs. A model $\mathcal{N} = (A, F, M)$ has a constant state dimension d and rational parameters independent of graph size. On an unfeatured graph, the fixed initialization $A(n)$ is rational affine in the supplied integer $n = |V(G)|$; the upper construction initializes its designated n and q registers to n and every other unbounded register to zero. A global-sum model could instead compute n in one round. Its update is

$$h_v^{t+1} = F\left(h_v^t, \sum_{w \in N(v)} h_w^t, \sum_{w \in V(G)} h_w^t\right), \tag{1}$$

where F is a fixed rational ReLU MLP and $T : \mathbb{N} \to \mathbb{N}_0$. Here $\mathrm{ReLU}(x) = \max\{0, x\}$ is scalar, while the affine–ReLU composition is a continuous piecewise-affine (CPWA) map. We call this baseline a *fixed CPWA cell*. The last argument may be omitted. Node v may halt at an input-dependent $I_v \le T(n)$; completed states are held fixed. Neither I_v nor runtime is an output channel. The graph output is the fixed-dimensional exact rational vector obtained by applying a fixed rational ReLU MLP to the sum or average of the final node states. The model *refines* CR_ℓ if

$$\mathrm{CR}_\ell(G) \neq \mathrm{CR}_\ell(H) \quad \Longrightarrow \quad \mathcal{N}(G) \neq \mathcal{N}(H)$$

for all same-order graphs. For one uniform model $\mathcal{N}$, let $T_{\mathcal{N}}(n)$ be its worst-case recurrence count on n-vertex inputs. We write $\mathrm{RecComp}(\mathrm{CR}_\ell)$ for the infimum of these runtime functions over single models that refine CR_ℓ at every graph size, ordered by eventual asymptotic domination. Thus every lower bound below applies to each single uniform model, while every upper bound is achieved by one single uniform model; the notation never selects a different model separately at each n.

Robust refinement over reals. For $\delta : \mathbb{N} \to (0, 1]$, an otherwise identical model with arbitrary fixed finite real parameters *δ-robustly refines* CR_ℓ if

$$\mathrm{CR}_\ell(G) \neq \mathrm{CR}_\ell(H) \quad \Longrightarrow \quad \|\mathcal{N}(G) - \mathcal{N}(H)\|_\infty \ge \delta(n)$$

for all same-order graphs. The function δ is specified independently of the model outputs. This is a final-representation separation condition, not a claim about perturbations during execution. Define $\mathrm{RecComp}^{\mathbb{R}}_\delta(\mathrm{CR}_\ell)$ analogously over single arbitrary-real models satisfying this contract at every graph size.

Arithmetic gate bases and growth classes. For the gate comparison, replace the CPWA map F by any fixed arithmetic cell with a monotone envelope

$$\|F(z)\|_\infty \le A(1 + \|z\|_\infty)^q$$

for constants $A \ge 1$ and integer $q \ge 1$. Initial states have polynomial magnitude, width is fixed, and the graph readout is fixed-dimensional with inverse-polynomial separation. Fixed CPWA cells have $q = 1$. A *multiplication-augmented arithmetic cell* also admits exact binary gates $(x, y) \mapsto xy$ and has some constant $q > 1$. Multiplication is thus a gate, not an activation; scalar ReLUs remain in the CPWA control logic. We write $\mathrm{RecComp}_\times(\mathrm{CR}_k)$ for the augmented basis, requiring one fixed rational cell across all graph orders. Our upper arithmetic cell also takes the requested round count k as an integer input, so its architecture is fixed jointly in n and k.

Relation to prior work. Loukas (2020) proves that identifying general connected graphs needs quadratic message-passing communication capacity, covering depth, width, message size, global state, and adaptive architectures. Here the targets are the anonymous CR_1 and CR_2 partitions, and the resource is the recurrence count of one fixed rational cell; the paper does not claim the first information bottleneck for GNNs. Aamand et al. (2022) simulates WL with $\mathrm{polylog}(n)$ parameters and $O(\log n)$-bit messages, but its network depends on n and uses randomized weights with nonzero error. Barlag et al. (2024; 2026) relate GNNs to arithmetic circuits; we use this vocabulary for a task-specific recurrence separation and an explicit Sum-GNN cell. Products also appear in multiplicative or bilinear RNNs (Wu et al., 2016; Ebrahimi & Memisevic, 2025), GRU-based gated graph networks (Li et al., 2016), and learned arithmetic units (Trask et al., 2018; Madsen & Johansen,

2020). We do not claim multiplication as a new operation: our distinction is an exact product of potentially unbounded registers in a fixed rational circuit, used for a worst-case recurrence theorem rather than trainability. Maron et al. (2019) use matrix multiplication on second-order tensors to reach 3-WL expressivity; our scalar bilinear gate stays within the 1-WL message-passing boundary and changes recurrence complexity rather than distinguishing power. Ahvonen et al. (2024) characterizes recurrent GNN expressivity over reals and floats without quantitative recurrence bounds and leaves global readout outside its main characterization. Closest in resource focus, Cui et al. (2024) derive depth–bandwidth–precision tradeoffs for one node-level WL relabeling step under oblivious or instance-dependent parameter choices. Its network may be selected for a bounded graph family and order, and its resource is edge communication. Our model fixes one cell across all orders, targets graph-level CR_ℓ refinement, and measures recurrence under exact or final-output-margin contracts. The model quantifiers and cost measures therefore differ.

Our margin is a uniform separation requirement among distinct graph-level CR_ℓ classes, unlike graphon continuity, classification margins, or bit-flip robustness. Böker et al. (2023) metrizes finite-depth MPNN expressivity through graphons and continuous WL; Franks et al. (2024) asks when added WL expressivity improves generalization; and Kummer et al. (2025) studies loss of multiset expressivity under bit-flip attacks. Combined with a range bound for one shared recurrent cell, it yields a quantitative recurrence lower bound even when the fixed parameters are arbitrary reals.

The second-round object itself is classical. Barrus & Donovan (2018) studies neighborhood degree lists, including graphicality, switching, and unique labeled realizations. Degree-spectra matrices (Bassler et al., 2015) and degree-polynomial sequences (Jafarpour-Golzari, 2022) encode closely related local degree data. NDLs are therefore classical; here we count their *graph-level equivalence classes* over all graphs of order n and derive consequences for uniform recurrent computation.

All round counts in the following comparison use our convention that the initial constant coloring is CR_0. Most directly, Rosenbluth & Grohe (2026) proves uniform sufficiency with a generic polynomial compiler, from which an $O(n^{10}k^2)$ bound can be extracted, and explicitly asks for lower complexity and lower bounds. For graph-level color refinement, we show that changing the cell basis changes recurrence complexity by a polynomial factor. Rosenbluth (2026) bounds the outputs of a network whose entire depth is fixed before n varies and gives a $2^{\Theta(n)}$ lower-bound family for two-round refinement. That family matches the entropy order already attained by first-round degree multisets and therefore does not establish the transition proved here. Its architecture-level constant is also not a bound uniform in a depth $T(n)$. Our nested-denominator and magnitude lemmas hold simultaneously for every recurrence of one shared cell, yielding $2^{O(T(n)\log n)}$ outputs for input-dependent stopping, including mixed node stopping and global sum. Rybicki et al. (2026) independently uses two rounds of neighbor-degree aggregation followed by another refinement layer to identify almost every random graph in an anonymous-network model. Their colors are initialized by degree, so their second refinement corresponds to our CR_3, not our CR_2. That result concerns the next round and average random inputs; our second-round entropy theorem is worst-case, constructive, and does not identify the graphs in its carrier family. Results on bounded piecewise-polynomial GNNs (Khalife & Basu, 2023) and efficient non-uniform expressivity (Khalife & Tonelli-Cueto, 2025) further show why graph-size access and the uniform quantifier must be explicit.

Stable color refinement itself may require linearly many iterations (Kiefer & McKay, 2020), and same-order graph pairs can remain indistinguishable in two-variable counting logic for $n - O(\sqrt{n})$ depth (Krebs & Verbitsky, 2015). We use the latter only to transfer an $\Omega(n)$ locality lower bound to graph-level stable refinement; neither stable-CR lower bound is claimed as new.

3 A RECURRENCE INFORMATION-RATE LOWER BOUND

We write the bit length of a reduced rational p/q as $\mathrm{bits}(p/q) = O(\log(|p|+1) + \log(q+1))$.

Lemma 1 (Nested denominators)**.** *For every fixed model there are constants a, b and an integer $Q \geq 2$ such that every state coordinate after round t has a denominator dividing $D_t = Q^{at+b}$. The sequence (D_t) is nested.*

Proof. Take Q divisible by all parameter and fixed-initialization denominators. Neighbor and global sums introduce no new denominator; each rational affine layer multiplies it by at most a fixed factor;

ReLU selects the value or zero. The fixed number of affine layers per recurrence is absorbed into a. □

Lemma 2 (Magnitude growth)**.** *For any fixed finite real parameters, there is a model-dependent constant C such that every coordinate after round t has absolute value at most $2^{C(t+1)\log(n+2)}$.*

Proof. A fixed ReLU MLP is globally bounded by an affine function of its input norm. Each aggregate contains at most n states. Thus, for a suitable constant B, $M_{t+1}+1 \le B(n+1)(M_t+1)$. The initial magnitude is $O(n+1)$, and iteration gives the claim. □

Theorem 1 (Recurrence information rate)**.** *For a fixed-dimensional, fixed-rational deterministic CPWA recurrent Sum-GNN whose cell and readout are realized by ReLU MLPs, the canonical numerator–denominator description of every rational graph output after at most $T(n)$ recurrences has $O((T(n)+1)\log(n+2))$ bits. The number of distinct outputs on n-vertex graphs is at most $2^{O((T(n)+1)\log(n+2))}$. Both statements remain true with global sum and node-dependent stopping times, provided stopping time is not separately observable.*

Proof. Lemmas 1–2 bound each reduced numerator and denominator. For mixed stopping times, every node coordinate still has a denominator dividing D_T and the common worst-case magnitude bound; summing arbitrary final states adds only $O(\log n)$ bits. We therefore count possible fixed-dimensional rational outputs directly, without multiplying by the $(T+1)^n$ possible stopping schedules. A fixed readout changes only constants. Using average rather than sum readout introduces a factor n in the common denominator and hence only another $O(\log n)$ bits. Adding a global sum changes the aggregate magnitude by at most another factor n and leaves the proof unchanged. □

Let $C_t(n)$ be the number of distinct values of CR_t on n-vertex simple graphs.

Theorem 2 (Final-output-margin recurrence rate over arbitrary reals)**.** *Let a fixed-dimensional deterministic recurrent CPWA Sum-GNN whose cell and readout are realized by ReLU MLPs have arbitrary fixed finite real parameters and graph-output dimension p. There is a model-dependent constant K such that, if it δ-robustly refines CR_ℓ in at most $T(n)$ recurrences, then*

$$\log_2 C_\ell(n) \le p\left(K(T(n)+1)\log_2(n+2) + \log_2 \frac{3}{\delta(n)}\right). \tag{1}$$

The statement allows global sum and unobservable node-dependent stopping.

Proof. Lemma 2 does not use rationality. After absorbing the fixed readout, every output lies in $[-R_n, R_n]^p$ for $R_n = (n+2)^{K(T(n)+1)}$. Choose one graph from each CR_ℓ class. Their outputs are pairwise $\delta(n)$-separated in ℓ_∞. A δ-separated subset of the cube has size at most

$$\left(1 + \frac{2R_n}{\delta(n)}\right)^p \le \left(\frac{3R_n}{\delta(n)}\right)^p,$$

because $R_n \ge 1$ and $\delta(n) \le 1$. Taking logarithms proves (1). Mixed stopping and global sum remain inside the same worst-case cube. □

We next bound the number of graph-level invariants.

Lemma 3 (First-round entropy)**.**

$$C_1(n) = 2^{\Theta(n)}.$$

Proof. CR_1 is exactly the degree multiset. The number of size-n multisets over $\{0,\dots,n-1\}$ is at most $\binom{2n-1}{n} \le 4^n$. For the lower bound, consider threshold graphs with canonical creation string $x_1 = 0, x_2, \dots, x_n \in \{0,1\}$, where a zero adds an isolated vertex and a one adds a dominating vertex. Their degree multisets recover the creation string recursively: a final zero is witnessed by degree zero and no degree $n-1$, while a final one is witnessed by degree $n-1$ and no degree zero. Delete the witness, decrement the other degrees in the latter case, and repeat. Thus the 2^{n-1} strings give distinct degree multisets. □

Lemma 4 (Dense regular matrices). *Let $\mathcal{R}(m, d)$ be the set of m-by-m binary matrices with every row and column sum equal to $d = \lfloor m/2 \rfloor$. Then*

$$|\mathcal{R}(m, d)| = 2^{m^2 - m \log_2 m + O(m)}.$$

Proof. Apply Theorem 1(b) of Canfield & McKay (2005) with equal dimensions and density $\lambda = d/m = 1/2 + O(1/m)$. Its square-matrix, bounded-density, and density-deviation conditions hold. Substituting its asymptotic formula and using Stirling's approximation gives

$$\log_2 |\mathcal{R}(m, d)| = m^2 - m \log_2 m + O(m).$$

The substitution and condition check are expanded in Appendix B. □

Theorem 3 (Second-round graph entropy).

$$C_2(n) = 2^{\Theta(n^2)}.$$

The lower bound already holds on connected graphs.

Proof. Let A_m be the connected antiregular graph whose degree multiset is

$$\{1, 2, \ldots, m-1\} \text{ plus one extra copy of } \lfloor m/2 \rfloor.$$

Write its degrees as $\delta_1, \ldots, \delta_m$, and let $Q = \{p, q\}$ be its unique repeated-degree class. For each $M \in \mathcal{R}(m, \lfloor m/2 \rfloor)$, take copies $L = \{\ell_i\}$ and $R = \{r_i\}$ of A_m and add $\ell_i r_j$ exactly when $M_{ij} = 1$. Call the connected graph H_M. Every ℓ_i and r_i has degree $\delta_i + d$.

It remains to bound how many matrices can share one CR_2 value. A second-round vertex color is its degree together with the multiset of its neighbors' degrees. For each of the $m - 2$ singleton degree classes of A_m, assigning the two observed colors to its left and right roles has at most two choices. The four colors over Q have at most $4!$ role assignments. Fix one assignment. If $j \notin Q$, the color of ℓ_i gives M_{ij} after subtracting the fixed same-copy adjacency into degree class $\delta_j + d$. If $j \in Q$ and $i \notin Q$, the color of r_j gives the same entry by looking back into degree class $\delta_i + d$. Only the Q-by-Q binary block remains. Its two row sums and two column sums are fixed by the assigned colors, leaving at most two matrices. Therefore every fiber has size at most

$$2^{m-2}\, 4!\, 2 = 48 \cdot 2^{m-2}.$$

For $n = 2m$, Lemma 4 now gives at least

$$\frac{|\mathcal{R}(m, d)|}{48 \cdot 2^{m-2}} = 2^{n^2/4 - O(n \log n)}$$

different connected CR_2 values. For odd n, add one universal vertex: its unique degree identifies it in CR_2, and deleting it and shifting the old degrees back recovers the unpadded invariant. The total number of graphs gives the matching $2^{O(n^2)}$ upper bound. Appendix B gives the antiregular construction and all recovery details. □

Corollary 1 (Fixed-dimensional output-precision floor). *Fix $k \geq 2$ and a constant output dimension p. If Φ_n maps n-vertex graphs into $[-M(n), M(n)]^p$ and outputs from distinct CR_k classes have ℓ_∞ separation at least $\delta(n) \in (0, 1]$, then*

$$p \log_2 \left(1 + \frac{2M(n)}{\delta(n)}\right) \geq \log_2 C_k(n) = \Theta(n^2).$$

Consequently, $\log_2(M(n) + 1) + \log_2(1/\delta(n)) = \Omega(n^2)$. For unit-separated integer outputs, some n-vertex input has a coordinate with $\Omega(n^2)$ bits.

The architecture-independent packing proof is in Appendix A.4.

Corollary 2 (First-to-second-round recurrence transition). *Every model in Theorem 1 refining CR_1 satisfies*

$$T(n) = \Omega(n/\log n).$$

For every fixed $\ell \geq 2$, every such model refining CR_ℓ satisfies

$$T(n) = \Omega(n^2/\log n).$$

Both bounds hold even when every recurrence receives a global sum. For arbitrary fixed real parameters, the same two lower bounds hold under $\delta(n)$-robust refinement whenever $\log(1/\delta(n)) = O(\log n)$, in particular for $\delta(n) \geq n^{-c}$ with any fixed c.

Proof. A refining output must take at least $C_\ell(n)$ distinct values. Compare Lemmas 3 and Theorem 3 with Theorem 1. For $\ell > 2$, every recursive color retains its second-round ancestor, so refining CR_ℓ also refines CR_2. For the robust real-parameter statement, use Theorem 2; the additive $O(\log(1/\delta(n))) = O(\log n)$ term is lower order than $\log C_1(n) = \Theta(n)$ and $\log C_2(n) = \Theta(n^2)$. □

Theorem 4 (Polynomial-growth packing rate)**.** *Fix $k \geq 2$. Let a fixed-dimensional deterministic recurrent Sum-GNN have polynomial-magnitude initialization and a cell/readout envelope of degree $q \geq 1$. If its graph outputs for distinct CR_k classes are separated by $\delta(n) \geq n^{-O(1)}$, then*

$$\begin{cases} T(n) = \Omega(n^2/\log n), & q = 1, \\ T(n) = \Omega(\log n), & q > 1. \end{cases}$$

The statement allows neighbor and recurrent global sums.

Proof. Let M_t bound all state and intermediate gate magnitudes. Aggregation multiplies the input norm by at most n, so fixed constants B, c give

$$M_0 \leq n^c, \qquad M_{t+1} \leq A(1 + BnM_t)^q.$$

For $q = 1$, $\log M_T = O((T+1)\log n)$. For fixed $q > 1$, iterating the logarithmic recurrence gives $\log M_T \leq q^T O(\log n)$. A fixed-dimensional output cube of radius $M_T^{O(1)}$ contains at most $\exp(O(\log M_T + \log(1/\delta)))$ separated points. By Theorem 3, refining any fixed CR_k, $k \geq 2$, requires $2^{\Theta(n^2)}$ points. Comparing logarithms yields the two cases. □

4 A SINGLE MULTIPLICATION-AUGMENTED COMPILER FOR ALL ROUNDS

Theorem 5 (Uniform CPWA (ReLU-MLP) upper bounds)**.** *There are fixed-dimensional, fixed-rational deterministic CPWA recurrent Sum-GNNs, realized by ReLU MLPs and using recurrent global sum, that output unit-separated integer encodings equivalent to $\mathrm{CR}_1(G)$ in $O(n)$ recurrences and to $\mathrm{CR}_k(G)$ in $O_k(n^2)$ recurrences for every fixed $k \geq 2$.*

Lemma 5 (Unary synchronization)**.** *For a vertex partition $P_1, \ldots, P_r$, one fixed recurrent CPWA (ReLU-MLP) cell can write every node's fields $u(|N(v) \cap P_j|)$ in $O(n+r)$ synchronized recurrences.*

Proof. For field j, neighbor sum loads the integer count q_v. Each recurrence emits one exactly when $q_v > 0$ and replaces it by $\mathrm{ReLU}(q_v - 1)$; global sum detects when all nodes finish, after which every node emits a delimiter. The padded cost is at most $|P_j| + O(1)$, and summing over the partition gives $n + O(r)$. □

Proof of Theorem 5. The construction serializes unary class counts with a fixed CPWA (ReLU-MLP) stack machine. A global controller traverses the trie of at most n canonical node-color keys; the sum of key lengths is $O_k(n^2)$. A phase-independent base-four integer trace gives unit output separation. The controller details, register invariants, and recurrence accounting are in Appendices C–D. Set $K_1(v) = u(\deg v)$ and form $K_{t+1}(v)$ by appending to $K_t(v)$ the unary neighbor count for every class in the lexicographically ordered dictionary of distinct K_t keys. If L_t is the maximum key length, then $L_1 \leq n$ and $L_{t+1} \leq L_t + 2n$, because class counts sum to the degree and there are at most n delimiters. Hence $L_t = O(tn)$, and the synchronized fields, trie traversal, path copies, dictionary, and integer trace take $O(nL_t)$ recurrences at stage t. Unrolling the finite phase block for fixed k costs $O(k^2n^2) = O_k(n^2)$ and uses one fixed controller. Induction on t proves that K_t is equivalent to the recursive color c_t. □

Adding exact binary multiplication to the CPWA basis changes this recurrence scale.

Theorem 6 (One fixed cell for every refinement depth)**.** *There is a 22-dimensional, fixed-rational, deterministic recurrent Sum-GNN multiplication-augmented arithmetic cell over affine maps, scalar ReLU gates, and exact binary multiplication gates such that, for every $n \geq 1$ and integer input $1 \leq k \leq n$:*

1. recurrence uses neighbor sum but no recurrent global sum;

2. *after exactly*
$$T(n,k) = 2(k+1)\lceil \log_2(2n) \rceil + 4k + 4$$
recurrences, a final graph sum outputs an integer pair $(n, Y_k(G))$*;*

3. $Y_k(G) = Y_k(H)$ *if and only if* $\mathrm{CR}_k(G) = \mathrm{CR}_k(H)$ *for same-order graphs;*

4. *distinct outputs have* ℓ_∞ *distance at least one; and*

5. *every state and internal gate value has* $n^{O(k+1)}$ *bits.*

The same cell and parameters are used for all n *and* k*.*

Proof. Fix the graph order n and a refinement stage. Inductively suppose every node has a positive integer code $0 < r_v < R$, canonical across all n-vertex graphs at this stage. Let
$$h = 2^{\lceil \log_2(2n) \rceil}, \quad Q = R^h, \quad K = nQ + 1,$$
and define
$$S_\ell(r) = \sum_{j=0}^{\ell-1} K^{\ell-1-j} r^j.$$
The neighbor aggregate
$$N_v = \sum_{u \in N(v)} S_h(r_u)$$
has base-K digits $P_j(v) = \sum_{u \in N(v)} r_u^j$. There is no carry because $P_j(v) < nR^h < K$. The digit P_0 gives the multiset size, and the first n power sums determine every integer multiset of size at most $n-1$ by Newton's identities. Since $h - 1 \geq n$, N_v is injective on neighbor-code multisets. Set
$$r'_v = r_v + RN_v, \qquad R' = R(nK^h + 1).$$
Then $0 < r'_v < R'$, and $r'_v = r'_w$ exactly when the old code and neighbor-code multiset agree. Thus one block implements one canonical CR update. The common bound R' depends only on (n, stage), not on the graph.

It remains to generate S_h with one shared scalar state. At length ℓ, maintain
$$p_v = r_v^\ell, \quad b = K^\ell, \quad s_v = S_\ell(r_v), \quad u_v = p_v s_v, \quad z_v = \sum_{w \in N(v)} S_\ell(r_w).$$
The identities
$$S_{2\ell}(r) = (K^\ell + r^\ell) S_\ell(r)$$
and
$$\sum_w S_{2\ell}(r_w) = K^\ell \sum_w S_\ell(r_w) + \sum_w r_w^\ell S_\ell(r_w)$$
give the fixed update
$$p'_v = p_v^2, \qquad b' = b^2, \qquad s'_v = (b + p_v) s_v,$$
$$u'_v = p'_v s'_v, \quad z'_v = b z_v + \sum_{w \in N(v)} u_w, \quad \ell' = 2\ell.$$
Starting at $\ell = 1$, $\lceil \log_2(2n) \rceil$ recurrences reach h. The same number of preparatory squarings produces $Q = R^h$. Each refinement block therefore costs $2\lceil \log_2(2n) \rceil + 4$ recurrences. After k updates, one final block packs the multiset of node codes and exposes $Y_k(G) = \sum_v S_h(r_v)$ through the graph sum; with n summands, its digits still satisfy $P_j < nR^h < K$. Accounting for initialization and the final block gives the displayed exact total.

Six one-hot phases (INIT, PREP, EVALINIT, EVAL, COMMIT, DONE) and integer positive tests $\mathrm{ReLU}(x) - \mathrm{ReLU}(\mathrm{ReLU}(x) - 1)$ compile this schedule into one branch-free fixed cell. Its machine-readable DAG has 24 inputs, 22 outputs, 124 total nodes, and depth 11. Appendix E lists the state transition and certificate.

For the bit bound, if $L_j = \lceil \log_2 R_j \rceil$, then $h < 4n$ and

$$L_{j+1} \le (1 + 16n^2)L_j + 4n \log_2(2n).$$

Starting from $R_0 = 2$ gives $L_j = n^{O(j)}$. Neighbor sums add only $O(\log n)$ bits and the fixed-depth arithmetic DAG adds input bit lengths only a constant number of times, proving the bound for all states and intermediate gates. □

Corollary 3 (Arithmetic-growth recurrence hierarchy)**.** *For every fixed $k \ge 2$,*

$$\mathrm{RecComp}_{\times}(\mathrm{CR}_k) = \Theta(\log n).$$

In contrast, the fixed CPWA (ReLU-MLP) model satisfies

$$\Omega(n^2/\log n) \le \mathrm{RecComp}(\mathrm{CR}_k) \le O_k(n^2).$$

The multiplication-augmented result uses one cell jointly for all input k, whereas the CPWA upper controller may depend on the fixed k.

Proof. The multiplication upper bound is Theorem 6; its cell has a fixed polynomial-growth envelope, so Theorem 4 supplies the matching lower bound. At every finite order use the effective request $k_n = k - \mathrm{ReLU}(k - n) = \min\{k, n\}$; this is a fixed rational ReLU selector, and CR_n is already stable when $n < k$. Thus one model covers all orders, including the finitely many $n < k$. The ReLU bounds combine Corollary 2 and Theorem 5. □

Corollary 4 (Stable color refinement)**.** *In the same multiplication-augmented arithmetic model,*

$$\Omega(n) \le \mathrm{RecComp}_{\times}(\text{stable CR}) \le O(n \log n).$$

Proof. For an n-vertex graph, CR_n records the stabilized quotient: its histogram gives every stable class size, while its colors give the neighbor-count rows. Consequently, equal same-order CR_n histograms remain equal at every later round; see Appendix E. Setting the ordinary input $k = n$ in Theorem 6 therefore gives the upper bound. Appendix E proves that even recurrent global sum cannot separate graphs with equal CR_T before round T. The eventually distinguishable pairs of Krebs & Verbitsky (2015) remain equivalent for $n - O(\sqrt{n})$ rounds, yielding the lower bound. □

5 Implications and Limitations

Weight sharing fixes parameter count, not the rate at which a recurrent state acquires exact information. Fixed CPWA cells expose only $O(\log n)$ robust output bits per recurrence, whereas exact multiplication permits exponentially faster bit growth and $O(\log n)$ recurrence. Corollary 1 shows that any fixed-dimensional, unit-separated representation of fixed CR_k, $k \ge 2$, has an $\Omega(n^2)$-bit output coordinate on some input. Thus the augmented gate basis reduces recurrence count but does not imply a bit-complexity or wall-clock improvement.

Size-dependent networks, growing width, randomized fingerprints (Aamand et al., 2022), or vanishing output separation change the contract. Our construction keeps one rational, permutation-equivariant cell independent of n and uses no recurrent global sum, while allowing exact integers with $n^{O(k+1)}$ bits. The lower bounds concern exact refinement of every graph, not average accuracy, approximate embeddings, or a fixed maximum order.

Open gaps. The CPWA interval remains $\Omega(n^2/\log n)$ to $O_k(n^2)$; the multiplication-augmented precision upper bound is not matched to the $\Omega(n^2)$ floor; and stable refinement remains $\Omega(n)$ to $O(n \log n)$. The extent to which comparable recurrence lower bounds survive when the required output separation may vanish faster than inverse-polynomial remains open. The compiler is an existence construction, not a trainable or numerically stable architecture. The result neither implies fixed-word execution nor claims that CR_2 identifies almost all graphs.

6 Conclusion

For every fixed $k \ge 2$, fixed CPWA cells realized by ReLU MLPs need $\Omega(n^2/\log n)$ recurrences, whereas one 22-dimensional multiplication-augmented arithmetic cell attains $\Theta(\log n)$. The latter still obeys an architecture-independent $\Omega(n^2)$ output-precision floor. For stable refinement, the corresponding recurrence interval is $\Omega(n)$–$O(n \log n)$.

AI USE STATEMENT

Generative AI tools were used to help develop and critique the theoretical framing, formulate and check mathematical claims, assist with proof writing, refine hypotheses and methodology, implement proof-regression checkers, and interpret exact-check outputs. They were also used for literature search, paper drafting and restructuring, readability edits, title and keyword suggestions, and reference formatting. They were not used to generate datasets, translate text, clean or reformat datasets, or conduct qualitative or thematic data analysis; those tasks are not applicable to this work. The authors checked the resulting claims and citations against primary sources, reviewed every proof, and tested AI-assisted code through exact computations and independent implementations. AI output was not treated as scientific evidence. The authors take responsibility for the final text, claims, and artifacts.

ETHICS STATEMENT

This theoretical work uses no human participants, private or sensitive data, or deployed decision system. We identify no special ethical risk beyond the general obligations of research integrity, accurate attribution, and truthful reporting of AI assistance.

REPRODUCIBILITY STATEMENT

The appendices give the assumptions, complete proofs, fixed-cell transition, and complexity accounting. The supplement provides exact-rational scripts, frozen outputs, runtime versions, independent color-refinement and reverse reconstruction implementations, the arithmetic DAG, and certificate checks. These finite computations are regression checks, not evidence for the asymptotic theorems.

REFERENCES

Anders Aamand, Justin Chen, Piotr Indyk, Shyam Narayanan, Ronitt Rubinfeld, Nicholas Schiefer, Sandeep Silwal, and Tal Wagner. Exponentially improving the complexity of simulating the weisfeiler–lehman test with graph neural networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27333–27346, 2022.

Veeti Ahvonen, Damian Heiman, Antti Kuusisto, and Carsten Lutz. Logical characterizations of recurrent graph neural networks with reals and floats. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Timon Barlag, Vivian Holzapfel, Laura Strieker, Jonni Virtema, and Heribert Vollmer. Graph neural networks and arithmetic circuits. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0175.

Timon Barlag, Vivian Holzapfel, Laura Strieker, Jonni Virtema, and Heribert Vollmer. Recurrent graph neural networks and arithmetic circuits. In *Proceedings of the 23rd International Conference on Principles of Knowledge Representation and Reasoning*, pp. 826–836, 2026. doi: 10.24963/kr.2026/78.

Michael D. Barrus and Elizabeth A. Donovan. Neighborhood degree lists of graphs. *Discrete Mathematics*, 341(1):175–183, 2018. doi: 10.1016/j.disc.2017.08.027.

Kevin E. Bassler, Charo I. Del Genio, Péter L. Erdős, István Miklós, and Zoltán Toroczkai. Exact sampling of graphs with prescribed degree correlations. *New Journal of Physics*, 17(8):083052, 2015. doi: 10.1088/1367-2630/17/8/083052.

Jan Böker, Ron Levie, Ningyuan Huang, Soledad Villar, and Christopher Morris. Fine-grained expressivity of graph neural networks. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

E. Rodney Canfield and Brendan D. McKay. Asymptotic enumeration of dense 0–1 matrices with equal row sums and equal column sums. *The Electronic Journal of Combinatorics*, 12:R29, 2005. URL https://www.combinatorics.org/Volume_12/PDF/v12i1r29.pdf.

Guanyu Cui, Yuhe Guo, Zhewei Wei, and Hsin-Hao Su. How hard is it for message-passing gnns to simulate one weisfeiler–lehman color-refinement step? *arXiv preprint arXiv:2410.01308*, 2024. doi: 10.48550/arXiv.2410.01308.

MohammadReza Ebrahimi and Roland Memisevic. Revisiting bi-linear state transitions in recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://openreview.net/forum?id=xcqrAzYzCj.

Billy Joe Franks, Christopher Morris, Ameya Velingker, and Floris Geerts. Weisfeiler–leman at the margin: When more expressivity matters. In *International Conference on Machine Learning*, volume 235, pp. 13885–13926, 2024.

Reza Jafarpour-Golzari. Degree polynomial for vertices in a graph and its behavior under graph operations. *Commentationes Mathematicae Universitatis Carolinae*, 63(4):397–413, 2022. doi: 10.14712/1213-7243.2023.006. URL https://eudml.org/doc/299038.

Sammy Khalife and Amitabh Basu. On the power of graph neural networks and the role of the activation function. *arXiv preprint arXiv:2307.04661*, 2023.

Sammy Khalife and Josué Tonelli-Cueto. Is uniform expressivity too restrictive? towards efficient expressivity of gnns. In *International Conference on Learning Representations*, 2025.

Sandra Kiefer and Brendan D. McKay. The iteration number of colour refinement. In *47th International Colloquium on Automata, Languages, and Programming*, volume 168 of *Leibniz International Proceedings in Informatics*, pp. 73:1–73:19, 2020. doi: 10.4230/LIPIcs.ICALP.2020.73.

Andreas Krebs and Oleg Verbitsky. Universal covers, color refinement, and two-variable counting logic: Lower bounds for the depth. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science*, pp. 689–700, 2015. doi: 10.1109/LICS.2015.69.

Lorenz Kummer, Wilfried N. Gansterer, and Nils Morten Kriege. On the relationship between robustness and expressivity of graph neural networks. In *International Conference on Artificial Intelligence and Statistics*, volume 258, pp. 1243–1251, 2025.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. In *International Conference on Learning Representations*, 2016.

Andreas Loukas. How hard is to distinguish graphs with graph neural networks? In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Andreas Madsen and Alexander Rosenberg Johansen. Neural arithmetic units. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1gNOeHKPS.

Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI Conference on Artificial Intelligence*, 2019.

Christopher Morris, Yaron Lipman, Haggai Maron, Bastian Rieck, Nils M. Kriege, Martin Grohe, Matthias Fey, and Karsten M. Borgwardt. Weisfeiler and leman go machine learning: The story so far. *Journal of Machine Learning Research*, 24(333):1–59, 2023. URL https://jmlr.org/papers/v24/22-0240.html.

Eran Rosenbluth. Lost in aggregation: On a fundamental expressivity limit of message-passing graph neural networks. *arXiv preprint arXiv:2603.14846*, 2026.

Eran Rosenbluth and Martin Grohe. Repetition makes perfect: Recurrent graph neural networks match message passing limit. In *AAAI Conference on Artificial Intelligence*, volume 40, pp. 25168–25175, 2026. doi: 10.1609/aaai.v40i30.39707.

Joel Rybicki, Oleg Verbitsky, and Maksim Zhukovskii. What can be computed in average anonymous networks? *arXiv preprint arXiv:2604.02192*, 2026.

Andrew Trask, Felix Hill, Scott Reed, Jack Rae, Chris Dyer, and Phil Blunsom. Neural arithmetic logic units. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Yuhuai Wu, Saizheng Zhang, Ying Zhang, Yoshua Bengio, and Ruslan Salakhutdinov. On multiplicative integration with recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

A Lower-bound details

A.1 Model details and mixed stopping times

Let the fixed cell contain L rational affine layers. Choose an integer Q divisible by every parameter and fixed feature denominator. Because n is an integer, there is a constant b such that all initial coordinates have denominator dividing Q^b. If all round-t coordinates have denominators dividing D_t, a coordinate-wise sum preserves that divisor. One affine layer changes it to a divisor of QD_t, and ReLU adds no denominator. Hence $D_t = Q^{Lt+b}$ is a common nested denominator.

For the magnitude, a fixed ReLU MLP has constants α, β such that $\|F(z)\|_\infty \le \alpha\|z\|_\infty + \beta$. Both the neighborhood and global sum have magnitude at most nM_t. After absorbing the self-state and bias,

$$M_{t+1} + 1 \le C(n+1)(M_t + 1).$$

Since $M_0 = O(n+1)$, $\log(M_t + 1) = O((t+1)\log(n+2))$.

For node-dependent stopping, let $I_v \le T$. Every stopped coordinate has a denominator dividing D_T and magnitude at most $\max_{t \le T} M_t$. A sum of any mixture of these values has the same common denominator and at most an additional factor n in magnitude. Thus the number of possible mixed final sums is bounded by the range of a fixed-dimensional rational vector of $O(T \log n)$ bits; no factor for the number of stopping schedules is needed. A node that halts can equivalently remain in an absorbing state, which keeps it inside the recurrence bounds.

A.2 Output convention

The graph output is a fixed-dimensional exact rational vector. The graph readout is a fixed rational ReLU MLP and changes the numerator–denominator description length only by a model-dependent constant; average aggregation adds only the denominator factor n. There is no external decoder, independent output tape, or stopping-time channel. The upper-bound construction uses one integer coordinate whose base-four digits encode the canonical finite bit string; equality of outputs is therefore ordinary integer equality, and distinct outputs are separated by at least one.

A.3 Arbitrary-real output packing

The magnitude argument above is independent of rationality. Let the cell and readout instead have arbitrary fixed finite real parameters, and let the graph output have fixed dimension p. The finitely many operator norms, biases, layers, and coordinates are absorbed into a constant $B \ge 1$. With M_t denoting the largest state magnitude at any node after any round at most t,

$$M_{t+1} + 1 \le B(n+1)(M_t + 1), \qquad M_0 + 1 \le B(n+1).$$

Increasing a model-dependent exponent constant if necessary therefore gives

$$M_t + 1 \le (n+2)^{K_0(t+1)}.$$

Sum readout contributes at most one additional factor n, average readout no more, and the fixed real ReLU readout changes only the exponent constant. Thus all graph outputs after at most $T(n)$ recurrences lie in

$$[-R_n, R_n]^p, \qquad R_n = (n+2)^{K(T(n)+1)}.$$

This also covers mixed stopping: every stopped node selects one state from a round at most $T(n)$, all of which obey the same bound.

Around every point in a $\delta(n)$-separated set, place the open ℓ_∞ ball of radius $\delta(n)/2$. These balls have disjoint interiors and all lie in the cube enlarged by $\delta(n)/2$ in every coordinate. Comparing volumes shows that the set has cardinality at most

$$\left(1+\frac{2R_n}{\delta(n)}\right)^p \le \left(\frac{3R_n}{\delta(n)}\right)^p.$$

Selecting one graph from every graph-level CR_ℓ class proves Theorem 2. The argument neither counts nor approximates the real parameters. Infinite parameter precision can create arbitrarily small distinctions, but it cannot create more $\delta(n)$-separated points inside the bounded output cube.

A.4 Architecture-independent output precision

Choose one graph from every graph-level CR_k class. Their output vectors are pairwise $\delta(n)$-separated in $[-M(n), M(n)]^p$. Partitioning each coordinate interval into subintervals of length $\delta(n)$ gives

$$C_k(n) \le \left(1+\frac{2M(n)}{\delta(n)}\right)^p.$$

For every fixed $k \ge 2$, recursive colors retain their second-round ancestors, so Theorem 3 and the total graph count give $\log_2 C_k(n) = \Theta(n^2)$. Taking logarithms proves the first statement of Corollary 1; the inequality $1+2M/\delta \le 3(M+1)/\delta$ gives the magnitude–margin form. With $\delta = 1$ and integer outputs, let M be the largest absolute coordinate over all n-vertex inputs. Hence, for every sufficiently large n, some input-coordinate pair requires $\Omega(n^2)$ bits. No property of the cell architecture is used beyond fixed output dimension and the stated separation.

B CR1–CR2 entropy details

B.1 First-round classes

There are at most

$$\binom{2n-1}{n} \le 4^n$$

size-n multisets over the n possible degrees. For the matching lower bound, let $x_1 = 0$ and choose $x_2, \dots, x_n \in \{0,1\}$. At step j, a zero adds a vertex adjacent to no earlier vertex and a one adds a vertex adjacent to every earlier vertex.

If $x_n = 0$, the final graph has an isolated vertex and no universal vertex. If $x_n = 1$, it has a universal vertex and no isolated vertex. The degree multiset therefore determines x_n. In the first case delete one zero; in the second delete one $n-1$ and subtract one from every remaining degree. The resulting multiset belongs to the graph built from $x_1, \dots, x_{n-1}$. Induction recovers the whole string, proving that all 2^{n-1} degree multisets are distinct.

B.2 Dense regular matrix count

Write $B(m, s; n, t)$ for the number of m-by-n binary matrices with row sum s and column sum t, and let $\lambda = s/n = t/m$. Theorem 1(b) of Canfield & McKay (2005) states, under its relatively-square and density-deviation conditions,

$$B(m,s;n,t) = \frac{\binom{n}{s}^m \binom{m}{t}^n}{\binom{mn}{\lambda mn}} \left(\frac{m-1}{m}\right)^{(m-1)/2} \left(\frac{n-1}{n}\right)^{(n-1)/2} \exp(1/2 + o(1)).$$

Use equal dimensions, $s = t = d = \lfloor m/2 \rfloor$, and $\lambda = d/m = 1/2 + O(1/m)$. Both dimensions tend to infinity, their ratio is one, and $\lambda(1-\lambda)$ is bounded away from zero. Moreover, $(1-2\lambda)^2 m = O(1/m)$, which is smaller than the right side of the theorem's density-deviation condition. Thus

$$|\mathcal{R}(m,d)| = \frac{\binom{m}{d}^{2m}}{\binom{m^2}{dm}} \left(\frac{m-1}{m}\right)^{m-1} \exp(1/2 + o(1)).$$

Stirling's formula gives

$$\log_2 \binom{m}{d} = mh_2(\lambda) - \tfrac{1}{2}\log_2 m + O(1),$$
$$\log_2 \binom{m^2}{dm} = m^2 h_2(\lambda) - \log_2 m + O(1),$$

where $h_2(\lambda) = 1 - O(1/m^2)$. The remaining multiplicative factor has constant logarithm. Consequently,

$$\log_2 |\mathcal{R}(m,d)| = m^2 - m\log_2 m + O(m).$$

B.3 The antiregular carrier

For each $m \geq 2$, construct a threshold graph A_m by setting its first creation bit to zero and, for $j \geq 2$, making vertex j dominating exactly when j and m have the same parity. The final vertex is dominating, so A_m is connected.

If $m = 2k$, dominating vertex $2a$ has final degree

$$(2a-1) + (k-a) = k + a - 1, \qquad a = 1, \ldots, k,$$

and non-dominating vertex $2a-1$ has final degree

$$k - a + 1, \qquad a = 1, \ldots, k.$$

The two lists together are $1, \ldots, 2k-1$ with one additional copy of k. If $m = 2k+1$, dominating vertex $2a+1$ has degree $2a + (k-a) = k + a$, while the first vertex and vertices $2a$ have degrees k and $k - a + 1$, respectively. These are $1, \ldots, 2k$ with one additional copy of k. Hence in both cases the degree multiset is

$$\{1, \ldots, m-1\} \text{ plus one extra } \lfloor m/2 \rfloor.$$

Index the vertices so their degrees are $\delta_1, \ldots, \delta_m$, and write $Q = \{p, q\}$ for the unique repeated-degree class. Given $M \in \mathcal{R}(m,d)$, take copies $L = \{\ell_1, \ldots, \ell_m\}$ and $R = \{r_1, \ldots, r_m\}$ of A_m and add $\ell_i r_j$ exactly when $M_{ij} = 1$. In the resulting graph H_M,

$$\deg(\ell_i) = \deg(r_i) = \delta_i + d.$$

Both copies are connected and $d \geq 1$, so H_M is connected.

B.4 Fiber reconstruction

Fix one graph-level second-round invariant Y. Since

$$c_2(v) = (\deg(v), \{\{\deg(w) : w \in N(v)\}\}),$$

Y first groups its colors by the degree of their owning vertices. For each singleton degree class $i \notin Q$, assign its two occurrences to ℓ_i, r_i in at most two ways. Assign the four occurrences over Q to ℓ_p, ℓ_q, r_p, r_q in at most $4!$ ways. There are therefore at most $2^{m-2}4!$ role assignments.

Fix one such assignment. For $j \notin Q$, the assigned color of ℓ_i reveals its number of neighbors of degree $\delta_j + d$. This degree class contains ℓ_j and r_j. Subtracting the known internal A_m adjacency from ℓ_i to ℓ_j leaves exactly M_{ij}.

When $j \in Q$ but $i \notin Q$, use the assigned color of r_j instead. Its neighbors of degree $\delta_i + d$ are the fixed same-copy neighbor r_i and the possible cross-copy neighbor ℓ_i. Subtracting the first recovers M_{ij}. Thus only the Q-by-Q binary block remains.

The assigned colors of ℓ_p, ℓ_q determine the two row sums of this block after fixed internal adjacencies are subtracted. The colors of r_p, r_q similarly determine its two column sums. Once the upper-left bit of a binary 2-by-2 matrix is chosen, these four sums force the other three entries. There are at most two choices. Therefore

$$|\{M \in \mathcal{R}(m,d) : \mathrm{CR}_2(H_M) = Y\}| \leq 2^{m-2}4!\,2 = 48 \cdot 2^{m-2}.$$

For $n = 2m$, the number of distinct connected invariants among the H_M is therefore

$$\frac{2^{m^2 - m\log_2 m + O(m)}}{48 \cdot 2^{m-2}} = 2^{n^2/4 - O(n\log n)}.$$

For $n = 2m + 1$, add a universal vertex z to every H_M. It is the unique vertex of degree $2m$, because each old degree is at most $m - 1 + \lfloor m/2 \rfloor + 1 < 2m$. From the padded CR_2 value, remove the color of z; in each old color remove the unique neighbor degree $2m$ and subtract one from the owning degree and every remaining neighbor degree. This recovers the original invariant exactly, so no classes merge. The padded graphs remain connected. Together with the trivial $2^{\binom{n}{2}}$ upper bound, this proves $C_2(n) = 2^{\Theta(n^2)}$ on all graphs and on connected graphs.

C Fixed CPWA (ReLU-MLP) stack machine

C.1 Exact primitives

For nonnegative integer z, $\mathrm{pos}(z) = \mathrm{ReLU}(z) - \mathrm{ReLU}(z-1)$ is exactly Boolean. Integer equality follows by applying it to $|x-y| = \mathrm{ReLU}(x-y) + \mathrm{ReLU}(y-x)$. Boolean conjunction is $\mathrm{ReLU}(a + b - 1)$.

For a valid gap stack $x = S(b_1 \dots b_m)$, the first base-four digit is either one or three. Therefore

$$\begin{aligned}
\mathrm{top}(x) &= \mathrm{clip}_{01}(4x - 2),\\
\mathrm{pop}(x) &= 4x - (2\,\mathrm{top}(x) + 1),\\
\mathrm{pop}_0(x) &= \mathrm{ReLU}(\mathrm{pop}(x)),\\
\mathrm{push}(b, x) &= (2b + 1 + x)/4,\\
\mathrm{nonempty}(x) &= \mathrm{clip}_{01}(4x),
\end{aligned}$$

where $\mathrm{clip}_{01}(y) = \mathrm{ReLU}(y) - \mathrm{ReLU}(y-1)$. The nonempty formula is valid because every nonempty stack is at least $1/4$, while the empty stack is zero. The safe candidate pop_0 equals pop on every valid nonempty stack and is zero on the empty stack. For Boolean a and bounded stack coordinate x, $\mathrm{gate}(a, x) = \mathrm{ReLU}(x) - \mathrm{ReLU}(x - a)$ is x when $a = 1$ and zero otherwise.

The map S is injective on finite bit strings. A strict extension increases the value by a positive tail. Otherwise, at the first differing bit j the base-four digit gap contributes $2 \cdot 4^{-j}$, while the absolute value of all later digits is at most $\sum_{i>j} 3 \cdot 4^{-i} = 4^{-j}$. Thus distinct finite strings have distinct rational coordinates. This fact lets each construction manipulate its canonical graph code with bounded exact stacks. The final graph output instead uses the constant-margin integer trace below.

C.2 One fixed cell and unbounded registers

All phase-dependent quantities are Boolean or bounded stack coordinates. A finite one-hot phase controller selects among their candidate updates using the bounded gate above. Unbounded integer coordinates are never selected by such a gate. They obey the same formula in every active phase:

$$\begin{aligned}
n^+ &= n, & d^+ &= d + C_{\mathrm{deg}},\\
k^+ &= k + \mathrm{inc}_k, & q^+ &= \mathrm{ReLU}(q-1) + C_{\mathrm{deg}} + C_{\mathrm{nei}} + C_{\mathrm{glob}},\\
z^+ &= 4z + e + 2\,\mathrm{ReLU}(e + b - 1).
\end{aligned}$$

Only Boolean outgoing flags are phase-gated. Their neighbor or global sum creates $C_{\mathrm{deg}}, C_{\mathrm{nei}}, C_{\mathrm{glob}}$. The protocol invariant makes these load pulses disjoint and loads q only at zero. Thus a fixed finite composition of rational affine maps and ReLUs realizes every microstep. Here e is the bounded Boolean output-emission flag and b is the emitted Boolean bit; outside the final output reversal $e = 0$. The number of coordinates is the number of named scalar flags, counters, message channels, and stacks below, all independent of n.

Lemma 6 (Fixed-cell compilation)**.** *Consider a finite microprogram whose state consists of a one-hot finite controller, finitely many Boolean flags, finitely many valid gap stacks, and the five non-negative integer registers (n, d, k, q, z). Suppose each transition: (i) tests only integer zero/equality,*

Boolean flags, and stack top/nonempty; (ii) applies only hold, push, or safe-pop to gap stacks; (iii) publishes only Boolean neighbor/global messages; and (iv) updates unbounded registers by the five phase-independent equations above, with the three incoming load channels pairwise disjoint and every load into q occurring when $q = 0$. Then one fixed-dimensional rational CPWA Sum-GNN cell, realized by a ReLU MLP, realizes one microprogram transition in a fixed number of recurrences.

Proof. Integer tests and Boolean transition logic are the fixed ReLU circuits above. Every candidate bounded-stack update lies in $[0, 1]$ and is selected by a one-hot Boolean using gate; a finite sum of selected candidates gives the next stack. Outgoing coordinates similarly gate only Boolean flags. Neighbor and global aggregation then create the three integer load channels. The five unbounded updates are single fixed continuous piecewise-affine formulas and are never selected by a phase flag. Since the controller and the number of coordinates, candidate transitions, and ReLU primitives are finite, their parallel composition is one fixed rational ReLU MLP, hence one fixed CPWA cell. A fixed message-latency constant is represented by explicit publish/receive phases. □

Fixed-cell realization and the first-round specialization. Fix a requested depth k. The phase table below, together with the finite FIELD, REVERSE, COPY, and TRIE templates, has finitely many one-hot phases and uses exactly the finite register and message inventory listed in Appendix D. Every branch predicate and every selected bounded update is one of the explicit ReLU circuits above, while the five unbounded coordinates obey the phase-independent equations in Lemma 6. Lemma 6 therefore compiles the unrolled fixed-k program to one finite-width rational CPWA (ReLU-MLP) cell, with width and weights independent of n. For $k = 1$, the specialization stops after DEGFIELD: the neighbor sum gives $d_v = \deg(v)$ and FIELD writes the canonical unary key $u(d_v)$. Reverse it into a read stack, run TRIE on those keys, and at each leaf use the listed global publish/receive plus FIELD block to append the key and its multiplicity. The final integer-trace block therefore encodes exactly the degree histogram, i.e. CR_1.

The compiler below satisfies the lemma's load premise through the following register invariants. Register n is initialized to the supplied graph size and held forever. Register d starts at zero, receives $C_{\deg}$ exactly once in the degree-receive phase, and then holds. Register k starts at zero, increments once after each completed degree-class field, stops when $k = n$, and is never reset or reused. Register q is initialized to n to write $u(n)$; thereafter a field may load it only after the preceding countdown has reached zero. Its update then decrements it on every count recurrence until zero, which is its reset. Degree-class and second-round-class counts use only C_{nei}; multiplicities use only C_{glob}. All other global probes have separate bounded message coordinates and cannot enter q. Register z starts at zero. Every active recurrence applies its fixed equation, so it stays zero before output. During final reversal, $e = 1$ and b is the bounded stack top. Once the last output bit is popped, all nodes halt synchronously and the model's completed-state semantics holds z fixed.

C.3 CONSTANT-MARGIN INTEGER TRACE

Let the common canonical output stack contain logical string $b_1 \dots b_L$. Because append operations stored the string in reverse, the final output block pops $b_L, \dots, b_1$. Starting from zero, the L emitting updates give

$$Z(b_1 \dots b_L) = \sum_{i=1}^{L} (2b_i + 1)4^{i-1}.$$

All earlier active recurrences have $e = 0$ while $z = 0$, so their number has no effect. The digits of Z are in $\{1, 3\}$ and its most significant digit is nonzero. Uniqueness of finite base-four representation therefore gives

$$x \neq y \quad \Longrightarrow \quad Z(x) \neq Z(y) \quad \Longrightarrow \quad |Z(x) - Z(y)| \geq 1.$$

The graph readout needs only the integer Z itself. Repeated division by four would recover the original string from least-significant digits, but no such decoder is part of the model. The coordinate may have $\Theta(L)$ bits; this is a final-output separation result, not a bounded-dynamic-range or finite-word implementation claim.

D COMPILER INVARIANTS AND RECURRENCE ACCOUNTING

D.1 STATE INVENTORY

The local state uses: five unbounded scalars (n, d, k, q, z); one-hot finite phase flags; local Boolean $(\mathrm{active}, \mathrm{candidate})$; fixed neighbor and global message coordinates; append/read stacks for the output, K_2, K_3, the two dictionaries, and unary class counters; three copy/path temporary stacks; and the four DFS stacks $(P_{\mathrm{rev}}, H_{\mathrm{active}}, H_{\mathrm{move}}, H_{\mathrm{bit}})$ plus a constant-alphabet frame stack. Reusing stacks between stages is possible but unnecessary. This is a fixed state dimension.

D.2 MACRO-PHASE TABLE

Table 2 lists every phase block of the compiler. A named FIELD, REVERSE, or COPY block is the fixed microprogram given immediately after the table; the only recursive block, TRIE, is specified by Table 3. Together, the two tables define a finite program in which every loop body is instantiated.

Table 2: Macro-phase program.

Block	Bounded state and messages	Unbounded-register effect	Exit
NFIELD	Run FIELD into A_{out}; q is initialized to n.	$q^+ = \mathrm{ReLU}(q-1)$ until zero; n, d, k hold.	DEGPUB
DEGPUB	Every node publishes Boolean one on the dedicated degree channel.	All registers hold because the incoming degree channel is zero.	DEGRECV
DEGRECV	Clear the outgoing degree flag.	Since $q = 0$, $d^+ = d + C_{\mathrm{deg}}$ and $q^+ = C_{\mathrm{deg}}$; the other load channels are zero.	DEGFIELD
DEGFIELD	Run FIELD into A_2.	Countdown q to zero.	KTEST
KTEST	Test $[k = n]$.	Registers hold.	true: REV2; false: KPUB
KPUB	Publish Boolean $[d = k]$ on the class-neighbor channel.	Registers hold.	KRECV
KRECV	Clear the outgoing class flag.	Since $q = 0$, set $q^+ = C_{\mathrm{nei}} = \lvert N(v) \cap \{d = k\}\rvert$.	KFIELD
KFIELD	Run FIELD into A_2.	Countdown q to zero.	KINC
KINC	Emit the Boolean pulse $\mathrm{inc}_k = 1$.	$k^+ = k + 1$; all other registers hold.	KTEST
REV2	Run REVERSE(A_2, K_2).	All unbounded registers hold at $q = 0$.	COPY2
COPY2	Run COPY(K_2, A_3) preserving	All unbounded registers hold.	TRIE2
TRIE2	Run TRIE on K_2. Each leaf copies its real path to A_{D2} and pushes one to A_r.	At a leaf, publish active on the class-neighbor channel; receive its sum into zero q and run FIELD into A_3.	after root return: FINISH2
FINISH2	Push zero to A_r; reverse A_r, A_{D2}, A_3; copy $u(r)$ and D_2 into A_{out}.	All unbounded registers hold at $q = 0$.	TRIE3
TRIE3	Run TRIE on K_3. Each leaf copies its path to A_{D3} and pushes one to A_s.	At a leaf, publish active globally; receive the multiplicity into zero q and run FIELD into A_{D3}.	after root return: FINISH3
FINISH3	Push zero to A_s; reverse A_s, A_{D3}; copy $u(s)$ and the third-round list into A_{out}.	All unbounded registers hold at $q = 0$.	OUTPUT
OUTPUT	Pop A_{out} into K_{out}; each pop sets $e = 1$ and exposes bit b, and the last pop halts all nodes synchronously.	$z^+ = 4z + 1 + 2b$ on every emitting step; other registers hold.	HALT

The three non-DFS templates are finite. FIELD has a START phase and a recurrent COUNT phase. START conditionally pushes one exactly when $\mathrm{pos}(q) = 1$ and publishes that Boolean on a dedicated global completion channel and ignores the stale incoming value; a node with $q = 0$ holds its stack.

Each COUNT recurrence receives the previous positive sum. If it is positive, every node again conditionally pushes one exactly when its current $\mathrm{pos}(q) = 1$ and publishes that Boolean; already-zero nodes wait without pushing. If the received sum is zero, all current q are already zero, so every node pushes the single zero delimiter and exits. The phase-independent equation decrements q on every such recurrence, including START; there is no intervening control phase that could silently consume a count. For $q = 0$, START publishes no one and the next recurrence writes the delimiter. REVERSE conditionally pops every nonempty source and pushes the popped bit to the destination until the global sum of source-nonempty flags is zero. Preserving COPY first performs the same synchronized pass into both the destination and a temporary stack, then reverses the temporary stack back into the source. Every selection concerns only Boolean or bounded stack coordinates. The final OUTPUT block is a specialized reversal because every node has the same nonempty A_{out} stack. Each active recurrence pops and emits one bit. The bounded predicate $\neg\,\mathrm{nonempty}(\mathrm{pop}_0(A_{\mathrm{out}}))$ detects the last pop in that same recurrence and requests synchronous halt, so no non-emitting recurrence follows a nonzero z update.

Table 3: Phases of TRIE(K).

Phase	Per-node/controller transition	Next phase
LEAFPUB	Publish $\mathrm{active} \wedge \neg\,\mathrm{nonempty}(K_{\mathrm{cur}})$.	LEAFRECV
LEAFRECV	If the global sum is positive, execute the stage-specific leaf body in Table 2.	leaf body, then CHILD0PUB; otherwise CHILD0PUB
CHILD0PUB	Publish $\mathrm{active} \wedge \mathrm{nonempty}(K_{\mathrm{cur}}) \wedge [\mathrm{top}(K_{\mathrm{cur}}) = 0]$.	CHILD0RECV
CHILD0RECV	If the sum is positive, push parent histories and frame, conditionally pop matching cursors, set active to candidate, and push path bit zero.	child LEAFPUB; if absent CHILD1PUB
CHILD1PUB	Publish the analogous candidate flag for bit one.	CHILD1RECV
CHILD1RECV	If the sum is positive, perform the same descent with path bit one.	child LEAFPUB; if absent RETURN
RETURN	If at the root, terminate the trie. Otherwise pop histories/frame/path, restore a moved cursor bit, restore parent active, and mark in the parent frame which child returned.	parent CHILD1PUB after child zero; parent RETURN after child one

All nodes carry the same one-hot phase, frame stack, and path stack. Phase branches depend only on global sums, hence cannot desynchronize. Local candidate flags change cursors and messages but never the global controller. The stage-specific leaf body itself consists only of the listed COPY, publish/receive, and FIELD blocks, after which the saved trie phase resumes.

D.3 MESSAGE LATENCY AND UNARY COUNTDOWN

A load has two microsteps. In the publish state, a Boolean flag is written to the outgoing coordinate. In the next cell application its sum is available as C_{nei} or C_{glob} and is added to zero scratch q. In count state, a node conditionally pushes one and updates q to $\mathrm{ReLU}(q - 1)$. The positive flag of the old q is simultaneously published globally. After the last one, the next global probe is zero; a delimiter is then pushed. This adds only constant latency to the maximum count.

D.4 COPY DIRECTIONS

Logical bits are emitted left to right into an append stack, and therefore stored in reverse. Popping that stack into an empty read stack restores the logical order. To copy a read stack while preserving it, pop each bit into both the destination append stack and a temporary stack, then pop the temporary stack back into the source. Both passes are linear. A controller path is stored newest-edge first; moving it to a temporary stack exposes the root-first order, and emitting while restoring preserves the path in twice its depth.

D.5 DFS INVARIANT

At a current global prefix p, each node has a suffix stack K_{cur}. The invariant is:

1. $\mathrm{active}_v = 1$ iff the key of v begins with p;
2. for active nodes, K_{cur} is the key after deleting p;
3. for inactive nodes, the move and bit histories record every conditional cursor change needed to restore the nearest active ancestor;
4. every node has the same controller frame and path stack.

The local child candidate is

$$\mathrm{active}_v \wedge \mathrm{nonempty}(K_{\mathrm{cur}}) \wedge [\mathrm{top}(K_{\mathrm{cur}}) = b].$$

The global sum decides whether the child exists. On descent, every node pushes its parent active flag, candidate/move flag, and observed top bit. Only a candidate pops its key cursor, and active becomes candidate. On return, only a moved node restores the saved bit, after which parent active is popped. The four invariant clauses follow by induction on the DFS stack depth.

Each existing trie node generates one leaf probe and two child probes. Each probe has fixed message latency; each descent and return has a fixed number of bounded stack operations. An absent child costs one probe. Hence a trie on key set X costs $O(1 + \sum_{x \in X} |x|)$. Copying every leaf path has the same sum-of-lengths bound.

D.6 Stage totals

Degree fields and the n degree-class fields take $O(n)$ by Lemma 5. The K_2 trie and dictionary contain at most n keys of length $3n - 1$, hence cost $O(n^2)$. The r color classes form a partition, so all counts appended to K_3 take $O(n)$ synchronized time. The K_3 trie has at most n keys of length $5n - 2$ and costs $O(n^2)$. Third-round multiplicities sum to n. The final code has $O(n^2)$ bits, so all reversal, copying, and construction of the injective integer trace cost $O(n^2)$. In the final phase every node has the same z coordinate; the fixed graph readout selects its average.

D.7 Arbitrary fixed refinement depth

For a fixed requested depth k, repeat the dictionary-and-field stage used above: from the canonical key $K_t(v)$, enumerate the at most n distinct keys by the same trie template and append one unary neighbor count per class. The resulting key is equivalent to $c_{t+1}(v)$. Two reusable dictionaries, field stacks, and path stacks suffice because a stage is complete before the next starts. If L_t is the maximum key length, then $L_{t+1} \le L_t + 2n$ and hence $L_t = O(tn)$. Stage t costs $O(nL_t) = O(tn^2)$ recurrences, so the first k stages cost $O(k^2 n^2) = O_k(n^2)$. Unrolling these k copies produces a finite controller whose dimension and parameters may depend on fixed k but never on n, which is exactly the quantifier in Theorem 5.

E The fixed multiplicative cell

E.1 Power sums as a canonical multiset code

The following lemma isolates the only number-theoretic ingredient in the multiplication-augmented compiler.

Lemma 7 (No-carry power-sum packing)**.** *Let $x_1, \dots, x_m$ be positive integers below R, with $m \le n$, and let*

$$h = 2^{\lceil \log_2(2n) \rceil}, \qquad Q = R^h, \qquad K = nQ + 1.$$

For $S_h(x) = \sum_{j=0}^{h-1} K^{h-1-j} x^j$, the integer $\sum_{i=1}^{m} S_h(x_i)$ uniquely determines the multiset $\{\{x_1, \dots, x_m\}\}$.

Proof. The coefficient of K^{h-1-j} before carrying is $P_j = \sum_i x_i^j$. Since $m \le n$ and $x_i^j < R^h = Q$ for $j < h$, $0 \le P_j < nQ < K$; hence these coefficients are the actual base-K digits. The digit $P_0 = m$. Over the integers, Newton's identities recursively recover the first m elementary symmetric polynomials from $P_1, \dots, P_m$. They determine the monic polynomial $\prod_i (X - x_i)$, including multiplicities. Since $h - 1 \ge n \ge m$, all required power sums are present. □

Suppose at stage j that $0 < r_v < R$ and equal codes mean equal j-round colors across all graphs of order n. Let

$$N_v = \sum_{u \in N(v)} S_h(r_u), \qquad r_v^+ = r_v + RN_v, \qquad B = K^h, \qquad R^+ = R(nB+1).$$

Lemma 7 shows that N_v is equivalent to the multiset of neighbor codes. Reduction modulo R recovers r_v from r_v^+, after which integer division conceptually recovers N_v; the cell itself performs neither operation. Since $S_h(r) < B$, we have $N_v < nB$ and therefore $0 < r_v^+ < R^+$. Induction proves that $r_v^+ = r_w^+$ exactly when the $(j+1)$-round recursive color objects agree. The common bound R^+ is a function of (n, j) alone, so this is cross-graph canonicality, not merely a within-graph relabeling.

E.2 Logarithmic doubling block

For a fixed K and current code r_v, a node maintains

$$p_v = r_v^\ell, \quad b = K^\ell, \quad s_v = S_\ell(r_v), \quad u_v = p_v s_v, \quad z_v = \sum_{w \in N(v)} S_\ell(r_w).$$

Initialization at $\ell = 1$ is

$$(p_v, b, s_v, u_v, z_v) = (r_v, K, 1, r_v, \deg v).$$

One neighbor-sum recurrence supplies $\sum_{w \in N(v)} u_w$ and applies

$$p_v^+ = p_v^2, \qquad b^+ = b^2, \qquad s_v^+ = (b + p_v)s_v,$$
$$u_v^+ = p_v^+ s_v^+, \quad z_v^+ = bz_v + \sum_{w \in N(v)} u_w, \quad \ell^+ = 2\ell.$$

The two identities

$$S_{2\ell}(r) = (K^\ell + r^\ell)S_\ell(r), \qquad \sum_w S_{2\ell}(r_w) = K^\ell \sum_w S_\ell(r_w) + \sum_w r_w^\ell S_\ell(r_w)$$

prove the invariant by induction. Thus $a = \lceil \log_2(2n) \rceil$ doubling steps reach $\ell = h$. An independent preparation block starts from $(\ell, Q) = (1, R)$ and applies $(\ell, Q) \leftarrow (2\ell, Q^2)$ for the same a steps, producing $Q = R^h$.

E.3 Reference transition

Write the 22 coordinates as

$$(n, k, 1, \iota, \pi, \epsilon_0, \epsilon, \kappa, \delta, f, j,$$
$$r, R, \lambda, Q, p, b, s, u, z, \ell, H).$$

The six Greek flags are one-hot phases Init, Prep, EvalInit, Eval, Commit, and Done; f marks the final graph-packing block, j is the completed refinement count, and λ is the current preparation length. The only aggregated coordinates are

$$d_v = \sum_{w \in N(v)} 1, \qquad m_v = \sum_{w \in N(v)} u_w.$$

Starting from $(\iota, n, k, 1) = (1, n, k, 1)$ and all other coordinates zero, the reference transition is:

Init $(r, R, \lambda, Q) \leftarrow (1, 2, 1, 2)$ and enter Prep.

Prep If $\lambda < 2n$, set $(\lambda, Q) \leftarrow (2\lambda, Q^2)$; otherwise enter EvalInit.

EvalInit Set $K = nQ + 1$ and

$$(p, b, s, u, z, \ell, H) \leftarrow (r, K, 1, r, d_v, 1, \lambda),$$

then enter Eval.

Eval If $\ell < H$, apply the doubling equations above using m_v. If $\ell = H$ and $f = 0$, enter Commit; if $\ell = H$ and $f = 1$, enter Done.

COMMIT Set

$$r \leftarrow r + Rz, \quad R \leftarrow R(nb+1), \quad j \leftarrow j+1, \quad \lambda \leftarrow 1, \quad Q \leftarrow R,$$

where the last Q receives the updated R. Set $f = 1$ exactly when the updated j equals k, and enter PREP.

DONE Hold the state fixed.

This piecewise display specifies the transition but does not add conditional gates to the model. For an integer x, define

$$\mathrm{pos}(x) = \mathrm{ReLU}(x) - \mathrm{ReLU}(\mathrm{ReLU}(x) - 1),$$

which equals $\mathbf{1}[x > 0]$. The three tests

$$c_P = \mathrm{pos}(2n - \lambda), \quad c_E = \mathrm{pos}(H - \ell), \quad c_S = \mathrm{pos}(k - j - 1)$$

are Boolean on every reachable state. Multiplying each candidate assignment by its one-hot phase and the relevant test, then adding the candidates, produces a branch-free multiplication-augmented arithmetic circuit over affine, scalar ReLU, and exact binary multiplication gates. For example,

$$\begin{aligned} \pi^+ &= \iota + \pi c_P + \kappa, & \epsilon_0^+ &= \pi(1 - c_P), \\ \epsilon^+ &= \epsilon_0 + \epsilon c_E, & \kappa^+ &= \epsilon(1 - c_E)(1 - f), \\ \delta^+ &= \delta + \epsilon(1 - c_E)f, & f^+ &= \kappa(1 - c_S) + (1 - \kappa)f. \end{aligned}$$

All data coordinates use the same disjoint selectors. The machine-readable expansion in the supplement contains 24 inputs (the state plus d_v, m_v), 22 outputs, 44 affine nodes, six ReLU nodes, 49 multiplication nodes, and one constant node; its total node count is 124 and maximum depth is 11. Its affine coefficients and biases belong to $\{-1, 0, 1, 2\}$, so the cell is fixed and rational independently of n and k.

E.4 CORRECTNESS, RECURRENCE COUNT, AND OUTPUT

The one-hot invariant follows directly from the six phase equations. The preparation and evaluation invariants above then prove by induction on j that, at the start of refinement block j, r_v is canonical for $c_j(v)$ and $0 < r_v < R$. Commit applies exactly the update proved after Lemma 7. After the kth commit, f selects one more packing block without a further commit. In DONE, graph sum exposes

$$\left(\sum_v 1, \sum_v s_v \right) = (n, Y_k(G)).$$

The second coordinate is a no-carry power-sum packing of the final node-code multiset, and therefore two same-order graphs have equal outputs exactly when their CR_k objects agree. The coordinates are integers, so unequal outputs have ℓ_∞ separation at least one.

There is one initial transition. Each packing block uses a preparation squarings, one preparation exit, one evaluation initialization, a doublings, and one evaluation exit, totaling $2a + 3$ recurrences. The first k blocks additionally use one commit each; the final block enters DONE directly. Hence

$$1 + (k+1)(2a+3) + k = 2(k+1)\lceil \log_2(2n) \rceil + 4k + 4.$$

For the bit bound, let $L_j = \lceil \log_2 R_j \rceil$. Since $h < 4n$,

$$\begin{aligned} \log_2 Q &= hL_j, & \log_2 K &\le \log_2(2n) + hL_j, \\ \log_2 B &= h \log_2 K, & L_{j+1} &\le (1 + 16n^2)L_j + 4n \log_2(2n). \end{aligned}$$

With $R_0 = 2$, induction gives $L_j = n^{O(j)}$. Neighbor sums add at most $\lceil \log_2 n \rceil$ bits. Every inactive candidate is still evaluated, but a constant-depth multiplication-augmented arithmetic DAG increases bit length by only a constant factor per transition. Thus all state, aggregate, and internal circuit values use $n^{O(k+1)}$ bits.

E.5 STABLE REFINEMENT LOWER BOUND

Color refinement stabilizes after at most $n - 1$ strict partition refinements, so the input $k = n$ computes the stable graph-level invariant in $O(n \log n)$ recurrences.

Let P be the partition at round n. It is stable, and a canonical round-n color contains its round-$(n-1)$ ancestor together with the multiset of neighboring ancestor colors. Because the stable round-n colors relabel the same classes injectively, this data determines both every class size and every interclass neighbor count—the stable quotient. Hence equal same-order CR_n histograms identify equal stable quotients, their next-round histograms are equal, and induction gives equality at all later rounds. Conversely, unequal CR_n histograms cannot later merge because each recursive color retains its ancestor. Therefore the stable graph-level invariant is exactly determined at round n, not merely the node partition of each graph separately.

Lemma 8 (Global-broadcast transfer)**.** *For two same-order graphs with equal graph-level CR_T objects, every fixed permutation-equivariant recurrent message-passing cell with neighbor sum and recurrent global sum has equal graph-level outputs through recurrence T.*

Proof. Without global sum, induction makes each node state after t recurrences a function of $c_t(v)$ and the shared inputs. With global sum, equal CR_T objects project to equal graph-level histograms at every earlier round. Inductively, the globally summed state and its broadcast are therefore equal in both executions at every step, and node states remain functions of their current recursive color and the common broadcast history. Graph readout is thus equal through recurrence T. □

For the lower bound, Krebs & Verbitsky (2015) construct same-order graph pairs that are eventually distinguished but remain equivalent in two-variable counting logic for $n - O(\sqrt{n})$ quantifier depth. Under the standard correspondence with color refinement, their graph-level color multisets agree for the same number of rounds up to an additive convention constant. Lemma 8 therefore implies that no such message-passing cell, regardless of arithmetic basis, can compute the stable invariant in $o(n)$ recurrences.

E.6 CERTIFICATE CHECKS

The certificate regenerates the arithmetic DAG from named operations and checks it against the reference transition on representative graphs and all six phases: 1,425 node transitions and 626 unique circuit inputs, with zero mismatches. An independent CR implementation checks the graph-level equivalence relation on all 1,252 positive-order atlas graphs, together with exact larger-integer stress cases.

F VALIDATION SUMMARY

Table 4: Exact all-atlas class counts.

n	graphs	CR_1	CR_2	CR_3	max $\lvert K_2 \rvert$
1	1	1	1	1	2
2	2	2	2	2	5
3	4	4	4	4	8
4	11	11	11	11	11
5	34	31	34	34	14
6	156	102	152	152	17
7	1,044	342	952	1,018	20

All threshold creation strings through $n = 12$ have distinct degree histograms. Enumerating regular matrices through $m = 6$ covers 297,200 matrices at the largest order and yields 33,692 distinct CR_2 classes; the largest observed fiber is 32, below the analytic bound 768. A separate reverse implementation reconstructs every fiber exactly through $m = 5$, and universal-vertex padding preserves every tested partition.

The canonical ReLU compiler and a direct recursive CR_3 implementation induce identical partitions for every $n \leq 7$. Microstep checks cover 65 unary countdowns, 1,554 delayed multi-node field schedules, 511 preserving stack copies, 2,044 DFS history restorations, 1,154 integer flags, and 8,190 stack cases. Seven graph families at $n \in \{8, 16, 32, 64\}$ satisfy the proved key, synchronization, and serialization bounds. A fixed two-layer rational CPWA (ReLU-MLP) cell executed for up to eight recurrences on every atlas graph yields 152,550 exact coordinates with no denominator, magnitude, or bit-length violation. The integer-trace control exhausts all 2,046 nonempty binary strings through length ten and verifies unit-margin partition equivalence and relabeling invariance on the atlas.

For the multiplication-augmented compiler, an independent verifier records 3,886,707 structural pair assertions, 39 exact compiler runs, 133 canonical composite-code cases, 2,548 phase invariants, 6,420 generic packing cases, and 130 exact graph-pair checks. The largest inspected integer has 18,810,308 bits. The separately generated circuit certificate has 124 nodes, depth 11, 24 inputs, and 22 outputs; tests exercise 1,425 one-step transitions, 626 unique selector inputs, and all six phases, with no failures.