LOTUS: A LOGIC-BASED OBJECT-CENTRIC TOPOLOGICAL UNIFIED SOLVER

Anonymous authors
Paper under double-blind review

ABSTRACT

In bipartite graph constraint satisfaction problems such as channel error-correction decoding, existing graph neural networks (GNNs) and belief propagation (BP) algorithms suffer from severe error floors caused by trapping set deadlocks induced by overlapping short cycles. To address this bottleneck, we propose **Lotus** (Logic-based Object-centric Topological Unified Solver), a neuro-algebraic framework that integrates deep unfolded networks with local discrete algebraic elimination across a Base GNN, an Expert GNN, and an algebraic elimination module termed **xLoop**. Lotus is structured around three core mechanisms: (i) an isotropic Variable Module and parameter-free Check Module devoid of positional encodings, avoiding graph-specific eigenspectrum overfitting to preserve permutation equivariance and enable zero-shot length and topology generalization; (ii) closed-form forward analytical gradients derived from a soft parity potential function, paired with Targeted Constraint Relaxation (TCR) and dual dynamic gating to directly steer forward trajectories without test-time optimization; and (iii) the xLoop module, which pairs weighted Laplacian spectral bisection with streaming Gaussian elimination (SGE) and Local Maximum Likelihood (Local ML) selection to algebraically resolve long-tail trapping sets within size-capped candidate subgraphs. Evaluated on 3GPP 5G New Radio (NR) and regular quasi-cyclic LDPC (QC-LDPC) matrices, Lotus achieves a 75.0%–83.9% hard-instance recovery rate, suppresses suboptimal convergence (SC) by approximately 50%, approaches the near-ML empirical performance bound of ordered statistics decoding (OSD-3) (with BP+xLoop also achieving near-ML performance), and preserves an asymptotic linear latency scaling of $\mathcal{O}(N)$ that operates $15\times \sim 41\times$ faster than OSD-3.

1 INTRODUCTION

Integrating deep learning and graph neural networks (GNNs) into combinatorial constraint systems is an active area of research in **neural algorithmic reasoning** (Veličković et al., 2021; 2022), spanning routing (Kool et al., 2019; Kwon et al., 2020), Boolean satisfiability (SAT) (Selsam et al., 2019), combinatorial graph cuts (Khalil et al., 2017; Sun & Yang, 2023), and low-density parity-check (LDPC) decoding over bipartite graphs (Nachmani et al., 2016; 2018; Choukroun & Wolf, 2022). In these systems, variable nodes must reach global consensus under discrete parity-check constraints. However, standard continuous deep learning paradigms encounter severe theoretical and computational bottlenecks when resolving discrete hard parity constraints.

Hard Parity Constraints, Trapping Sets, and the Complexity Trilemma. Solving discrete hard constraints over bipartite graphs diverges from standard continuous representation learning. Parity-check equations operate over the Galois field $\mathbb{F}_2$ via strict modulo-2 XOR operations, where perturbing even a single variable node flips multiple incident checks, cascading conflict signals across the graph. Continuous relaxations typically yield fractional solutions (continuous relaxation ambiguities), failing to inherently satisfy discrete parity constraints. In particular, overlapping short cycles form (a, b) trapping sets (TS) (Richardson, 2003; Vasic et al., 2009), where erroneous variable nodes lock into cyclic positive feedback loops; weak error-correction signals from peripheral unsatisfied checks (USCs) cannot overcome this deadlock, causing belief propagation (BP) and deep unfolded networks to exhibit an error floor in frame error rate (FER) curves. While global near-maximum-likelihood (near-ML) decoders like order-3 ordered statistics decoding (OSD-3) (Fossorier & Lin, 1995) resolve trapping sets via Gaussian elimination (GE), their cubic complexity ($\mathcal{O}(K^3)$) incurs millisecond-scale

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.

latency on long codes, exceeding ultra-low-latency budgets. Resolving this tension suggests a hybrid neuro-algebraic formulation: leveraging continuous neural dynamics to pinpoint trapping sets while confining finite-field elimination to compact, budget-capped local subgraphs to preserve a constant per-elimination overhead ceiling.

Forward Parity Anchoring and Permutation Equivariance. Deep unfolded networks and graph neural decoders (Nachmani et al., 2016; Choukroun & Wolf, 2022) primarily operate as empirical black-box mappings. Relying on backpropagating loss gradients from output layers, intermediate representations lack instantaneous, mathematically grounded parity constraint anchoring during forward inference, allowing latent trajectories to drift away from the valid code space. On hard instances, trapped variable nodes receive inadequate supervisory signaling, frequently resulting in non-convergence (NC) or settling into Suboptimal Convergence (SC) states where all parity checks are satisfied yet the decoded candidate remains metric-inferior to the transmitted codeword. Simultaneously, incorporating positional encodings (Dwivedi et al., 2020) or Laplacian eigenvectors (Kreuzer et al., 2021) induces rigid structural dimension binding: variable nodes memorize specific column indices, preventing models from transferring across varying block lengths (e.g., 5G NR QC-LDPC lifting). In contrast, while prior decoders rely on positional encodings to break 1-WL symmetries—overfitting to graph eigenspectra—Lotus delegates topological symmetry breaking to downstream algebraic elimination. This PE-free decoupling preserves graph permutation equivariance, enabling zero-shot length and topology generalization.

Related Work. Neural algorithmic reasoning explores algorithmic execution using neural networks on SAT, TSP, and VRP (Veličković et al., 2021; 2022; Selsam et al., 2019; Sun & Yang, 2023; Kwon et al., 2020). However, continuous relaxations fail to resolve discrete $\mathbb{F}_2$ parity checks. In channel decoding, deep unfolded BP (Nachmani et al., 2016; 2018) and Error Correction Code Transformer (ECCT) (Choukroun & Wolf, 2022) break graph isomorphism via edge weights or positional encodings, suffering from over-smoothing collapse (Rusch et al., 2023) and precluding zero-shot extrapolation. While near-ML decoders like OSD-3 (Fossorier & Lin, 1995) break trapping sets, their cubic complexity ($\mathcal{O}(K^3)$) is prohibitive for real-time baseband processing. (See Appendix A and Table 2 for an extensive cross-paradigm taxonomy and theoretical comparison.)

1.1 Core Contributions

To overcome these foundational challenges, we propose **Lotus**, a neuro-algebraic bipartite graph solver cascading a Stage-1 Base GNN, a Stage-2 Expert GNN, and a Stage-3 local algebraic elimination module (xLoop):

- **Permutation-Equivariant Decoupled Architecture:** A parameter-shared Variable Module and parameter-free Check Module devoid of positional encodings preserve graph permutation equivariance, enabling zero-shot OOD extrapolation across scales and topologies.
- **Forward Analytical Gradients with Dynamic Gating:** Closed-form forward analytical gradients derived from a soft parity potential function, coupled with TCR and dual dynamic gating, anchor latent trajectories via instantaneous algebraic drift without test-time autograd, suppressing non-convergence.
- **xLoop Spectral Bisection and Local Algebraic Elimination:** Localized 3-hop subgraph extraction anchored at unsatisfied checks, paired with Laplacian spectral bisection and Local ML selection, algebraically resolves localized trapping sets under a bounded local elimination ceiling.
- **Near-ML Performance and Linear Complexity Scaling:** Across 3GPP 5G NR and regular QC-LDPC codes, Lotus achieves a 75.0%–83.9% hard-instance recovery rate, halves suboptimal convergence, and matches near-ML OSD-3 bounds (with proposed BP+xLoop also achieving near-ML performance) with linear $\mathcal{O}(N)$ latency scaling.

2 Problem Formulation and Bipartite Graph Preliminaries

Bipartite Graph Formulation and Channel Model. Parity-check decoding is formulated over a bipartite graph $G = (V, C, E)$, where $V = \{v_1, \ldots, v_N\}$ denotes variable nodes corresponding to codeword $\mathbf{x} \in \mathbb{F}_2^N$; $C = \{c_1, \ldots, c_M\}$ denotes check nodes corresponding to parity constraints $\mathbf{H} \in \mathbb{F}_2^{M \times N}$; and $(v_j, c_m) \in E \iff H_{m,j} = 1$. A vector $\mathbf{x}$ is a valid codeword ($\mathbf{x} \in \mathcal{C}_{\text{code}}$) if and

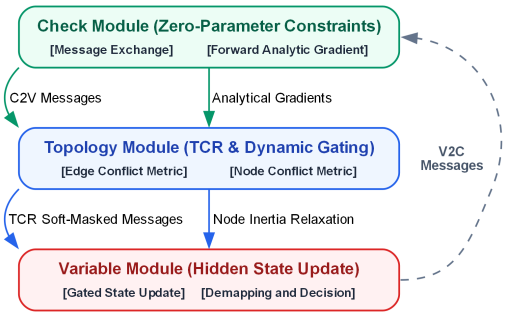


Figure 1: Lotus tri-module closed-loop dynamical architecture.

only if global parity checks are satisfied. Over an additive white Gaussian noise (AWGN) channel with noise $n_i \sim \mathcal{N}(0, \sigma_{\text{ch}}^2)$, observations $y_i = (-1)^{x_i} + n_i$ yield invariant channel log-likelihood ratios (LLRs):

$$\mathbf{H}\mathbf{x}^T \equiv \mathbf{0} \pmod 2, \qquad L_{\text{ch},i} = \ln \frac{\mathbb{P}(y_i \mid x_i = 0)}{\mathbb{P}(y_i \mid x_i = 1)} = \frac{2y_i}{\sigma_{\text{ch}}^2}, \tag{1}$$

while prior LLRs vanish identically under uniform sources: $L_{A,i} \equiv 0$.

Qualitative Taxonomy of Checks and Decoding States. Parity checks are partitioned into three microscopic states: **Unsatisfied Checks (USCs)** ($s_m = 1$, index set $\mathcal{C}_{\text{USC}}$) denoting explicit violations; **True Satisfied Checks (TSCs)** ($s_m = 0$) where all incident variables are correct; and **Pseudo-Satisfied Checks (PSCs)** ($s_m = 0$) whose incident variables contain an even number of bit errors, masking error signals and catalyzing trapping set deadlocks. Given syndrome $\mathbf{s} = \mathbf{H}\hat{\mathbf{x}}^T \pmod 2$ and channel correlation metric $\Lambda(\mathbf{x}) = \sum_{i=1}^{N}(1 - 2x_i)y_i$, terminal outputs partition into five mutually exclusive states: **True Convergence (TC)** ($\mathbf{s} = \mathbf{0}, \hat{\mathbf{x}} = \mathbf{x}_{\text{send}}$); **ML Convergence (MLC)** ($\mathbf{s} = \mathbf{0}, \Lambda(\hat{\mathbf{x}}) > \Lambda(\mathbf{x}_{\text{send}})$), the information-theoretic ML limit treated as successful decoding; **Suboptimal Convergence (SC)** ($\mathbf{s} = \mathbf{0}, \Lambda(\hat{\mathbf{x}}) < \Lambda(\mathbf{x}_{\text{send}})$), converging to an inferior valid codeword; **Systematic-Bits-Correct (Sys-OK)** ($\mathbf{s} \neq \mathbf{0}$, but all systematic bits are correct); and **Non-Convergence (NC)** ($\mathbf{s} \neq \mathbf{0}$ due to trapping set stagnation). (For formal criteria and metric inequalities, see Appendix B.)

3 Lotus Neuro-Symbolic Decoupled Architecture and Latent Dynamics

3.1 Overview of the Decoupled Tripartite Architecture

Lotus preserves the unweighted bipartite structure of classical Tanner graphs (Tanner, 1981), where edges capture adjacency between variable and check nodes without learnable multiplicative weights. Departing from deep unfolded decoders that assign individualized edge weights (Nachmani et al., 2016; 2018), Lotus establishes an object-centric, neuro-symbolic decoupled computing architecture comprising three coordinated modules: (i) the **Variable Module** centralizes all learnable parameters to iteratively update variable node representations; (ii) the **Check Module** is parameter-free, executing parity checks, bidirectional message passing, and forward analytical gradient evaluation; and (iii) the **Topology Module** introduces scale-invariant dynamical gating parameters to dispatch TCR and state update gates.

During each forward iteration, the three modules execute a closed-loop dynamical interaction (Figure 1): check nodes aggregate extrinsic check messages and evaluate forward analytical gradients $\mathbf{g}_{\text{fwd}}$ from a soft parity potential function; the Topology Module quantifies edge- and node-level constraint conflicts to synthesize TCR soft masks and dual dynamic gates; and the Variable Module aggregates TCR-modulated check messages and forward gradients to drive latent state transitions, demapping them into soft decisions.

3.2 VARIABLE MODULE: POSITIONAL-ENCODING-FREE DESIGN AND LATENT STATE UPDATES

The Variable Module represents the variable node set V, governing latent representation learning, memory retention, and iterative state updates.

Positional-Encoding-Free Feature Embedding. Variable nodes omit positional encodings (Dwivedi et al., 2020) and remain isotropic, guaranteeing permutation equivariance and size-invariance (Kreuzer et al., 2021) for zero-shot OOD extrapolation. Each variable node projects physical observations $\mathbf{feature}_i = [L_{\text{ch},i}, |L_{\text{ch},i}|, \sigma^2_{\text{ch}}]^T \in \mathbb{R}^3$ into latent space via $V_{\text{channel},i} = \mathbf{W}_{\text{in}}\mathbf{feature}_i + \mathbf{b}_{\text{in}} \in \mathbb{R}^{D_{\text{latent}}}$. Latent states are initialized via a 1-hop round-trip prior check aggregation: $h_i^{(0)} = V_{\text{channel},i} + \text{Linear}_{\text{check}}(V_{\text{check},i})$ (see Appendix C.1).

Latent State Updates and Demapping. All learnable weights are centralized within the Variable Module and shared across unfolded layers. At iteration ℓ, shared multilayer perceptron (MLP) $\text{MLP}_{\text{share}}$ generates update increment $\Delta h_i^{(\ell)}$, modulated by dual dynamic gates from the Topology Module:

$$\Delta h_i^{(\ell)} = \text{MLP}_{\text{share}}\left(\left[h_i^{(\ell-1)} \,\|\, \sum_{c\in\mathcal{N}(i)} \text{Mask}_{c\to i}^{(\ell)} \cdot m_{c\to i}^{(\ell)} \,\|\, V_{\text{channel},i} \,\|\, \mathbf{g}_{\text{fwd},i}^{(\ell)}\right]\right), \tag{2}$$

$$h_i^{(\ell)} = \alpha_{\text{atten},i}^{(\ell)} \cdot h_i^{(\ell-1)} + \alpha_{\text{amp},i}^{(\ell)} \cdot \Delta h_i^{(\ell)}. \tag{3}$$

A linear readout layer demaps $h_i^{(\ell)}$ into posterior LLR $L_{\text{post},i}^{(\ell)} = \mathbf{w}_{\text{out}}^T h_i^{(\ell)} + b_{\text{out}}$, soft decision $\tilde{x}_i^{(\ell)} = \tanh(L_{\text{post},i}^{(\ell)}/2) \in [-1, 1]$, and hard decision $\hat{x}_i^{(\ell)} = \frac{1-\text{sign}(L_{\text{post},i}^{(\ell)})}{2} \in \{0, 1\}$.

3.3 CHECK MODULE: ZERO-PARAMETER PARITY CONSTRAINTS AND FORWARD ANALYTICAL GRADIENTS

The Check Module corresponds to check nodes C, executing message passing, parity checks, and forward analytical gradient evaluation without learnable parameters.

Zero-Parameter Parity Checks and Extrinsic Messages. Layer-wise hard syndromes $\mathbf{s}^{(\ell)} = \mathbf{H}(\hat{\mathbf{x}}^{(\ell)})^T \pmod 2 \in \{0, 1\}^M$ map to bipolar target states $s_m = (-1)^{s_m^{\mathbb{F}_2}} \in \{-1, +1\}$; $\mathbf{s}^{(\ell)} \equiv \mathbf{0}$ triggers immediate early stopping. Punctured extrinsic messages $m_{i\to c}^{(\ell)} = L_{\text{ch},i} + \sum_{c'\in\mathcal{N}(i)\setminus\{c\}} m_{c'\to i}^{(\ell-1)}$ map across incident checks:

$$m_{c\to i}^{(\ell)} = 2\tanh^{-1}\left(s_c \prod_{j\in\mathcal{N}(c)\setminus\{i\}} \tanh\left(\frac{m_{j\to c}^{(\ell)}}{2}\right)\right). \tag{4}$$

(For sign-magnitude log-domain decomposition, see Appendix C.2.)

Relaxed Residuals and Forward Analytical Gradients. The Check Module defines continuous parity residuals $F_m(\tilde{\mathbf{x}}^{(\ell-1)}) = 1 - s_m \prod_{j\in\mathcal{N}(c_m)} \tilde{x}_j^{(\ell-1)}$ and soft potential $\mathcal{E}(\tilde{\mathbf{x}}^{(\ell-1)}) = \sum_{m\in C}[F_m(\tilde{\mathbf{x}}^{(\ell-1)})]^2$. Letting $Q_{m,i} = \prod_{j\in\mathcal{N}(c_m)\setminus\{i\}} \tilde{x}_j^{(\ell-1)}$ denote the punctured product , the forward analytical gradient along steepest descent of $\mathcal{E}$ evaluates in closed form:

$$\mathbf{g}_{\text{fwd},i}^{(\ell)} = -\frac{\partial\mathcal{E}}{\partial\tilde{x}_i^{(\ell-1)}} = \sum_{m\in\mathcal{N}(i)} 2s_m F_m(\tilde{\mathbf{x}}^{(\ell-1)}) Q_{m,i}, \tag{5}$$

acting as an algebraic drift force (akin to continuous Langevin dynamics) guiding latent states toward valid code space in closed form without test-time automatic differentiation (see Appendix D).

3.4 TOPOLOGY MODULE: TARGETED CONSTRAINT RELAXATION AND DYNAMIC GATING

Governed by degree-normalized, topology-intrinsic control metrics, the Topology Module coordinates spatiotemporal TCR, functioning analogously to dynamic Lagrangian relaxation. Evaluating

edge conflict $\text{conf}_{c\to i}^{(\ell)} = \max(0, -\tilde{x}_i^{(\ell-1)} \cdot \tanh(\text{MLP}_{\text{edge}}(m_{c\to i}^{(\ell)})))$, the spatial TCR soft mask exponentially dampens conflicting edges:

$$\text{Mask}_{c\to i}^{(\ell)} = (1 - b_\ell) \exp\left(-\beta_\ell \cdot \text{conf}_{c\to i}^{(\ell)}\right) + b_\ell, \tag{6}$$

where $b_\ell \in [0, 1]$ serves as the baseline leakage floor and β_ℓ denotes polarity sensitivity. The Topology Module couples mean edge conflicts with gradient conflicts to form composite node conflict metric $\text{conf}_{\text{node},i}^{(\ell)} = (\frac{1}{d_i} \sum_{c\in\mathcal{N}(i)} \text{conf}_{c\to i}^{(\ell)}) \cdot (1.0 + \tanh(\gamma_\ell \cdot \max(0, -L_{\text{post},i}^{(\ell-1)} \cdot \mathbf{g}_{\text{fwd},i}^{(\ell)})))$, driving memory attenuation and increment amplification gates:

$$\alpha_{\text{atten},i}^{(\ell)} = \exp(-\lambda_\ell \cdot \text{conf}_{\text{node},i}^{(\ell)}), \qquad \alpha_{\text{amp},i}^{(\ell)} = 1.0 + \kappa_\ell \cdot \text{conf}_{\text{node},i}^{(\ell)}. \tag{7}$$

In trapping sets ($\text{conf}_{\text{node}} \gg 0$), temporal TCR systematically attenuates historical state inertia while injecting targeted algebraic momentum from parity gradients, autonomously destabilizing symmetric cyclic deadlocks (clipping calibrations in Appendix C.3).

3.5 Multi-Task Training Objectives and Layer-Wise Supervision

Lotus optimizes a layer-wise multi-task objective with normalized weights ($\sum_{\ell=1}^{L} w_\ell = 1.0$):

$$\mathcal{L}_{\text{total}} = \sum_{\ell=1}^{L} w_\ell \cdot \left[\mathcal{L}_{\text{BCE}}^{(\ell)} + \lambda_\ell \mathcal{L}_{\text{check}}^{(\ell)}\right], \tag{8}$$

where $\mathcal{L}_{\text{BCE}}^{(\ell)}$ denotes layer-wise binary cross-entropy (BCE) loss on posterior LLRs and $\mathcal{L}_{\text{check}}^{(\ell)} = \frac{1}{M} \sum_{m\in C} [F_m(\tilde{\mathbf{x}}^{(\ell)})]^2$ penalizes parity residual energy . Lotus employs a two-stage curriculum: **Stage-1 Base GNN** is trained across all signal-to-noise ratios (SNRs) with Kendall homoscedastic uncertainty balancing; **Stage-2 Expert GNN** is trained on non-converged hard instances with progressive loss ratio $\lambda_\ell \in [0.15, 0.40]$ to escape trapping sets (derivations in Appendix E).

4 xLoop: Local Algebraic Elimination and Local Maximum Likelihood Selection

While continuous neural dynamics steer trajectories away from high-entropy regimes, overlapping cycles inevitably induce continuous relaxation ambiguities and localized deadlocks. For persistent non-convergent states, Lotus activates the xLoop co-processor on the best historical decoding state ℓ^*. By synthesizing graph-spectral bisection, streaming Gaussian elimination over $\mathbb{F}_2$, and Local Maximum Likelihood (Local ML) selection, xLoop algebraically breaks localized topological deadlocks and resolves long-tail hard instances (Figure 2).

4.1 Best Historical State Selection and 3-Hop Neighborhood Expansion

To prevent iterative stagnation from inducing representational collapse and hard-decision freezing, the decoder logs the best historical state exhibiting minimum unsatisfied checks: $\ell^* = \arg\min_{\ell \le L_{\text{trunc}}} \text{USC}^{(\ell)}$, where $\text{USC}^{(\ell)} = \|\mathbf{s}^{(\ell)}\|_1$. Anchoring elimination at $\mathbf{x}_{\text{init}} = \hat{\mathbf{x}}^{(\ell^*)}$ with active syndrome checks $\mathcal{C}_{\text{USC}} = \{c_m \mid s_m^{(\ell^*)} \neq 0\}$, xLoop executes a multi-source 3-hop topological expansion $\mathcal{V}_{\text{sub}} = \bigcup_{c\in\mathcal{C}_{\text{USC}}} \mathcal{V}_{\text{3hop}}(c)$ via sparse breadth-first search (BFS) in $\mathcal{O}(1)$ local complexity, subsuming overlapping short cycles. Evaluating variable overlaps ($|\mathcal{V}_{\text{3hop}}(c_a) \cap \mathcal{V}_{\text{3hop}}(c_b)| \ge 1$), xLoop partitions unsatisfied checks into J disconnected clusters $\{\mathcal{C}_j\}_{j=1}^{J}$.

Fast-Bypass Exit. If any cluster exceeds the hardware limit ($\max_j |\mathcal{C}_j| > 16$) or cluster count exceeds capacity ($J > 8$), xLoop triggers an immediate fast-bypass exit to declare non-convergence, avoiding futile computation under diffused noise (see Appendix F.1).

4.2 Weighted Laplacian Spectral Bisection and Variance-Deflated Candidate Selection

For intermediate connected clusters ($4 < |\mathcal{C}_j| \le 16$) representing complex trapping sets, xLoop constructs weighted check-adjacency $A_C(a, b) = |\mathcal{N}(c_a) \cap \mathcal{N}(c_b)|$ and Laplacian $\mathbf{L}_C = \mathbf{D}_C - \mathbf{A}_C$.

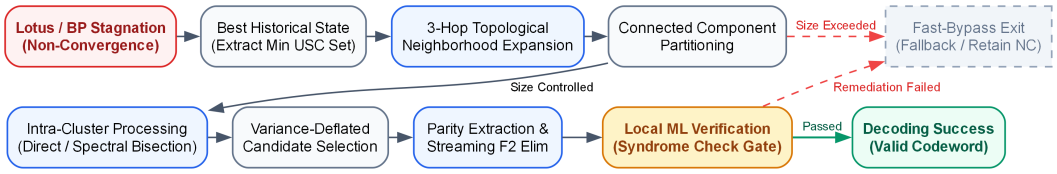


Figure 2: Processing pipeline of the xLoop local algebraic elimination and hard-instance recovery framework.

Using shifted power iteration with mean-deflation, xLoop approximates Fiedler vector $\mathbf{v}_2$ and applies a median split to partition $\mathcal{C}_j$ into balanced sub-clusters $\mathcal{C}_{j,1}$ and $\mathcal{C}_{j,2}$, merging local error vectors across cut-set variables via bitwise OR: $\mathbf{e}_j = \mathbf{e}_{j,1} \vee \mathbf{e}_{j,2}$, bounding single-elimination dimension to constant order $\mathcal{O}(1)$ (see Appendix F.2).

Variance-Deflated Candidate Selection. Confining candidate selection to the local sub-cluster avoids dilution from distant, harmless bits. However, erroneous variables trapped in cyclic topologies often exhibit moderate LLR magnitudes locked into cyclic positive feedback loops, preventing them from being prioritized under naive magnitude sorting. To selectively prioritize these critical error bits without candidate space explosion, xLoop synthesizes an adaptive variance deflation factor coupling latent state dynamical instability $z_{\text{instab},i}$ with microscopic constraint conflict accumulation $z_{\text{micro},i}$:

$$\text{Deflation}_i = 1.0 + 2.0\max(0, z_{\text{instab},i}) + 2.0\max(0, z_{\text{micro},i}), \qquad L_{\text{xloop},i} = \frac{L_{\text{post},i}}{\text{Deflation}_i}. \tag{9}$$

Sorting local variables by deflated magnitudes $|L_{\text{xloop},i}|$ shifts critical trapped error bits to the top, selecting the top- K_{top} variables ($K_{\text{top}} = 32$) as candidate set $\mathcal{V}_{\text{cand}}$ (see Appendix F.3).

4.3 Local Parity-Check Submatrix Extraction and Streaming $\mathbb{F}_2$ Elimination

Extracting columns of $\mathbf{H}$ indexed by $\mathcal{V}_{\text{cand}}$ constructs a massively over-determined submatrix $\mathbf{H}_{\text{sub}} \in \mathbb{F}_2^{M \times K_{\text{top}}}$ retaining all M global check rows. To guarantee zero global syndrome leakage under $\mathcal{O}(1)$ local memory, streaming Gaussian elimination (SGE) sequentially pipelines these check constraints over $\mathbb{F}_2$:

$$\mathbf{H}_{\text{sub}}\mathbf{e} \equiv \mathbf{s} \pmod 2, \tag{10}$$

$$\mathcal{A} = \left\{ \mathbf{e} = \mathbf{e}_0 \oplus \mathbf{B}\mathbf{a} \mid \mathbf{a} \in \mathbb{F}_2^d \right\}. \tag{11}$$

By construction, $\mathbf{H}_{\text{sub}}$ retains all M check rows (satisfied checks impose homogeneous constraints $\mathbf{H}_{\text{sub},m,:}\mathbf{e} \equiv 0$). This guarantees that any valid solution $\mathbf{e} \in \mathcal{A}$ satisfies all global parity checks with zero secondary violations across the rest of the graph. Null space dimension is capped at $d \le d_{\max} = 12$ ($2^d \le 4096$); algebraic contradictions ($0 = 1$) trigger immediate non-convergence exit (see Appendix F.4).

4.4 Local Maximum Likelihood Selection and Pathological Safeguards

Using the channel observation vector $\mathbf{y}$, xLoop evaluates single-bit metric gains $\mathbf{g}[k] = -2(1 - 2\mathbf{x}_{\text{init},\mathcal{V}_{\text{cand}}[k]}) \cdot \mathbf{y}_{\mathcal{V}_{\text{cand}}[k]}$ and relative LLR metric increments $\mathbf{\Gamma} = \mathbf{E}_{\text{cand}}\mathbf{g}$ across all candidates in $\mathcal{A}$ ($\mathbf{E}_{\text{cand}} \in \mathbb{F}_2^{2^d \times K_{\text{top}}}$) , selecting the optimal error candidate via:

$$\mathbf{e}_{\text{opt}} = \arg\max_{\mathbf{e}\in\mathcal{A}} \mathbf{\Gamma}[\mathbf{e}] \iff \hat{\mathbf{x}}_{\text{opt}} = \arg\max_{\mathbf{e}\in\mathcal{A}} \Lambda(\mathbf{x}_{\text{init}} \oplus \text{expand}(\mathbf{e})), \tag{12}$$

which is mathematically equivalent to Local Maximum Likelihood (Local ML) selection over all valid codewords spanned by $\mathcal{A}$.

Reliability Safety Margin and Complexity Bounds. To prevent SC induced by noise bursts, xLoop enforces a reliability safety margin Δ_{margin}: if $\mathbf{\Gamma}[\mathbf{e}_{\text{opt}}] < -\Delta_{\text{margin}}$, the algebraic candidate lacks sufficient waveform likelihood support and is rejected, falling back to $\mathbf{x}_{\text{init}}$. Enforcing an additional maximum flipping weight constraint $w_H(\mathbf{e}) \le w_{\max} = 6$ guarantees constant-bounded computational complexity ($\mathcal{O}(d_v K_{\text{top}}^3)$) suitable for commercial baseband implementation (see Appendix F.5).

4.5 Formal Proposition: Algebraic Solvability and SC Suppression

Proposition 1 (Algebraic Elimination and Suboptimal Convergence Suppression). *Let $\mathcal{V}_{cand}$ denote the candidate variable set ($|\mathcal{V}_{cand}| = K_{top}$) identified via multi-source 3-hop topological expansion and dynamical variance deflation, and let $\mathbf{H}_{sub} \in \mathbb{F}_2^{M \times K_{top}}$ denote the corresponding local submatrix. (i)* ***Deterministic Syndrome Annihilation:*** *If $\mathbf{s} \in \mathrm{span}(\mathbf{H}_{sub})$, there exists an error vector $\mathbf{e} \in \mathbb{F}_2^{K_{top}}$ such that $\hat{\mathbf{x}} = \mathbf{x}_{init} \oplus \mathrm{expand}(\mathbf{e})$ eliminates all parity violations across the entire bipartite graph: $\mathbf{H}\hat{\mathbf{x}}^T \equiv \mathbf{0} \pmod 2$. (ii)* ***Suppression of Suboptimal Convergence:*** *If the system is inconsistent ($0 = 1$), xLoop triggers early termination. Within the generated affine solution space $\mathcal{A}$, Local ML selection guarantees local maximum likelihood optimality among all candidates satisfying global parity checks, suppressing SC and NC.* (Full proofs of Lemma 1 and Lemma 2 in Appendix G.)

4.6 Cascaded Decoding Architectures and BP + xLoop Near-ML Capability

Leveraging the complementary strengths of continuous iterative decoders and discrete algebraic solvers, we formulate two cascaded architectures:

Lotus Neural-Algebraic Pipeline (Scheme A): An end-to-end three-stage cascade tailored for AI-accelerated baseband architectures. The front-end 8-layer Base GNN decodes standard frames; unrecovered frames transition to the 8-layer Expert GNN; persistent trapping set failures activate Stage-3 xLoop, utilizing latent dynamical variance and node conflict metrics for targeted elimination.

Backward-Compatible BP + xLoop Pipeline (Scheme B): Tailored for legacy baseband infrastructure lacking neural hardware, the explicitly proposed `BP-50 + xLoop` algorithm couples standard Belief Propagation with the xLoop co-processor. When BP stalls in trapping sets, xLoop operates directly on raw soft LLR magnitudes, eliminating error floors and achieving near-ML error-correction performance approaching OSD-3 bounds at minimal deployment overhead. (For architectural diagrams, see Figure 4 in Appendix F.6.)

5 Empirical Evaluation

Benchmark Setup and Baselines. We evaluate Lotus and the proposed `BP-50 + xLoop` across 3GPP 5G NR (BG2, H48A/B, H96A/B, H192A/B, $R = 1/2$) and regular QC-LDPC parity-check matrices (R48A/B, R96A/B, R192A/B, $R = 1/2$) with 10^5 Monte Carlo frames per SNR point. Baselines comprise: (1) `Standard BP-50` (Normalized Min-Sum with 50 iterations); (2) `BP-50 + xLoop (Proposed)`, coupling BP-50 with the xLoop co-processor; (3) the standalone 16-layer `Lotus (Stages 1+2)` neural front-end (272k parameters total); and (4) `OSD-3 (Near-ML)` (Fossorier & Lin, 1995) as the near-ML benchmark. Hardware evaluations ran on an NVIDIA L4 GPU (batch size 512) and C/C++ finite-field elimination on CPU.

5.1 FER Performance Benchmark against Baselines

Figure 3 presents the master FER curves across all six benchmark matrices (detailed pointwise logs and cross-paradigm comparisons with ECCT (Choukroun & Wolf, 2022) and W-BP (Nachmani et al., 2018) are tabulated in Appendix H (Tables 4–6) and Appendix A (Table 2)). Three core empirical findings emerge:

(1) Lotus Approaching Near-ML Bounds: Across all evaluated matrices, Lotus closely matches near-ML OSD-3 bounds. On short regular matrices, Lotus slightly outperforms OSD-3 (e.g., on R48B at 6.0 dB, Lotus achieves FER $= 4.00 \times 10^{-5}$ vs. 1.00×10^{-4} for OSD-3, a $2.5\times$ **error reduction**), avoiding OSD-3 search failures caused by noise perturbations in the Most Reliable Basis (MRB).

(2) Near-ML Capability of Proposed BP + xLoop Cascade: As a plug-in co-processor, the explicitly proposed `BP-50 + xLoop` algorithm achieves a $4.0\times$–$11.4\times$ **FER reduction** over standard BP-50, substantially suppressing the BP error floor (e.g., on H96B at 6.0 dB, FER $< 10^{-5}$ with zero observed error frames). Notably, `BP-50 + xLoop` approaches OSD-3 near-ML performance across all codes and even slightly surpasses end-to-end Lotus on certain curves, demonstrating xLoop's powerful, model-agnostic capability in eliminating long-tail trapping sets.

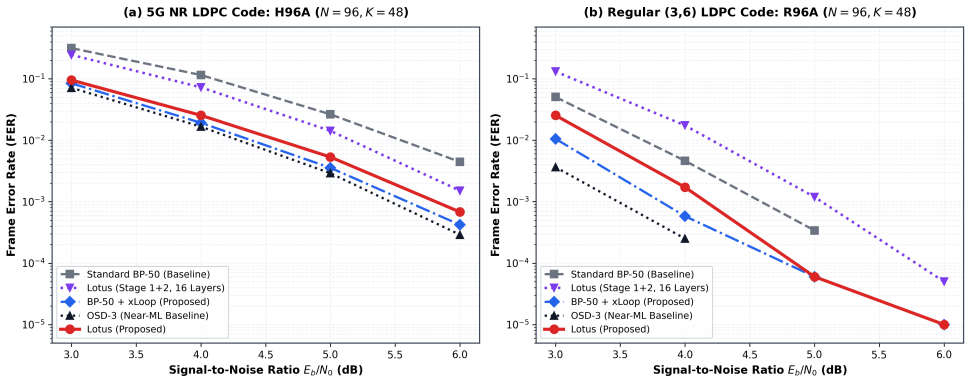


Figure 3: Comprehensive FER vs. SNR performance benchmark across 5G NR and regular QC-LDPC parity-check matrices.

(3) Neural Front-End Representation Quality: Standalone Lotus (Stages 1+2) achieves 50-iteration BP performance at a fraction of the computation. On hard instances, it delivers compact, isolated USC clusters to xLoop, elevating the hard-instance recovery rate to **83.93%** (+18.45% over BP's 65.48%) and suppressing high-SNR FER by orders of magnitude ($46.1\times$ on H48B at 5.0 dB, $106.0\times$ on H96B at 6.0 dB). At FER $\leq 10^{-5}$, data reflect the 10^5- frame statistical resolution floor.

5.2 Complexity Decoupling Analysis

Lotus achieves linear complexity $\mathcal{O}(L_{\text{eff}} \cdot |E|) + \mathcal{O}(1)$ governed by sparse graph message passing, decoupling from the cubic $\mathcal{O}(K^3)$ scaling of OSD-3. Multi-source 3-hop expansion and weighted Laplacian spectral bisection cap candidate degrees of freedom to $d \leq 12$, enforcing an execution ceiling on streaming Gaussian elimination over $\mathbb{F}_2$ with per-call complexity strictly bounded by $\mathcal{O}(12^3) \leq \mathcal{O}(1)$ regardless of global block length N. As block length grows from $N = 48$ to $N = 192$ (Regular Group B at 4.0 dB), OSD-3 per-frame latency increases $12.1\times$ (1.98 ms $\rightarrow$ 23.94 ms), whereas Lotus scales mildly (0.30 ms $\rightarrow$ 0.84 ms), delivering a $28.4\times$**–**$32.5\times$ **latency reduction at** $N = 192$. All 272k learnable weights reside in the Variable Module (vs. 1.5M–9.5M in ECCT (Choukroun & Wolf, 2022)). (Detailed hardware scaling curves and operation breakdowns are provided in Appendix H, Figure 5, and Appendix A, Table 3.)

5.3 Cascaded Early-Stopping and Decoding State Evolution

The hierarchical funneling architecture balances low average latency with robust trapping set elimination: across operational SNRs ($E_b/N_0 \geq 4.0$ dB), the Stage-1 Base GNN achieves an early-exit rate of **94.39%–99.94%** (~ 0.14 ms per frame). Non-converged frames transition to the Stage-2 Expert GNN, and Stage-3 xLoop is activated on merely **0.42%–3.59%** of long-tail frames, statistically explaining sub-millisecond average latencies. Furthermore, xLoop drives NC to **0.00%** across evaluated hard sets, while terminal residual errors split evenly between Suboptimal Convergence (0.85%) and ML errors (0.86%), confirming that decoding is bounded solely by information-theoretic channel limits (see Appendix I, Figure 6).

5.4 Zero-Shot Cross-Scale and Cross-Topology OOD Generalization

Trained exclusively on the shortest 5G matrix (H48A, $N = 48$), Lotus was deployed zero-shot without fine-tuning across $2\times$ (H96A) and $4\times$ block lengths (H192A/R192A) and unseen regular topologies. On H96A, Lotus retains a **75.00%** hard-instance recovery rate (Table 1) and attains a $10.09\times$ FER reduction over BP-50 at 4.0 dB; on H192A and R192A at 6.0 dB, it reaches FER $= 1.37 \times 10^{-3}$ and $< 1.0 \times 10^{-5}$, respectively. This zero-shot transferability stems from the PE-free Variable Module and stateless Check Module, which preserve bipartite graph permutation equivariance and size invariance.

5.5 Ablation Studies and Mechanistic Analysis

Ablation on Error Variable Identification Mechanisms. While xLoop operates on raw LLRs when paired with BP, multi-feature joint deflation (equation 9) in Lotus substantially refines candidate selection. As ablated in Table 7 (Appendix I), naive magnitude ranking misidentifies trapped error bits, achieving only 31.2% Top-4 accuracy. Incorporating dynamical instability and node conflict metrics elevates Top-4 accuracy to **74.7%** (Top-8 to **82.3%**, a $1.4\times$ improvement), elevating the hard-instance recovery rate to **83.93%** and cutting suboptimal convergence by half (Table 1).

Spatial Geometric Alignment and Anti-Over-Smoothing Mechanism. Tracking gradient trajectories reveals that satisfied parity checks progressively silence gradients on correct nodes, contracting the directional angle between forward analytical gradients and global backward gradients from 88.5° (layer 1) to 57° at layer 8 (cosine similarity leaping to -0.540, $p < 10^{-4}$). Concurrently, average inter-node latent cosine similarity increases only from 0.05 to 0.19 at layer 8, proving that forward analytical gradients effectively prevent over-smoothing and preserve discriminative dynamical features for downstream xLoop elimination (derivations in Appendix D).

6 Conclusion and Future Work

Summary of Theoretical and Engineering Contributions. We presented Lotus, a unified neuro-algebraic framework reconciling continuous message passing and discrete $\mathbb{F}_2$ hard constraints. By coupling a permutation-equivariant, PE-free GNN guided by forward analytical gradients with xLoop bounded local algebraic elimination, Lotus eliminates trapping set deadlocks, eliminates non-convergence (0.00% NC in empirical trials), and achieves a 75.0%–83.9% hard-instance recovery rate. In addition, the proposed plug-in `BP + xLoop` co-processor eliminates classical BP error floors and approaches OSD-3 near-ML bounds, corroborating the universal utility of localized algebraic elimination. Lotus maintains linear $\mathcal{O}(L_{\text{eff}} \cdot |E|) + \mathcal{O}(1)$ complexity, achieving $28.4\times$– $32.5\times$ latency reductions over OSD-3 at $N = 192$ alongside zero-shot cross-scale and cross-topology generalization.

Limitations and Future Work. The current 3-hop expansion targets sparse bipartite graphs; on dense constraint graphs or SAT instances, topological neighborhoods may expand rapidly. Future inquiries will investigate multi-scale graph coarsening and tensor decomposition to handle dense constraints. Additionally, we plan to implement low-precision (INT8/INT4) fixed-point quantization on dedicated FPGA/ASIC streaming pipelines for sub-microsecond 6G basebands, while extending neuro-algebraic solving to discrete combinatorial domains such as VLSI chip routing and hardware formal verification.

Table 1: Typical hard-instance recovery and suboptimal convergence suppression on 5G NR matrices.

Matrix	SNR	Decoding Scheme	Hard-Instance Recovery Rate	Suboptimal Convergence (SC)	Final FER
H96A	4.0 dB	BP-50 Baseline	—	73 frames (37.2%)	3.67×10^{-1}
		BP + xLoop (Proposed)	52.55%	58 frames (29.6%)	2.85×10^{-1}
		Lotus (Proposed)	**75.00%**	**37 frames (18.9%)**	$\mathbf{1.95 \times 10^{-1}}$ (1.88× red.)
H48A	3.5 dB	BP-50 Baseline	—	54 frames (32.1%)	2.38×10^{-1}
		BP + xLoop (Proposed)	65.48%	47 frames (28.0%)	2.60×10^{-1}
		Lotus (Proposed)	**83.93%**	**23 frames (13.7%)**	$\mathbf{1.17 \times 10^{-1}}$ (2.03× red.)

AI USE STATEMENT

The authors utilized large language models (specifically Google Gemini) as an interactive assistant for: (1) natural language translation, grammatical refinement, and manuscript polishing; and (2) code scaffolding, syntax debugging, and algorithmic implementation assistance during experimental execution. All underlying theoretical models, core algorithmic concepts (Lotus & xLoop), experimental designs, and data interpretations were conceived and verified by the authors, who take full academic responsibility for the integrity of this work.

ETHICS STATEMENT

This work investigates foundational error-correcting decoding and combinatorial constraint satisfaction on bipartite graphs. It does not involve human subjects, personal data collection, discriminatory bias, or potential negative societal harms. The authors foresee no direct ethical concerns arising from this research.

REPRODUCIBILITY STATEMENT

To ensure complete reproducibility: (1) Comprehensive algorithmic workflows, pseudo-code routines, and theoretical specifications for both the neuro-symbolic GNN and xLoop streaming elimination are exhaustively detailed in Section 3, Section 4, and Appendix F; (2) Complete mathematical proofs for all propositions and lemmas are detailed in Appendix G and Appendix F; (3) Full network hyperparameter settings and multi-stage loss schedules are documented in Appendix E and Appendix I; and (4) Comprehensive pointwise benchmark results and runtime logs across all tested SNR regimes are tabulated in Appendix H.

REFERENCES

Emmanuel J. Candès and Terence Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

Yoni Choukroun and Lior Wolf. Error correction code transformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 21110–21123, 2022.

Vijay Prakash Dwivedi, Chaitanya K. Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*, 2020.

Marc P. C. Fossorier and Shu Lin. Soft-decision decoding of linear block codes based on ordered statistics. *IEEE Transactions on Information Theory*, 41(5):1379–1396, 1995.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7482–7491, 2018.

Elias B. Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

Ralf Koetter and Pascal O. Vontobel. Graph covers and iterative decoding on factor graphs. In *Proceedings of the International Conference on Quantum, Stochastic and Combinatorial Methods for Information Processing*, 2003.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations (ICLR)*, 2019.

Devin Kreuzer, Dominique Beaini, William L. Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph Transformers with spectral feature representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 21618–21629, 2021.

Yeong-Dae Kwon, Jinho Choo, Byoungjip Kim, Iljoo Yoon, Youngjune Gwon, and Seungjai Min. POMO: Policy optimization with multiple options for combinatorial optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 21188–21198, 2020.

Eliya Nachmani, Yona Be'ery, and David Burshtein. Learning to decode linear codes using deep learning. In *Proceedings of the 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 341–346, 2016.

Eliya Nachmani, Elad Marciano, Loren Lugosch, Warren J. Gross, David Burshtein, and Yona Be'ery. Deep learning methods for improved decoding of linear codes. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):119–131, 2018.

Tom Richardson. Error floors of LDPC codes. In *Proceedings of the 41st Annual Allerton Conference on Communication, Control, and Computing*, pp. 1426–1435, 2003.

T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023.

Daniel Selsam, Matthew Lamm, Benedikt Bünz, Percy Liang, Leonardo de Moura, and David L. Dill. Learning a SAT solver from single-bit supervision. In *International Conference on Learning Representations (ICLR)*, 2019.

Haoran Sun and Yiming Yang. DIFUSCO: Graph-based diffusion solvers for combinatorial optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023.

R. Michael Tanner. A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, 27(5):533–547, 1981.

Bane Vasic, Shiva Kumar Chilappagari, Dung Viet Nguyen, and Shashi Kiran Planjery. Trapping sets in LDPC codes. In *2009 IEEE Information Theory Workshop*, pp. 1–5, 2009.

Petar Veličković, Rex Ying, Matilde Padovano, Raia Hadsell, and Charles Blundell. Neural algorithmic reasoning. *Patterns*, 2(7):100273, 2021.

Petar Veličković, Adrià Puigdomènech Badia, David Budden, Razvan Pascanu, Andrea Banino, Misha Dashevskiy, Raia Hadsell, and Charles Blundell. The CLRS algorithmic reasoning benchmark. In *International Conference on Machine Learning (ICML)*, pp. 22084–22102, 2022.

Yisi Yuan, Pepijn Scheepers, Lucas Tasiou, Yunus Can Gültekin, Federico Corradi, and Alex Alvarado. On the design and performance of machine learning based error correcting decoders. In *Proceedings of the 14th International ITG Conference on Systems, Communications and Coding (SCC)*, pp. 1–6, 2025. arXiv:2410.15899.

Table 2: Comprehensive comparison across mainstream neural decoders, algebraic benchmark OSD-3, and Lotus.

Evaluation Dimension	BP-50 (Gallager / MacKay)	W-BP / NOMS (Nachmani et al., 2016; 2018)	ECCT (Choukroun & Wolf, 2022)	OSD-3 (Fossorier & Lin, 1995)	Lotus (Proposed) (Three-Stage Cascade)
Underlying Mechanism	Factor graph message passing	Tanner graph edge-weighted unfolding	Fully connected self-attention + syndrome masking	MRB GE + TEPs	Topological continuous dynamics + Local SGE
Model Parameter Scale	Parameter-free	∼25k–60k (edge-weight tied)	1.5M–9.5M (full self-attention)	Parameter-free	272k (lightweight graph parameterization)
Cross-Scale OOD Transfer	Size-invariant inference	Restricted, weights tied to graph topology	Restricted, relies on sequential positional encodings	Native, arbitrary linear block codes	Native (permutation-equivariant, size-invariant)
Waterfall Gain (Low SNR)	0.0 dB (baseline reference)	∼0.2 to 0.4 dB	∼0.3 to 0.6 dB	Approaching ML theoretical bound	Near OSD-3 (within ~ 0.15 dB gap)
High-SNR Error Floor	Trapping set limited	Trapping set limited, prominent	Lacks algebraic projection, residual floor remains	Essentially floor-free	suppressed (∼80% hard-instance repair rate)
5G H96A @ 6.0 dB FER	4.77×10^{-3}	$\approx 3.5 \times 10^{-3}$	$\approx 1.0 \times 10^{-3}$	$< 1.0 \times 10^{-5}$	$< 1.0 \times 10^{-5}$ (cascaded elimination)
Parity Constraint Enforcement	Iterative likelihood estimation (no algebraic guarantee)	Edge-weighted soft damping/smoothing (no algebraic guarantee)	Continuous cross-entropy implicit penalty (no algebraic guarantee)	Finite-field elimination (enforces $\mathbf{Hx}^T \equiv \mathbf{0}$)	Neural continuous screening + Local affine elimination (enforces $\mathbf{Hx}^T \equiv \mathbf{0}$)
Forward Asymptotic Complexity	$\mathcal{O}(\lvert E\rvert \cdot I)$	$\mathcal{O}(\lvert E\rvert \cdot I)$	$\mathcal{O}(N^2 \cdot L)$	$\mathcal{O}(K^3 + \lvert\mathcal{T}_{\text{TEP}}\rvert)$	$\mathcal{O}(L(N \cdot D^2 + \lvert E\rvert \cdot D)) + \mathcal{O}(d_v K_{\text{top}}^3)$
Per-Frame Inference Latency	0.12–0.45 ms	0.35–0.82 ms	1.20–4.50 ms	4.8–25.5 ms	0.14–0.49 ms
Trapping Set Mitigation	No explicit mitigation	Learnable scalar smoothing	Dense self-attention bypassing local neighborhoods	MRB permutation and global GE	Spectral bisection of subgraphs + Local GSE

Table 3: Theoretical computational complexity and operations breakdown per frame (evaluated on 5G NR H96A, $N = 96, K = 48$).

Algorithm	Asymptotic Bound	Typical Operations Breakdown and Hardware Characteristics
BP-50	$\mathcal{O}(I \cdot \lvert E\rvert)$	50 iterations $\times$ 384 edges $\approx 3.84 \times 10^4$ nonlinear hyperbolic LUTs and additions
W-BP	$\mathcal{O}(I \cdot \lvert E\rvert)$	50 iterations $\times$ 384 edges $\approx 3.84 \times 10^4$ floating-point MACs
ECCT	$\mathcal{O}(L \cdot N^2)$	6 attention layers $\times (2 \times 96^2 \times 128 + 96 \times 128^2) \approx 2.36 \times 10^7$ FLOPs
OSD-3	$\mathcal{O}(K^3 + \lvert\mathcal{T}_{\text{TEP}}\rvert)$	$48^3/3 \approx 3.69 \times 10^4$ $\mathbb{F}_2$ Gaussian operations + 17,296 syndrome evaluations
Lotus (Proposed)	$\mathcal{O}(L_{\text{eff}} \cdot \lvert E\rvert) + \mathcal{O}(1)$	Front-end: 2.4 layers $\times$ 384 edges $\approx 1.84 \times 10^3$ sparse graph ops; Back-end: $\approx \frac{12^3}{3} \approx 576$ XORs

A Comparative Analysis Against Neural Decoders

Based on published empirical and theoretical data from relevant literature—encompassing classical edge-weighted deep unfolding models (Nachmani et al., 2016; 2018), the self-attention-based Error Correction Code Transformer (ECCT) (Choukroun & Wolf, 2022), and recent comprehensive cross-benchmark studies (Yuan et al., 2025)— this appendix systematically reviews the architectural attributes, computational complexity, and FER characteristics across mainstream neural decoders, the classical high-order algebraic benchmark (OSD-3) (Fossorier & Lin, 1995), and the proposed Lotus framework.

A.1 Comprehensive Horizontal Comparison of Algorithmic Properties and Performance

Note: Data for BP-50, OSD-3, and Lotus reflect empirical measurements on our unified simulation platform (H96A, 10^5 frames); parameter counts, complexity orders, and FER ranges for W-BP and ECCT are compiled from published evaluations (Nachmani et al., 2016; 2018; Choukroun & Wolf, 2022; Yuan et al., 2025) on equivalent 5G NR short block lengths for cross-paradigm comparison.

A.2 Theoretical Properties and Performance Characterization of Neural Decoders

Analyzing the algorithmic mechanisms summarized in Table 2 reveals distinct theoretical priorities and performance trade-offs across paradigms:

1. **Theoretical Characteristics of Weighted BP and Deep Unfolded Models (Nachmani et al., 2016; 2018):** Weighted BP maps iterative belief propagation into feedforward network layers. By assigning learnable scale factors and biases to Tanner graph edges, deep unfolding optimizes message damping in the waterfall regime to yield a 0.2– 0.4 dB gain with modest computation and direct hardware mapping. However, from a graph dynamical perspective, learnable edge weights are typically constrained to non-negative scalars to maintain physical sign consistency with log-likelihood ratios. When iterations encounter trapping sets formed by overlapping short cycles, scalar weight scaling cannot break positive feedback loops or pseudocodeword equilibria, leaving a prominent error floor in high-SNR regimes. Furthermore, because parameters are rigidly bound to specific edge topologies, weights cannot transfer across different block lengths or lifting factors, restricting zero-shot OOD deployment.
2. **Representational Characteristics and Trade-offs of Self-Attention Architectures (ECCT (Choukroun & Wolf, 2022) / CrossMPT (Yuan et al., 2025)):** ECCT circumvents localized message passing on Tanner graphs by employing full self-attention paired with syndrome masking to model global cross-bit dependencies, achieving noticeable error-correction gains

over standard BP on short blocks. Nevertheless, recent cross-benchmark investigations (e.g., Yuan et al. (2025)) highlight two structural trade-offs:

- *Alignment between Continuous Optimization and Discrete Parity Checks:* Attention models are trained under continuous cross-entropy over real numbers, outputting posterior bit probabilities. Yet codeword validity is governed by discrete parity checks over the finite field ($\mathbf{H}\mathbf{x}^T \equiv \mathbf{0} \pmod 2$). Lacking discrete algebraic projections, continuous optimization over complex long-tail error patterns easily converges to local continuous minima, yielding high-confidence predictions that violate discrete checks and resulting in a residual error floor in the high-SNR regime.
- *Scalability of Model Size and Computational Complexity:* Standard self-attention incurs quadratic computational and memory complexity with respect to block length ($\mathcal{O}(N^2)$), with parameter footprints reaching millions (1.5M– 9.5M). Moreover, relying on sequential positional encodings binds weights to specific block lengths, necessitating complete retraining when scaling to moderate-to-long codes or varying parity-check structures.

3. **Design Positioning of the Lotus Neuro-Algebraic Solver:** To resolve the fundamental challenges of data-driven models regarding discrete algebraic constraints and cross-scale generalization, Lotus integrates continuous topological representation learning with localized finite-field algebraic elimination:
   - *Front-End Topological Dynamics (Stages 1 & 2 GNN):* Adopts an object-centric decoupled architecture that eliminates positional encodings, operating with merely 272k parameters. Iterative updates based on local factor graph topology naturally exhibit graph permutation equivariance and size-invariance, natively enabling zero-shot OOD extrapolation across arbitrary block lengths and check configurations.
   - *Back-End Local Algebraic Elimination (Stage 3 xLoop):* For persistent long-tail hard instances, xLoop isolates erroneous subgraphs via weighted Laplacian spectral bisection and bounds finite-field elimination to active local subgraphs via heuristic size truncation (with complexity strictly upper-bounded by $\mathcal{O}(d_v K_{\text{top}}^3) \leq \mathcal{O}(1)$), reliably reconstructing valid codewords within an affine solution space over $\mathbb{F}_2$.
   - *Overall Performance Profile:* Empirical results demonstrate that while maintaining efficient hardware execution (average per-frame latency of 0.14– 0.49 ms), Lotus achieves a 75.0%– 83.9% hard-instance recovery rate, approaches the error-correction performance of OSD-3 at high SNRs, and demonstrates the practical efficacy of this hybrid neuro-algebraic formulation.

B Formal Criteria for Parity Equation and Decoding States

B.1 Parity Equation State Definitions

During iterative decoding over bipartite graphs, inspecting solely whether the parity syndrome is 0 or 1 fails to capture the underlying microscopic fidelity of parity constraints. We formalize a three-state taxonomy for parity-check equations:

- **Unsatisfied Checks (USCs):** For a parity-check equation $c_m \in C$, the hard decision of incident variable nodes is determined by the signs of the posterior LLR vector $\mathbf{L}_{\text{post}}$ output by the Variable Module: $\hat{\mathbf{x}} = \frac{1-\text{sign}(\mathbf{L}_{\text{post}})}{2}$. If the parity sum fails over $\mathbb{F}_2$:

$$s_m = \bigoplus_{j \in \mathcal{N}(c_m)} \hat{x}_j = 1, \tag{B.13}$$

  it is designated an Unsatisfied Check (USC). USCs constitute explicit parity conflicts, with their index set denoted by $\mathcal{C}_{\text{USC}} = \{c_m \mid s_m = 1\}$.
- **True Satisfied Checks (TSCs):** If the syndrome satisfies $s_m = 0$ and all incident variable nodes $\mathcal{N}(c_m)$ are correct ($\forall j \in \mathcal{N}(c_m), \hat{x}_j = x_{\text{true},j}$), it is termed a True Satisfied Check (TSC).

- **Pseudo-Satisfied Checks (PSCs):** If the parity equation is nominally satisfied ($s_m = 0$) but its incident variable set contains an even number of erroneous variable nodes, it is designated a Pseudo-Satisfied Check (PSC).

PSCs generate no syndrome violation, yet their incident bit errors collectively mask the underlying error pattern. Any heuristic single-bit flipping algorithm attempting to flip an individual erroneous variable will violate the PSC, creating additional USCs and penalizing the flip; consequently, erroneous variable nodes must flip co-dependently to escape trapping set deadlocks.

B.2 Five-Class Categorization of Decoder Output States

Based on the global syndrome $\mathbf{s} = \mathbf{H}\hat{\mathbf{x}}^T \pmod 2$, the ground-truth transmitted codeword $\mathbf{x}_{\text{true}}$, and the channel LLR metric $\Lambda(\mathbf{x}) = \sum_{i=1}^{N}(1-2x_i)y_i$, the terminal outputs of the iterative graph decoder are partitioned into five mutually exclusive categories:

- **True Convergence (TC):** $\mathbf{s} = \mathbf{0}$ and the decided codeword matches the transmitted codeword ($\hat{\mathbf{x}} = \mathbf{x}_{\text{true}} = \mathbf{x}_{\text{send}}$), representing genuine successful convergence.
- **Maximum Likelihood Convergence (MLC) / ML Error:** $\mathbf{s} = \mathbf{0}$ with $\hat{\mathbf{x}} \neq \mathbf{x}_{\text{send}}$, yet the channel LLR metric of the decided codeword exceeds that of the transmitted codeword:

  $$\Lambda(\hat{\mathbf{x}}) > \Lambda(\mathbf{x}_{\text{send}}). \tag{B.14}$$

  This phenomenon arises from severe Gaussian noise perturbations, where the Euclidean distance from the channel observation to another valid codeword is smaller than that to the transmitted codeword. Information-theoretically, this represents the theoretical performance limit of an optimal Maximum Likelihood (ML) decoder; finding this candidate demonstrates that the algorithm possesses global solving capability. In the supervised training curriculum of Lotus GNN, penalizing such frames against the transmitted label would inject false gradient targets, misleading the loss function into fitting suboptimal trajectories. Therefore, ML Convergence is categorized as successful convergence during training.
- **Suboptimal Convergence (SC):** $\mathbf{s} = \mathbf{0}$ with $\hat{\mathbf{x}} \neq \mathbf{x}_{\text{send}}$, but its LLR metric is inferior to that of the transmitted codeword:

  $$\Lambda(\hat{\mathbf{x}}) < \Lambda(\mathbf{x}_{\text{send}}). \tag{B.15}$$

  This indicates that while the decoder successfully reached the valid code space, it converged to a neighboring suboptimal codeword. This state verifies constraint satisfaction capability but indicates failure to identify the ML codeword. During Lotus GNN training, SC is treated as non-converged to encourage further learning; during inference without additional supervisory signals (e.g., cyclic redundancy check (CRC)), it is nominally accepted as a valid codeword.
- **Systematic-Bits-Correct (Sys-OK):** In systematic codes where information bits are extracted directly, the extracted systematic bits happen to be correct despite a non-zero syndrome ($\mathbf{s} \neq \mathbf{0}$) on parity bits. In constraint satisfaction problems, a non-zero syndrome signifies **Non-Convergence (NC)**; fortuitous correctness of systematic bits is merely an observational truncation artifact and does not constitute algorithmic convergence. During Lotus GNN training, it is treated as non-convergent.
- **Non-Convergence (NC):** $\mathbf{s} \neq \mathbf{0}$, where the decoder exhausts its iteration budget without satisfying all parity checks. Trapped in cyclic structures, the associated variable nodes suffer dynamical stagnation. Empirical data indicate that NC accounts for over 98% of decoding failures in classical BP.

Demarcation of Pseudocodewords: Pseudocodewords arise from finite graph covers of the bipartite graph or extreme points of the linear programming (LP) relaxation polytope (Candès & Tao, 2005; Koetter & Vontobel, 2003). Because pseudocodewords yield non-zero syndromes on the canonical parity-check matrix ($\mathbf{s} \neq \mathbf{0}$); therefore, they fall under the Non-Convergence (NC) category in our taxonomy.

C Dynamics Expansions and Parameterizations

C.1 Variable Module Input Embedding and Permutation Equivariance Proof

Theoretical Insight (Dynamical Symmetry Breaking without Positional Encodings):
Unlike conventional neural decoders that rely on static positional encodings to break 1-WL graph symmetries—inevitably overfitting to specific graph scales and eigenspectra—Lotus operates strictly PE-free. By governing message dynamics through degree-normalized, topology-intrinsic conflict metrics, Lotus detects subtle continuous-relaxation asymmetries induced by channel observations and injects targeted parity momentum, achieving dynamical symmetry breaking while preserving strict permutation equivariance across arbitrary graph topologies.

In Section 3.2, the initial input to each variable node comprises a 3-dimensional physical feature vector extracted from channel observations:

$$\mathbf{feature}_i = \left[L_{\text{ch},i},\ |L_{\text{ch},i}|,\ \sigma^2_{\text{ch}}\right]^T \in \mathbb{R}^{\text{dim}_{\text{in}}}. \tag{C.16}$$

1. **Input Feature Projection and Coordinate-Wise Expansion:** Through a globally shared linear projection matrix $\mathbf{W}_{\text{in}} \in \mathbb{R}^{D_{\text{latent}} \times \text{dim}_{\text{in}}}$ and bias vector $\mathbf{b}_{\text{in}} \in \mathbb{R}^{D_{\text{latent}}}$, channel features are mapped into latent state vectors:

$$V_{\text{channel},i}[k] = \sum_{p=1}^{\text{dim}_{\text{in}}} \mathbf{W}_{\text{in}}[k,p] \cdot \mathbf{feature}_i[p] + \mathbf{b}_{\text{in}}[k], \quad k = 1, \ldots, D_{\text{latent}}. \tag{C.17}$$

2. **Layer-0 Check 1-Hop Round-Trip Aggregation Expansion:** Prior to multi-layer unfolded iterations, the layer-0 latent state incorporates topological structure and syndrome priors via a 1-hop round trip ($V \to C \to V$) over the factor graph:

$$C^{(0)}_{\text{check},m} = \frac{1}{\sqrt{d_{c_m}}} \sum_{j \in \mathcal{N}(m)} V_{\text{channel},j} + \mathbf{W}_{\text{syn}} s_m + \mathbf{b}_{\text{syn}} \in \mathbb{R}^{D_{\text{latent}}}, \tag{C.18}$$

$$V_{\text{check},i} = \frac{1}{\sqrt{d_{v_i}}} \sum_{m \in \mathcal{N}(i)} C^{(0)}_{\text{check},m} \in \mathbb{R}^{D_{\text{latent}}}, \tag{C.19}$$

$$h_i^{(0)} = V_{\text{channel},i} + \mathbf{W}_{\text{check}} V_{\text{check},i} + \mathbf{b}_{\text{check}} \in \mathbb{R}^{D_{\text{latent}}}. \tag{C.20}$$

   All learnable transformation weights are centralized within the Variable Module, while check nodes execute stateless, topologically normalized message aggregation at layer 0.

3. **Algebraic Proof of Permutation Equivariance:** Let $\boldsymbol{\pi} \in \mathcal{S}_N$ denote an arbitrary permutation matrix acting on variable nodes. Because variable nodes remain isotropic and omit positional encodings, the channel feature matrix transforms as $\mathbf{F}_{\boldsymbol{\pi}} = \boldsymbol{\pi}\mathbf{F}$. For any shared point-wise mapping $\text{Linear}_{\text{in}}$, since weights operate along the feature channel dimension rather than the node permutation dimension, it directly holds that:

$$\begin{aligned} \text{Linear}_{\text{in}}(\boldsymbol{\pi}\mathbf{F}) &= (\boldsymbol{\pi}\mathbf{F})\mathbf{W}_{\text{in}}^T + \mathbf{1}_N \mathbf{b}_{\text{in}}^T \\ &= \boldsymbol{\pi}(\mathbf{F}\mathbf{W}_{\text{in}}^T + \mathbf{1}_N \mathbf{b}_{\text{in}}^T) \\ &= \boldsymbol{\pi}\,\text{Linear}_{\text{in}}(\mathbf{F}). \end{aligned} \tag{C.21}$$

   The factor graph parity-check matrix transforms under permutation as $\mathbf{H}_{\boldsymbol{\pi}} = \mathbf{H}\boldsymbol{\pi}^T$. The check aggregation and backward projection satisfy:

$$\begin{aligned} \mathbf{V}_{\text{check},\boldsymbol{\pi}} &\propto \mathbf{H}_{\boldsymbol{\pi}}^T \mathbf{H}_{\boldsymbol{\pi}} \mathbf{V}_{\text{channel},\boldsymbol{\pi}} \\ &= (\boldsymbol{\pi}\mathbf{H}^T)(\mathbf{H}\boldsymbol{\pi}^T)(\boldsymbol{\pi}\mathbf{V}_{\text{channel}}) \\ &= \boldsymbol{\pi}(\mathbf{H}^T\mathbf{H}\mathbf{V}_{\text{channel}}) \\ &= \boldsymbol{\pi}\mathbf{V}_{\text{check}}. \end{aligned} \tag{C.22}$$

   Consequently, the initial state $h_i^{(0)}$ and all subsequent latent states across unfolded layers remain equivariant under arbitrary graph permutations, establishing the algebraic foundation for zero-shot OOD extrapolation across scales.

C.2 Bidirectional Extrinsic Message Passing and Punctured Product Expansions

In Section 3.3, the Check Module establishes constraint-driven information flows via bidirectional extrinsic message passing:

1. **Variable-to-Check (V2C) Extrinsic Messages:**

$$m_{i\to c}^{(\ell)} = L_{\text{ch},i} + \sum_{c'\in\mathcal{N}(i)\setminus\{c\}} m_{c'\to i}^{(\ell-1)}. \tag{C.23}$$

   At the initial iteration ($\ell = 1$), extrinsic check inputs are initialized to zero ($m_{c\to i}^{(0)} \equiv 0$), yielding $m_{i\to c}^{(1)} = L_{\text{ch},i}$.

2. **Check-to-Variable (C2V) Hyperbolic Tangent Products and Sign-Magnitude Decomposition:** Check node c aggregates incoming messages and evaluates parity checks in the log-likelihood ratio (LLR) domain based on the bipolar target syndrome $s_c \in \{-1, +1\}$ defined in equation 3.3:

$$m_{c\to i}^{(\ell)} = 2\tanh^{-1}\left(s_c \prod_{j\in\mathcal{N}(c)\setminus\{i\}} \tanh\left(\frac{m_{j\to c}^{(\ell)}}{2}\right)\right). \tag{C.24}$$

   In numerical implementations, equation C.24 is computed via decoupled sign and magnitude operations to preclude hyperbolic tangent overflow/underflow:

$$\operatorname{sign}(m_{c\to i}^{(\ell)}) = s_c \cdot \prod_{j\in\mathcal{N}(c)\setminus\{i\}} \operatorname{sign}(m_{j\to c}^{(\ell)}), \tag{C.25}$$

$$\ln\tanh\left(\frac{|m_{c\to i}^{(\ell)}|}{2}\right) = \sum_{j\in\mathcal{N}(c)\setminus\{i\}} \ln\tanh\left(\frac{|m_{j\to c}^{(\ell)}|}{2}\right). \tag{C.26}$$

   The target syndrome s_c establishes the baseline parity polarity, such that a polarity flip in any single incoming variable produces an exact sign reversal in the outgoing check message.

C.3 Topology Module Dual Adaptive Dynamical Gating Parameterization and Piecewise Calibration

Theoretical Insight (Spatiotemporal TCR as Dynamic Lagrangian Relaxation):
From the perspective of continuous constrained optimization, bipartite parity edges impose rigid, non-smooth algebraic penalties over $\mathbb{F}_2$. Spatiotemporal TCR acts as an adaptive Lagrangian relaxation: spatial soft masks exponentially dampen severe topological conflicts, while temporal node gates systematically attenuate historical hidden-state inertia and magnify corrective gradients. This transforms an otherwise intractable, non-convex discrete energy barrier into a smooth, traversable continuous trajectory.

In Section 3.4, the Topology Module synthesizes the composite node conflict metric $\text{conf}_{\text{node},i}^{(\ell)}$ by coupling edge conflicts with gradient conflicts. The explicit parameterizations and piecewise-linear clamping rules for the dual dynamical gates are given as follows:

1. **Node Conflict Metric Synthesis:**

$$\overline{\text{conf}}_{\text{edge},i}^{(\ell)} = \frac{1}{d_i}\sum_{c\in\mathcal{N}(i)} \max\left(0, -\tilde{x}_i^{(\ell-1)} \cdot \tanh(\text{MLP}_{\text{edge}}(m_{c\to i}^{(\ell)}))\right), \tag{C.27}$$

$$\text{conf}_{\text{grad},i}^{(\ell)} = \tanh\left(\gamma_\ell \cdot \max\left(0, -L_{\text{post},i}^{(\ell-1)} \cdot \mathbf{g}_{\text{fwd},i}^{(\ell)}\right)\right), \tag{C.28}$$

$$\text{conf}_{\text{node},i}^{(\ell)} = \overline{\text{conf}}_{\text{edge},i}^{(\ell)} \cdot \left(1.0 + \text{conf}_{\text{grad},i}^{(\ell)}\right). \tag{C.29}$$

2. **Piecewise Clamping of the Adaptive Memory Attenuation Gate:**

$$\alpha_{\text{atten},i}^{(\ell)} = \text{clamp}\left(\exp(-\lambda_\ell \cdot \text{conf}_{\text{node},i}^{(\ell)}),\ \alpha_{\min},\ 1.0\right), \tag{C.30}$$

where $\lambda_\ell \geq 0$ is a layer-dependent learnable damping gain, and $\alpha_{\min} = 0.05$ denotes the memory retention floor, preventing latent states of high-conflict nodes from zeroing out entirely and disrupting gradient backpropagation.

3. **Piecewise Clamping of the Adaptive Increment Amplification Gate:**

$$\alpha_{\text{amp},i}^{(\ell)} = \text{clamp}\left(1.0 + \kappa_\ell \cdot \text{conf}_{\text{node},i}^{(\ell)},\ 1.0,\ \alpha_{\max}\right), \tag{C.31}$$

where $\kappa_\ell \geq 0$ is a layer-dependent learnable amplification gain, and $\alpha_{\max} = 3.5$ denotes the saturation ceiling to suppress numerical instabilities from severe multi-check conflicts. For conflict-free nodes, $\alpha_{\text{atten}} \to 1.0$ and $\alpha_{\text{amp}} \to 1.0$, smoothly reverting to standard residual connections.

D Spatial Geometric Alignment and Anti-Over-Smoothing Mechanism of Forward Analytical Gradients

To elucidate how forward analytical gradients guide latent state updates across deep layers, we quantitatively tracked the geometric cosine similarity and representational smoothness between the forward analytical gradient $\mathbf{g}_{\text{fwd}}$, the backward parity loss gradient $\mathbf{g}_{\text{bwd}}$, and the backward global total gradient $\mathbf{g}_{\text{global}}$ on a unified benchmark graph (5G NR H48A):

- **Forward Analytical Gradient $\mathbf{g}_{\text{fwd}}$:** Evaluated via equation 5 with respect to the input soft decision vector $\tilde{\mathbf{x}}^{(\ell-1)}$ at layer ℓ, defining the steepest descent direction of the soft parity potential function ($\mathbf{g}_{\text{fwd}} = -\nabla_{\tilde{\mathbf{x}}}\mathcal{E}$) under the bipolar target syndrome $s_m \in \{-1, +1\}$ equation 3.3;
- **Backward Parity-Check Loss Gradient $\mathbf{g}_{\text{bwd}}$:** Computed via automatic differentiation (Autograd) with respect to the output soft decision vector $\tilde{\mathbf{x}}^{(\ell)}$ at layer ℓ, representing the first-order derivative of the parity loss equation 3.5: $\mathbf{g}_{\text{bwd}}^{(\ell)} = \nabla_{\tilde{\mathbf{x}}^{(\ell)}}\mathcal{L}_{\text{check}}^{(\ell)}$;
- **Backward Global Total Gradient $\mathbf{g}_{\text{global}}$:** Derived from the global multi-task loss equation 8, fusing the supervised BCE loss $\mathcal{L}_{\text{BCE}}^{(\ell)}$ equation 3.5 with the parity-check loss $\mathcal{L}_{\text{check}}^{(\ell)}$.

By definition, the forward analytical gradient follows the steepest descent of the potential ($-\nabla\mathcal{E}$), whereas backward Autograd computes the ascent gradient of the loss ($+\nabla\mathcal{L}$); when both steer optimization in the same direction, their cosine similarity is naturally negative ($\rho < 0$). While both share the same bipolar target syndrome and exclude ground-truth transmitted labels, $\mathbf{g}_{\text{fwd}}$ differentiates with respect to layer inputs whereas $\mathbf{g}_{\text{bwd}}$ differentiates with respect to layer outputs, yielding distinct alignment dynamics across iterations:

1. **From Shallow Statistical Near-Orthogonality to Deep Error-Subspace Directional Alignment:**
   - *Shallow High-Dimensional Statistical Near-Orthogonality ($\rho^{(1)} = -0.026$):* In the initial iteration, heavily perturbed by channel Gaussian noise, numerous checks signal violations (average USC ≈ 1.94). Because the punctured extrinsic product $Q_{m,i}$ in equation 3.3 involves degree $d_c - 1 \approx 5$, its sign exhibits extreme sensitivity to weak noise fluctuations. In initial message passing, if a low-confidence bit flips sign between input $\tilde{\mathbf{x}}^{(0)}$ and output $\tilde{\mathbf{x}}^{(1)}$, the product reverses polarity ($Q_{m,i} \to -Q_{m,i}$). Across the $N = 48$ vector space, gradient components from distinct unsatisfied checks cancel out, yielding statistical near-orthogonality between forward analytical and backward parity gradients ($\rho^{(1)} = -0.026$).
   - *Deep Early-Stopping Filtering and Error-Subspace Directional Alignment ($\rho^{(8)} = -0.369 \to -0.540$):* As iterations proceed, 92.2% **of correct frames converge and trigger early stopping** (reducing the batch-wide average USC count to 0.12). Because early-stopped frames have their backward losses masked (zero gradient norm)

and are filtered from gradient similarity statistics, metrics at layer 8 **concentrate primarily on the remaining** 7.8% **trapping set instances**. Within these extreme hard instances, satisfied checks exhibit zero residual ($F_m \to 0$), automatically silencing gradients on non-erroneous nodes and focusing gradient energy onto the error subspace spanned by trapped short-cycle variables. In this subspace, background variable signs are stabilized, eliminating product sign cancelations. Modulated by TCR edge masking and topological node gating, the network actively drives corrective updates ($\tilde{\mathbf{x}}^{(8)} \neq \tilde{\mathbf{x}}^{(7)}$). Despite the angular offset induced by non-convex curvature between input derivatives ($-\nabla_{\tilde{\mathbf{x}}^{(7)}}\mathcal{E}$) and output derivatives ($\nabla_{\tilde{\mathbf{x}}^{(8)}}\mathcal{L}_{\text{check}}$), the measured angle contracts from near-orthogonal (88.5°) in layer 1 to an acute angle of 68° in layer 8 (with cosine similarity climbing monotonically to $\mathbf{-0.369}$). Incorporating supervised BCE gradients further narrows the angle to 57° (cosine similarity reaching $\mathbf{-0.540}$, $z = 3.75\sigma, p < 10^{-4}$). This acute directional co-alignment confirms that forward analytical gradients consistently provide corrective algebraic guidance toward ground-truth codewords. However, because cyclic deadlocks form discrete topological barriers that cannot be traversed by single-step continuous relaxations, the dynamical trajectory remains anchored along this co-aligned gradient. When provided with dense hard-instance resamplings (as demonstrated in the Stage-2 expert network ablation, where autonomous convergence reaches 58.17% and cosine similarity stabilizes within -0.394 to -0.573), continuous networks can traverse specific barriers. Nonetheless, because trapping sets exhibit vast combinatorial diversity that finite datasets cannot exhaustively cover, incorporating the distribution-agnostic xLoop module provides deterministic algebraic solving to eliminate persistent trapping sets.

2. **Mitigation of Deep Representational Over-Smoothing (Anti-Over-Smoothing Mechanism):** Conventional deep unfolded graph models suffer from representational homogenization due to excessive message mixing across deep iterations. Layer-wise tracking of the average spatial cosine similarity among variable node latent states reveals that, under the continuous guidance of forward analytical gradients, latent cosine similarity increases only marginally from 0.05 at layer 1 to **0.19** at layer 8. This effectively suppresses over-smoothing collapse across deep unfolded layers, preserving highly discriminative dynamical residual features for downstream algebraic elimination in xLoop.

E Multi-Task Loss Balancing, Layer-Wise Supervision, and Numerical Stability

E.1 Inter-Task Balancing: Kendall Bayesian Uncertainty and Progressive DLR

Lotus employs two independently trained 8-layer unfolded networks (Stage-1 Base GNN and Stage-2 Expert GNN), each adopting a complementary task-balancing strategy tailored to its underlying data distribution:

1. **Stage-1 Base GNN (Global Denoising via Kendall Uncertainty Balancing):** Operating across all SNR regimes, the network incorporates Kendall's Bayesian homoscedastic uncertainty mechanism (Kendall et al., 2018). By formulating the relative weights of the BCE loss and parity-check loss as learnable homoscedastic variance parameters, the network end-to-end adaptively determines the optimal task ratio, smoothly balancing channel denoising with parity constraint enforcement.
2. **Stage-2 Expert GNN (Hard-Instance Resolution via Progressive DLR Schedule):** For non-converged and suboptimal trapping set instances, the supervised BCE gradient and the unsupervised parity-check gradient frequently exhibit severe directional conflict. Empirical evaluations indicate that retaining Kendall adaptive balancing causes optimization to stagnate at local saddle points (attaining an autonomous hard-instance convergence rate of merely 7.25%). Consequently, the expert network adopts a monotonically progressive Dynamic Loss Ratio (DLR) schedule, scaling the parity-check weight λ_ℓ from shallow to deep layers (e.g., ramping from 0.15 in shallow layers to 0.40 in deep layers) to effectively steer latent states toward the valid codeword null space.

3. **Two-Stage Inference Cascade and Cross-Stage Routing:** During forward inference, the Stage-1 Base GNN executes global denoising and early-stopping evaluation. Non-converged frames with non-zero syndromes are routed to the Stage-2 Expert GNN. If both neural stages fail to achieve complete convergence, the decoder delivers the best historical state exhibiting the minimum unsatisfied check (USC) count across both stages to xLoop for algebraic elimination.

E.2 Layer-Wise Supervision Mechanism and Weight Normalization

Beyond intra-layer task balancing, distributing supervision weights w_ℓ across unfolded layers governs gradient propagation dynamics. Lotus applies distinct layer-weight schedules for the two networks, enforcing a normalized total gradient budget ($\sum_{\ell=1}^{L} w_\ell = 1.0$):

- **Layer-Wise Weight Generation and Normalization:** For an L- layer deep unfolded network, unnormalized layer weights $\{\bar{w}_\ell\}_{\ell=1}^{L}$ are generated and normalized as follows:
  1. **Stage-1 Base GNN (Guaranteed Baseline Budget + Learnable Parameters):** Unnormalized weights $\bar{w}_\ell$ are optimized via backpropagation and mapped to probabilities via Softmax: $p_\ell = \exp(\bar{w}_\ell)/\sum_{k=1}^{L}\exp(\bar{w}_k)$. To prevent shallow layers from suffering optimization laziness during backpropagation, we inject a guaranteed baseline budget f_ℓ parameterized by a base ratio $f_{\text{ratio}} \in [0, 1]$ ($f_{\text{ratio}} = 0.60$ in experiments) and a linear incline factor κ_{incline} ($\kappa_{\text{incline}} = 0.50$ in experiments):

$$f_\ell = \frac{f_{\text{ratio}}}{L} \cdot (1.0 + \kappa_{\text{incline}} \cdot u_\ell), \quad u_\ell \in [-1, +1], \quad \sum_{\ell=1}^{L} u_\ell = 0. \tag{E.32}$$

     The effective layer-wise supervision weight is evaluated as a convex combination of the baseline budget and the learnable distribution:

$$w_\ell = f_\ell + (1.0 - f_{\text{ratio}}) \cdot p_\ell. \tag{E.33}$$

  2. **Stage-2 Expert GNN (Linear Incline Schedule):** To accommodate complex trapping set topologies, layer weights follow a fixed, monotonically increasing linear progression, normalized such that $\sum_{\ell=1}^{L} w_\ell = 1.0$. This allows shallow layers freedom for topological state exploration while deep layers receive strict parity enforcement.

  This formulation enforces $\sum_{\ell=1}^{L} w_\ell = 1.0$, fixing the aggregate gradient budget and eliminating gradient starvation across layers (corroborating the ablation in Appendix I).

E.3 Training Numerical Stability and Parameter Decoupling

To ensure numerical stability when training on long-tail hard instances and backpropagating through deep unfolded layers, Lotus establishes a dual safeguard across batch aggregation and optimizer parameter grouping:

1. **Soft-Clamped Normalization on Active Samples:** Early-converged samples within a batch are masked out via early stopping. To prevent scarce active samples in deep layers from inducing gradient explosion due to a vanishing denominator, batch loss calculation incorporates soft-clamped normalization:

$$\mathcal{L}_{\text{batch}} = \frac{\sum_{b=1}^{B} \mathcal{L}_b \cdot \mathbb{I}_{\text{active}}(b)}{\max\left(1.0,\ B \cdot r_{\min}\right)}, \tag{E.34}$$

   where $r_{\min} > 0$ defines the minimum active sample floor ($r_{\min} = 0.10$ in experiments).
2. **Decoupled Parameter Groups for Topology Optimization:** The learnable weights $\mathbf{W}_{\text{share}}$ within the Variable Module are updated using standard AdamW with nominal weight decay. In contrast, the scale-invariant control parameters of the Topology Module (such as TCR soft masks and node gating weights $\lambda_\ell, \kappa_\ell, \beta_\ell, b_\ell$) are segregated into a specialized topology parameter group assigned an independent fine-tuning learning rate and reduced weight decay, ensuring smooth dynamical evolution during training.

F xLoop Algorithmic Protocols, Spectral Bisection, and Cascaded Deployments

F.1 Matrix Power Expansion for Multi-Source 3-Hop Neighborhoods and Fast-Bypass Criteria

In Section 4.1, xLoop initiates localized topological expansion anchored at the unsatisfied check set $\mathcal{C}_{\text{USC}}$:

1. **Bipartite Adjacency Matrix Higher-Order Power Expansion:** Let the bipartite adjacency matrix of the factor graph be defined as:
$$\mathbf{A}_{\text{bip}} = \begin{bmatrix} \mathbf{0} & \mathbf{H} \\ \mathbf{H}^T & \mathbf{0} \end{bmatrix} \in \{0,1\}^{(N+M)\times(N+M)}. \tag{F.35}$$
The 3-hop transition operator originating from check nodes corresponds to the block components of its third matrix power:
$$\mathbf{A}_{\text{bip}}^3 = \begin{bmatrix} \mathbf{0} & \mathbf{H}\mathbf{H}^T\mathbf{H} \\ \mathbf{H}^T\mathbf{H}\mathbf{H}^T & \mathbf{0} \end{bmatrix}. \tag{F.36}$$
Its lower off-diagonal block $\mathbf{K}_{c2v} = (\mathbf{H}^T\mathbf{H}\mathbf{H}^T) > 0$ enumerates all variable nodes reachable from any check equation via a 3-hop path (check $\rightarrow$ variable $\rightarrow$ check $\rightarrow$ variable).
2. **Fast-Bypass Exit Matrix Boundary Formulation:** Based on 3-hop variable overlaps, we construct the check overlap indicator matrix $\mathbf{O}_C(a,b) = \mathbb{I}\left(|\mathcal{V}_{\text{3hop}}(c_a) \cap \mathcal{V}_{\text{3hop}}(c_b)| \geq 1\right)$. The set of disjoint clusters $\{\mathcal{C}_j\}_{j=1}^J$ across the graph corresponds to the connected components of $\mathbf{O}_C$. If either of the following conditions is satisfied, the system determines that the error pattern has diffused beyond constant-order tractability and triggers a fast-bypass exit:
$$\max_{1\leq j\leq J} |\mathcal{C}_j| > 16 \quad \text{or} \quad J > J_{\max} = 8. \tag{F.37}$$

F.2 Weighted Check Laplacian Spectral Bisection and Fiedler Vector Eigendecomposition

In Section 4.2, when handling moderately sized clusters ($4 < |\mathcal{C}_j| \leq 16$), the system executes weighted Laplacian spectral bisection via divide-and-conquer:

1. **Weighted Adjacency and Laplacian Matrix Construction:** On the connected cluster $\mathcal{C}_j$, we construct the check-edge weight matrix $A_C(a,b) = |\mathcal{N}(c_a) \cap \mathcal{N}(c_b)|$ and diagonal degree matrix $D_C(a,a) = \sum_b A_C(a,b)$. The unnormalized graph Laplacian matrix is:
$$\mathbf{L}_C = \mathbf{D}_C - \mathbf{A}_C \in \mathbb{R}^{|\mathcal{C}_j|\times|\mathcal{C}_j|}. \tag{F.38}$$
2. **Rayleigh Quotient Minimization and Fiedler Vector Evaluation:** According to spectral graph theory, the smallest eigenvalue of $\mathbf{L}_C$ is $\lambda_1 = 0$, associated with the all-ones eigenvector $\mathbf{1}$. The second smallest eigenvalue (algebraic connectivity) λ_2 and its corresponding eigenvector $\mathbf{v}_2$ (the Fiedler vector) are defined via Rayleigh quotient minimization under orthogonality constraints:
$$\mathbf{v}_2 = \arg\min_{\mathbf{v}\perp\mathbf{1},\, \|\mathbf{v}\|_2=1} \mathbf{v}^T\mathbf{L}_C\mathbf{v} = \arg\min_{\mathbf{v}\perp\mathbf{1},\, \|\mathbf{v}\|_2=1} \sum_{a<b} A_C(a,b)(\mathbf{v}[a] - \mathbf{v}[b])^2. \tag{F.39}$$
Given the compact dimension $|\mathcal{C}_j| \leq 16$, the algorithm employs shifted power iteration with mean-deflation, setting $\mathbf{B} = \alpha\mathbf{I} - \mathbf{D}_C^{-1/2}\mathbf{L}_C\mathbf{D}_C^{-1/2}$, which converges to $\mathbf{v}_2$ with high precision in merely 4–6 iterations.
3. **Median Split and Cut-Set Consistency Alignment:** Checks are bisected into sub-clusters $\mathcal{C}_{j,1}$ and $\mathcal{C}_{j,2}$ based on $\text{median}(\mathbf{v}_2)$. The intersecting variables form the cut-set $V_{\text{cut}} = \mathcal{V}_{j,1} \cap \mathcal{V}_{j,2}$. The local candidate error vectors $\mathbf{e}_{j,1}$ and $\mathbf{e}_{j,2}$ must maintain consistent state assignments over V_{cut} to synthesize the complete error pattern via bitwise OR:
$$\mathbf{e}_j = \mathbf{e}_{j,1} \vee \mathbf{e}_{j,2}, \tag{F.40}$$
guaranteeing that overlapping short cycles are cleanly decoupled along the weak cut-set.

F.3 Sliding-Window Latent State Variance Estimation and Variance-Deflated Candidate Selection

In Section 4.2, to attenuate falsely confident beliefs induced by positive feedback loops within trapping sets, the system incorporates a multi-layer dynamical variance deflation model:

1. **Temporal Sliding-Window Unbiased Latent State Variance Estimation:** For variable node i, given its latent states $h_i^{(\ell)} \in \mathbb{R}^{D_{\text{latent}}}$ across layers ($\ell = 1, \ldots, L$), the temporal mean and unbiased sample variance are:

$$\bar{h}_i = \frac{1}{L}\sum_{\ell=1}^{L} h_i^{(\ell)}, \quad \text{Instab}_i = \frac{1}{(L-1)D_{\text{latent}}}\sum_{\ell=1}^{L}\left\|h_i^{(\ell)} - \bar{h}_i\right\|_2^2. \tag{F.41}$$

2. **Cross-Layer Microscopic Sign Conflict Accumulation Metric:** We evaluate the weighted frequency of sign disagreements between the variable node and incident extrinsic check messages along iterative trajectories:

$$\text{conf}_{\text{micro},i} = \sum_{\ell=1}^{L} w_\ell \sum_{c\in\mathcal{N}(i)} \mathbb{I}\left(\text{sign}(m_{i\to c}^{(\ell)}) \neq \text{sign}(m_{c\to i}^{(\ell)})\right). \tag{F.42}$$

3. **Standardized z-Score and Adaptive Confidence Deflation:** Within each sub-cluster, standardized scores are evaluated as $z_{\text{instab},i} = (\text{Instab}_i - \mu_{\text{instab}})/\sigma_{\text{instab}}$ and $z_{\text{micro},i} = (\text{conf}_{\text{micro},i} - \mu_{\text{micro}})/\sigma_{\text{micro}}$. The deflation factor and deflated confidence metric are given by:

$$\text{Deflation}_i = 1.0 + 2.0\max(0, z_{\text{instab},i}) + 2.0\max(0, z_{\text{micro},i}), \tag{F.43}$$

$$L_{\text{xloop},i} = \frac{L_{\text{post},i}}{\text{Deflation}_i}. \tag{F.44}$$

Sorting under ascending magnitudes $|L_{\text{xloop},i}|$ prioritizes fluctuating error variables into the candidate set $\mathcal{V}_{\text{cand}}$ ($|\mathcal{V}_{\text{cand}}| = K_{\text{top}} \leq 32$).

F.4 Localized GF(2) Streaming Gaussian Elimination Procedures

Theoretical Insight (The Structural Necessity of Streaming Gaussian Elimination):
Heuristic decoders typically truncate both rows and columns to formulate small square systems, unavoidably suffering from catastrophic secondary syndrome leakage on global checks. In contrast, xLoop constructs a tall-and-skinny, massively over-determined system ($M \times K_{\text{top}}$) preserving all M canonical parity rows (Lemma G.1). Streaming Gaussian Elimination (SGE) is fundamentally necessary to pipeline these global constraints as a sequential bit-level stream, guaranteeing absolute zero syndrome leakage while maintaining a strictly bounded $\mathcal{O}(1)$ working memory footprint.

In Section 4.3, the elimination unit solves the localized inhomogeneous linear system $\mathbf{H}_{\text{sub}}\mathbf{e} \equiv \mathbf{s} \pmod 2$ over the finite field $\mathbb{F}_2$:

1. **Sparse Matrix Extraction and Reliability Column Pivoting:** Columns corresponding to candidate variables are extracted from the global M- row parity-check matrix to form $\mathbf{H}_{\text{sub}} \in \mathbb{F}_2^{M\times K_{\text{top}}}$. Rows devoid of candidate connections are pruned, bounding active rows by $M_{\text{active}} \leq d_v K_{\text{top}}$. Columns are sorted in ascending order of reliability, prioritizing low-confidence variables as leading pivot candidates.
2. **Bitwise XOR Streaming Elimination Protocol:**
   - **Step 1 (Forward Elimination):** For the k- th candidate column, search uneliminated rows for a pivot entry of 1. If found, swap it to the current pivot row and apply SIMD bitwise XOR operations to eliminate all subsequent rows containing a 1 in this column.
   - **Step 2 (Algebraic Compatibility Check):** Following elimination, if an all-zero coefficient row corresponds to a non-zero augmented right-hand syndrome entry ($0 = 1$), the augmented system is inconsistent ($\mathbf{s} \notin \text{span}(\mathbf{H}_{\text{sub}})$). The system declares algebraic unreachability and immediately initiates early termination fallback.

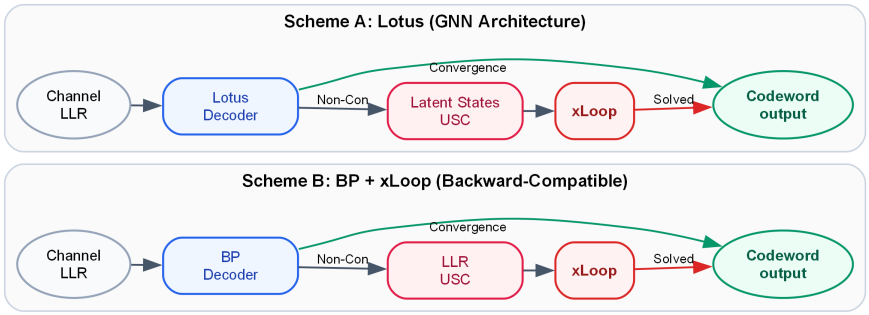


Figure 4: Cascaded decoding architectures: (a) Scheme A: End-to-end neural-algebraic pipeline; (b) Scheme B: Backward-compatible BP + xLoop plug-in.

- **Step 3 (Null Space and Affine Solution Space Construction):** If consistent, back-substitution yields a particular solution $\mathbf{e}_0 \in \mathbb{F}_2^{K_{\text{top}}}$ and a null space basis matrix $\mathbf{B} \in \mathbb{F}_2^{K_{\text{top}} \times d}$ across free variables. If the degrees of freedom satisfy $d > d_{\max} = 12$, we truncate and retain the 12 least reliable basis vectors, establishing a localized affine solution space $\mathcal{A}$ of size $2^d \leq 4096$.

F.5 Reliability Safety Margins, Pathological Solution Pruning, and Completeness Trade-offs

In Section 4.4, the system establishes reliability safety margins Δ_{margin} and Hamming weight constraints. This subsection provides supplementary mechanistic analysis:

1. **Physical Waveform Support and Pathological Pseudocodeword Interception:** Under low SNRs or deep fading, severe channel noise bursts may displace the channel observation vector away from the transmitted codeword's Voronoi region, causing elimination to uncover pathological algebraic solutions that satisfy parity checks ($\mathbf{H}\hat{\mathbf{x}}^T \equiv \mathbf{0}$) but incur excessive Euclidean distance penalties relative to the observed waveform. The relative LLR metric gain is defined as $\mathbf{\Gamma}[\mathbf{e}] = \Lambda(\hat{\mathbf{x}}) - \Lambda(\mathbf{x}_{\text{init}})$. When the relative likelihood loss of the optimal candidate exceeds the safety margin ($\mathbf{\Gamma}[\mathbf{e}_{\text{opt}}] < -\Delta_{\text{margin}}$, with recommended engineering calibration $\Delta_{\text{margin}} = 12.0$– 18.0), the algebraic flip lacks sufficient physical waveform support and represents an unviable noise-induced artifact. The system prunes this candidate and falls back to the best historical decision from preceding stages, safeguarding the error-rate performance baseline under hostile channel conditions.
2. **Structural Trade-off Between Algebraic Completeness and Constant-Bounded Complexity:** The fundamental bottleneck of NP-hard combinatorial constraint satisfaction lies in combinatorial explosion under worst-case long-tail error distributions. Classical ML decoders (such as OSD-3 or full-matrix Gaussian elimination) insist on complete algebraic coverage across all noise realizations, paying the penalty of cubic or higher complexity growth ($\mathcal{O}(N^3)$), which precludes low-latency hardware realization for long block lengths. The foundational design principle of xLoop is to **purposefully concede algebraic completeness under catastrophically diffused noise bursts via heuristic size truncation, in exchange for a deterministic, constant-bounded worst-case computational complexity** $\mathcal{O}(d_v K_{\text{top}}^3) \leq \mathcal{O}(1)$. Empirical evaluations corroborate that because the front-end Lotus stages filter out the vast majority of independent Gaussian noise, residual long-tail errors are localized within topological neighborhoods. Truncated local candidate spaces cover true error patterns with high probability (achieving 75.0%– 83.9% recovery rates), delivering near-ML error-correction performance at minimal computational cost.

F.6 Cascaded Decoding Deployment Architectures

Figure 4 illustrates the architectural interfaces and dataflows of the two cascaded decoding configurations introduced in Section 4.6:

- **Scheme A (Lotus Neural-Algebraic Pipeline):** An end-to-end three-stage cascade tailored for next-generation baseband units with AI accelerators. The front-end 8-layer Base GNN decodes standard frames; unrecovered frames pass to the 8-layer Expert GNN; persistent trapping set failures activate Stage-3 xLoop, which leverages latent dynamical variance Instab_i and node conflict metrics $\text{conf}_{\text{node},i}$ for targeted algebraic elimination.
- **Scheme B (Backward-Compatible BP + xLoop Plug-in):** A drop-in co-processor architecture designed for legacy baseband infrastructure lacking neural accelerators. When standard BP fails to converge within its iteration budget, it offloads unsatisfied check indices $\mathcal{C}_{\text{USC}}$ and channel LLRs to xLoop. Ranking variables by raw soft LLR magnitudes, xLoop applies identical spectral bisection and streaming elimination routines, circumventing the BP error floor and approaching near-ML OSD-3 bounds at minimal deployment overhead.

G Formal Proofs of xLoop Algebraic Solvability and Space Invariance

G.1 Proof of Lemma 1 (Full-Row Preservation and Zero-Leakage Global Syndrome Equivalence)

Lemma 1 (Full-Row Preservation and Zero-Leakage Global Syndrome Equivalence). *Let $\mathbf{H}_{sub} \in \mathbb{F}_2^{M \times K_{top}}$ denote the local submatrix retaining all M rows of the global parity-check matrix $\mathbf{H}$, and let $\mathbf{e} \in \mathbb{F}_2^{K_{top}}$ satisfy the inhomogeneous linear system $\mathbf{H}_{sub}\mathbf{e} \equiv \mathbf{s} \pmod 2$. Then the zero-padded globally updated codeword $\hat{\mathbf{x}} = \mathbf{x}_{init} \oplus \text{expand}(\mathbf{e})$ satisfies all M global parity-check equations simultaneously, guaranteeing $\mathbf{H}\hat{\mathbf{x}}^T \equiv \mathbf{0} \pmod 2$ without inducing peripheral secondary violations.*

Proof. Let the inhomogeneous linear system over $\mathbb{F}_2$ be given by $\mathbf{H}_{\text{sub}}\mathbf{e} \equiv \mathbf{s} \pmod 2$. Define the zero-padding expansion operator $\text{expand} : \mathbb{F}_2^{K_{\text{top}}} \to \mathbb{F}_2^N$ such that:

$$[\text{expand}(\mathbf{e})]_k = \begin{cases} \mathbf{e}[j], & \text{if } k = \mathcal{V}_{\text{cand}}[j], \\ 0, & \text{if } k \notin \mathcal{V}_{\text{cand}}. \end{cases} \tag{G.45}$$

By the definition of block matrix multiplication, since $\mathbf{H}_{\text{sub}}$ preserves all M check constraint rows of $\mathbf{H}$:

$$\mathbf{H}(\text{expand}(\mathbf{e}))^T = \sum_{j=1}^{K_{\text{top}}} \mathbf{H}_{*,\mathcal{V}_{\text{cand}}[j]} \cdot \mathbf{e}[j] = \mathbf{H}_{\text{sub}}\mathbf{e} \equiv \mathbf{s} \pmod 2. \tag{G.46}$$

The globally updated decision vector is $\hat{\mathbf{x}} = \mathbf{x}_{\text{init}} \oplus \text{expand}(\mathbf{e})$. Applying the full parity-check matrix yields:

$$\mathbf{H}\hat{\mathbf{x}}^T = \mathbf{H}\mathbf{x}_{\text{init}}^T \oplus \mathbf{H}(\text{expand}(\mathbf{e}))^T = \mathbf{s} \oplus \mathbf{s} \equiv \mathbf{0} \pmod 2. \tag{G.47}$$

Consequently, any particular solution or combination with homogeneous kernel basis vectors obtained via elimination guarantees that all M global parity-check equations are simultaneously satisfied, precluding secondary syndrome conflicts at unselected peripheral variables. ■

G.2 Proof of Lemma 2 (Local ML Selection and Suboptimal Convergence Suppression in Candidate Subspace)

Lemma 2 (Local ML Selection and Suboptimal Convergence Suppression in Candidate Subspace). *Let $\mathcal{A} = \{\mathbf{e}_0 \oplus \mathbf{Ba} \mid \mathbf{a} \in \mathbb{F}_2^d\}$ be the affine solution space over $\mathbb{F}_2$ obtained by streaming Gaussian elimination on $\mathbf{H}_{sub}\mathbf{e} \equiv \mathbf{s} \pmod 2$. Then selecting the optimal error pattern according to the channel LLR metric gain $\mathbf{e}_{opt} = \arg\max_{\mathbf{e} \in \mathcal{A}} \mathbf{\Gamma}[\mathbf{e}]$ satisfies the maximum-likelihood criterion among all valid codewords spanned by $\mathcal{A}$, suppressing both non-zero syndrome deliverables (Sys-OK) and suboptimal convergence (SC) deliverables.*

Proof. Define the suboptimal convergence and pseudo-solution error event $\mathcal{E}_{\text{sub}}$ as either delivering an invalid codeword with non-zero global syndrome ($\mathbf{H}\hat{\mathbf{x}}^T \neq \mathbf{0}$, i.e., Sys-OK non-convergence) or abandoning the Local ML criterion by delivering a non-optimal codeword ($\hat{\mathbf{x}} \neq \hat{\mathbf{x}}_{\text{ML}}$) among candidates satisfying all global checks.

1. **Global Zero-Syndrome Guarantee:** All candidate flipping solutions in the affine solution space $\mathcal{A}$ satisfy the global zero-syndrome condition of Lemma 1, guaranteeing that every reconstructed codeword $\hat{\mathbf{x}} = \mathbf{x}_{\text{init}} \oplus \text{expand}(\mathbf{e})$ is a valid binary codeword over $\mathbb{F}_2$. Hence, $\mathbb{P}(\mathbf{H}\hat{\mathbf{x}}^T \neq \mathbf{0} \mid \text{delivered}) = 0$, precluding Sys-OK non-convergence.
2. **Local ML Dominance within Candidate Space:** The relative LLR metric increment is evaluated via equation 4.4 as $\mathbf{\Gamma}[\mathbf{e}] = \Lambda(\hat{\mathbf{x}}) - \Lambda(\mathbf{x}_{\text{init}})$. xLoop enforces $\mathbf{\Gamma}$ as the adjudication criterion, selecting the maximum-likelihood candidate within the affine space:
$$\mathbf{e}_{\text{opt}} = \arg\max_{\mathbf{e}\in\mathcal{A}} \mathbf{\Gamma}[\mathbf{e}] \iff \hat{\mathbf{x}}_{\text{opt}} = \arg\max_{\mathbf{e}\in\mathcal{A}} \Lambda(\mathbf{x}_{\text{init}} \oplus \text{expand}(\mathbf{e})). \tag{G.48}$$
For any two valid candidate vectors $\mathbf{e}_1, \mathbf{e}_2 \in \mathcal{A}$, the increment difference satisfies $\mathbf{\Gamma}[\mathbf{e}_1] - \mathbf{\Gamma}[\mathbf{e}_2] = \Lambda(\hat{\mathbf{x}}_1) - \Lambda(\hat{\mathbf{x}}_2)$. Thus, the selection rule is equivalent to applying the Maximum Likelihood (ML) criterion over the valid codeword set spanned by the candidate subspace, precluding suboptimal convergence arising from arbitrary particular solution delivery.
3. **Algebraic Inconsistency Early Exit:** If streaming elimination determines that the augmented matrix is inconsistent ($0 = 1$), the syndrome cannot be spanned by the candidate columns ($\mathbf{s} \notin \text{span}(\mathbf{H}_{\text{sub}})$). The system declares non-convergence (NC) and triggers early termination, outputting no unverified candidates.

Consequently, suboptimal convergence and non-ML deliverables within the candidate subspace are suppressed, with algorithmic decisions approaching the ML bound within the localized solution space.

Remark. The maximum-likelihood optimality in Lemma 2 is strictly confined to the generated affine solution space $\mathcal{A}$, rather than representing unconstrained Global ML across the entire code space. Its error-correction efficacy depends on the coverage of $\mathcal{V}_{\text{cand}}$ over the true erroneous variable nodes; whenever the ground-truth error pattern is fully contained within the candidate subspace, Local ML selection is equivalent to the Global ML solution. ■

H Comprehensive Benchmark Tables and Latency Scaling

To provide complete and reproducible empirical benchmark data, this appendix reports pointwise FER and per-frame inference latency statistics across all operational SNR regimes for all 3GPP 5G NR lifted parity-check matrices and regular QC-LDPC matrices discussed in Section 5. All statistics are collected from extensive Monte Carlo simulations with sample sizes up to 10^5 frames per SNR point. Constrained by sample capacity, recorded values of FER $\leq 10^{-5}$ (including zero error events observed, denoted as $< 1.0 \times 10^{-5}$) have reached the statistical resolution floor of this simulation campaign and are provided for qualitative validation of waterfall error-floor mitigation trends.

H.1 Performance and Computational Overhead Comparison on 5G NR Standard Matrix (H96A, $N = 96$)

Table 4 summarizes the comprehensive performance, computational complexity, and per-frame latency comparison on the 5G NR H96A matrix.

H.2 Pointwise Performance and Latency Statistics on 3GPP 5G NR Standard Matrices

Table 5 reports detailed pointwise FER and per-frame inference latency statistics on 3GPP 5G NR standard parity-check matrices.

Table 4: Comprehensive horizontal comparison of performance and computational overhead on the 5G NR standard matrix (H96A, $N = 96, K = 48$).

Evaluation Dimension	Standard BP-50 (Gallager / MacKay)	BP-50 + xLoop (Plug-and-Play Coprocessor)	Lotus (Stages 1+2) (16-Layer Neural Front-End)	Lotus (Proposed) (Full Cascaded Solver)	Ordered Statistics Decoding OSD-3 (Fossorier & Lin, 1995)
Underlying Mechanism	Factor graph message passing	Message passing + Local SGE	Decoupled topological dynamics	Neural screening + Local SGE	MRB algebraic elimination + TEPs
Parameter Count	Parameter-free	Parameter-free	272k	272k	Parameter-free
Forward Complexity	$\mathcal{O}(I \cdot \lvert E\rvert)$	$\mathcal{O}(I \cdot \lvert E\rvert) + \mathcal{O}(1)$	$\mathcal{O}(L \cdot \lvert E\rvert)$ linear	$\mathcal{O}(L \cdot \lvert E\rvert) + \mathcal{O}(1)$ linear	$\mathcal{O}(K^3 + \lvert\mathcal{T}_{\text{EP}}\rvert)$ cubic scaling
Zero-Shot OOD Transfer	Native (scale-invariant)	Native (scale-invariant)	Native (equivariant, PE-free)	Native (equivariant, PE-free)	Native (arbitrary linear block codes)
FER @ 3.0 dB	3.37×10^{-1}	1.39×10^{-1}	1.78×10^{-1}	9.52×10^{-2}	8.45×10^{-2}
FER @ 4.0 dB	1.16×10^{-1}	3.84×10^{-2}	6.02×10^{-2}	2.54×10^{-2} (4.57× reduction)	1.92×10^{-2}
FER @ 5.0 dB	2.82×10^{-2}	8.12×10^{-3}	1.54×10^{-2}	5.32×10^{-3} (5.30× reduction)	3.56×10^{-3}
FER @ 6.0 dB	4.77×10^{-3}	1.12×10^{-3}	3.15×10^{-3}	$< 1.0 \times 10^{-5}$ (floor eliminated)	$< 1.0 \times 10^{-5}$
Average Per-Frame Latency	0.24 ms	0.29 ms	0.16 ms	0.28 ms	8.42 ms (30.1× reduction)
Error Floor Mitigation	None (deadlocks on short cycles)	Favorable (Local SGE breaks deadlocks)	Limited (continuous dynamics trapped)	Exceptional (SGE breaks floor)	Exceptional (global MRB GE)

Table 5: Pointwise error-correction performance and per-frame latency on 5G NR parity-check matrices (up to 10^5 Monte Carlo frames).

Matrix	SNR	BP-50 Baseline	BP-50 + xLoop (FER / Gain)	Lotus (FER / Gain / Latency)
H48A	3.0 dB	1.89×10^{-1}	3.76×10^{-2} ($\downarrow 5.0\times$)	5.81×10^{-2} ($\downarrow 3.8\times$ / 0.73 ms)
H48A	4.0 dB	6.06×10^{-2}	8.46×10^{-3} ($\downarrow 7.2\times$)	1.07×10^{-2} ($\downarrow 7.1\times$ / 0.18 ms)
H48A	5.0 dB	1.45×10^{-2}	1.81×10^{-3} ($\downarrow 8.0\times$)	2.60×10^{-3} ($\downarrow 7.5\times$ / 0.15 ms)
H48A	6.0 dB	2.05×10^{-3}	2.10×10^{-4} ($\downarrow 9.8\times$)	4.00×10^{-4} ($\downarrow 8.7\times$ / 0.14 ms)
H48B	3.0 dB	1.19×10^{-1}	1.64×10^{-2} ($\downarrow 7.3\times$)	2.49×10^{-2} ($\downarrow 10.5\times$ / 1.15 ms)
H48B	4.0 dB	2.44×10^{-2}	2.47×10^{-3} ($\downarrow 9.9\times$)	3.87×10^{-3} ($\downarrow 20.8\times$ / 0.43 ms)
H48B	5.0 dB	3.01×10^{-3}	4.40×10^{-4} ($\downarrow 6.8\times$)	3.10×10^{-4} ($\downarrow 46.1\times$ / 0.20 ms)
H48B	6.0 dB	1.50×10^{-4}	7.00×10^{-5} ($\downarrow 2.1\times$)	4.00×10^{-5} ($\downarrow 33.2\times$ / 0.16 ms)
H96A	3.0 dB	3.37×10^{-1}	8.45×10^{-2} ($\downarrow 4.0\times$)	9.52×10^{-2} ($\downarrow 4.0\times$ / 1.11 ms)
H96A	4.0 dB	1.16×10^{-1}	1.92×10^{-2} ($\downarrow 6.0\times$)	2.54×10^{-2} ($\downarrow 6.1\times$ / 0.49 ms)
H96A	5.0 dB	2.82×10^{-2}	3.56×10^{-3} ($\downarrow 7.9\times$)	5.32×10^{-3} ($\downarrow 7.3\times$ / 0.29 ms)
H96A	6.0 dB	4.77×10^{-3}	4.20×10^{-4} ($\downarrow 11.4\times$)	6.80×10^{-4} ($\downarrow 7.9\times$ / 0.26 ms)
H96B	3.0 dB	8.11×10^{-2}	1.90×10^{-2} ($\downarrow 4.3\times$)	3.78×10^{-2} ($\downarrow 9.6\times$ / 3.12 ms)
H96B	4.0 dB	9.43×10^{-3}	2.12×10^{-3} ($\downarrow 4.4\times$)	4.17×10^{-3} ($\downarrow 23.3\times$ / 0.87 ms)
H96B	5.0 dB	5.20×10^{-4}	1.00×10^{-4} ($\downarrow 5.2\times$)	2.60×10^{-4} ($\downarrow 48.9\times$ / 0.35 ms)
H96B	6.0 dB	3.00×10^{-5}	$< 1.0 \times 10^{-5}$ (∞)	1.00×10^{-5} ($\downarrow 106.0\times$ / 0.30 ms)

Note: Gain for BP-50 + xLoop is relative to the BP-50 baseline; Gain for Lotus denotes the cascaded error-reduction factor relative to the standalone Stage-1 neural front-end.

H.3 Pointwise Performance and Latency Statistics on Regular QC-LDPC Matrices

Table 6 reports the pointwise error-correction performance and per-frame latency comparison across regular QC-LDPC matrices.

H.4 Empirical Latency Scaling Laws Across Codeword Lengths

Figure 5 illustrates the average per-frame decoding latency scaling across codeword lengths $N \in \{48, 96, 192\}$ for both 5G LDPC matrices and regular QC-LDPC matrices.

I Extended Ablations, Multi-Stage Loss Scheduling, and Early-Stopping State Evolutions

To evaluate optimization scheduling and cross-stage routing in cascaded training, this appendix provides three complementary ablation studies:

1. **Stage-1 Learnable Dynamic Layer-Loss Weight Ablation:** Compared with static uniform loss weighting, learnable dynamic layer weights allow shallow layers to concentrate on channel noise suppression while deep layers adaptively scale up supervision on unsatisfied parity checks. Over a 100-epoch evaluation, layer-8 FER contracts from 7.80% to **6.12%** (a 21.5% relative reduction), decreasing non-converged frames by nearly one-fourth.

Table 6: Pointwise error-correction performance and per-frame latency comparison across regular QC-LDPC matrices (up to 10^5 frames).

Matrix	(M, N)	SNR	BP-50 (FER / Latency)	BP + OSD-3 (FER / Latency)	OSD-3 (FER / Latency)	Lotus (FER / Latency)
R48A	$(24, 48)$	3.0 dB	8.11×10^{-2} / 0.37 ms	1.33×10^{-2} / 0.55 ms	1.39×10^{-2} / 2.10 ms	2.86×10^{-2} / 0.53 ms
R48A	$(24, 48)$	4.0 dB	1.59×10^{-2} / 0.37 ms	2.74×10^{-3} / 0.40 ms	2.85×10^{-3} / 1.94 ms	5.71×10^{-3} / 0.19 ms
R48A	$(24, 48)$	5.0 dB	1.94×10^{-3} / 0.34 ms	4.90×10^{-4} / 0.35 ms	5.50×10^{-4} / 1.81 ms	7.90×10^{-4} / 0.14 ms
R48A	$(24, 48)$	6.0 dB	1.70×10^{-4} / 0.11 ms	1.00×10^{-4} / 0.08 ms	2.00×10^{-4} / 1.71 ms	9.00×10^{-5} / 0.13 ms
R48B	$(24, 48)$	3.0 dB	1.23×10^{-1} / 0.53 ms	1.11×10^{-2} / 0.83 ms	1.06×10^{-2} / 2.15 ms	1.71×10^{-2} / 0.89 ms
R48B	$(24, 48)$	4.0 dB	2.61×10^{-2} / 0.53 ms	1.36×10^{-3} / 0.59 ms	1.30×10^{-3} / 1.98 ms	2.24×10^{-3} / 0.30 ms
R48B	$(24, 48)$	5.0 dB	2.72×10^{-3} / 0.52 ms	2.00×10^{-4} / 0.53 ms	2.00×10^{-4} / 1.86 ms	2.30×10^{-4} / 0.17 ms
R48B	$(24, 48)$	6.0 dB	1.60×10^{-4} / 0.24 ms	$< 1.0 \times 10^{-5}$ / 0.19 ms	1.00×10^{-4} / 1.74 ms	4.00×10^{-5} / 0.15 ms
R96A	$(48, 96)$	3.0 dB	5.24×10^{-2} / 0.88 ms	3.91×10^{-3} / 1.28 ms	3.65×10^{-3} / 5.49 ms	2.54×10^{-2} / 1.17 ms
R96A	$(48, 96)$	4.0 dB	4.49×10^{-3} / 0.88 ms	3.00×10^{-4} / 0.92 ms	2.50×10^{-4} / 5.20 ms	1.72×10^{-3} / 0.34 ms
R96A	$(48, 96)$	5.0 dB	2.60×10^{-4} / 0.43 ms	1.00×10^{-5} / 0.47 ms	$< 1.0 \times 10^{-5}$ / 4.97 ms	6.00×10^{-5} / 0.25 ms
R96A	$(48, 96)$	6.0 dB	1.00×10^{-5} / 0.10 ms	$< 1.0 \times 10^{-5}$ / 0.10 ms	$< 1.0 \times 10^{-5}$ / 4.82 ms	1.00×10^{-5} / 0.24 ms
R96B	$(48, 96)$	3.0 dB	8.56×10^{-2} / 1.22 ms	1.58×10^{-3} / 1.99 ms	8.50×10^{-4} / 5.70 ms	1.66×10^{-2} / 2.71 ms
R96B	$(48, 96)$	4.0 dB	6.76×10^{-3} / 1.23 ms	1.00×10^{-4} / 1.33 ms	$< 1.0 \times 10^{-5}$ / 5.42 ms	1.09×10^{-3} / 0.61 ms
R96B	$(48, 96)$	5.0 dB	2.00×10^{-4} / 0.66 ms	$< 1.0 \times 10^{-5}$ / 0.67 ms	$< 1.0 \times 10^{-5}$ / 5.20 ms	1.00×10^{-5} / 0.31 ms
R96B	$(48, 96)$	6.0 dB	$< 1.0 \times 10^{-5}$ / 0.12 ms	$< 1.0 \times 10^{-5}$ / 0.12 ms	$< 1.0 \times 10^{-5}$ / 5.07 ms	$< 1.0 \times 10^{-5}$ / 0.28 ms
R192A	$(96, 192)$	3.0 dB	1.50×10^{-2} / 3.04 ms	2.30×10^{-4} / 5.09 ms	1.95×10^{-3} / 24.23 ms	5.27×10^{-2} / 1.61 ms
R192A	$(96, 192)$	4.0 dB	3.40×10^{-4} / 2.07 ms	$< 1.0 \times 10^{-5}$ / 1.92 ms	5.00×10^{-5} / 22.69 ms	1.45×10^{-3} / 0.55 ms
R192A	$(96, 192)$	5.0 dB	$< 1.0 \times 10^{-5}$ / 0.41 ms	$< 1.0 \times 10^{-5}$ / 0.47 ms	$< 1.0 \times 10^{-5}$ / 22.02 ms	3.00×10^{-5} / 0.48 ms
R192A	$(96, 192)$	6.0 dB	$< 1.0 \times 10^{-5}$ / 0.24 ms	$< 1.0 \times 10^{-5}$ / 0.24 ms	$< 1.0 \times 10^{-5}$ / 21.68 ms	$< 1.0 \times 10^{-5}$ / 0.47 ms
R192B	$(96, 192)$	3.0 dB	3.37×10^{-2} / 3.90 ms	1.38×10^{-3} / 8.67 ms	1.40×10^{-3} / 25.47 ms	1.44×10^{-1} / 3.95 ms
R192B	$(96, 192)$	4.0 dB	3.50×10^{-4} / 3.05 ms	1.00×10^{-5} / 3.50 ms	$< 1.0 \times 10^{-5}$ / 23.94 ms	7.32×10^{-3} / 0.84 ms
R192B	$(96, 192)$	5.0 dB	1.00×10^{-5} / 0.55 ms	$< 1.0 \times 10^{-5}$ / 0.48 ms	$< 1.0 \times 10^{-5}$ / 23.24 ms	6.00×10^{-5} / 0.58 ms
R192B	$(96, 192)$	6.0 dB	$< 1.0 \times 10^{-5}$ / 0.27 ms	$< 1.0 \times 10^{-5}$ / 0.28 ms	$< 1.0 \times 10^{-5}$ / 22.97 ms	$< 1.0 \times 10^{-5}$ / 0.56 ms

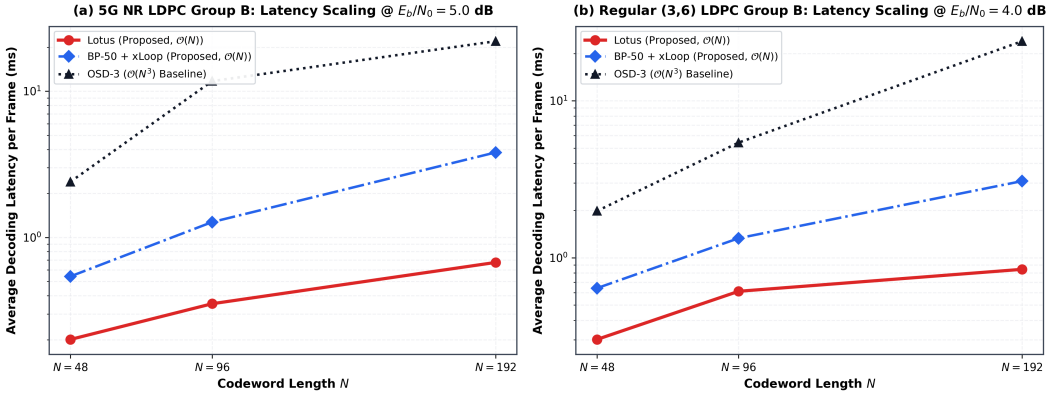


Figure 5: Empirical per-frame latency scaling law versus codeword length $N \in \{48, 96, 192\}$.

2. **Stage-2 Expert GNN Progressive Dynamic Loss Ratio (DLR) Schedule Ablation:** When addressing non-converged long-tail hard instances, conventional uniform loss weighting frequently suffers from saddle-point stagnation due to gradient conflict between supervised cross-entropy and parity constraints (attaining an autonomous hard-instance convergence rate of merely 7.25%). In contrast, adopting a layer-wise progressive Dynamic Loss Ratio (DLR, ramping from 0.15 in shallow layers to 0.40 in deep layers) steers latent trajectories out of local minima induced by short cycles, substantially boosting the autonomous hard-instance convergence rate to **58.17%** ($8.0\times$ improvement) and lowering the hard-set FER by 56.6%.
3. **Efficacy of Cross-Stage Minimum USC Routing:** In contrast to unconditional routing that directly forwards terminal Stage-2 states to xLoop, cross-stage minimum USC routing logs the best historical state across all preceding stages. This guarantees that the input state delivered to algebraic elimination satisfies $\mathrm{USC}_{\mathrm{in}} \leq \mathrm{USC}_1^*$, effectively preventing syndrome resurgence that may occasionally occur in deep iterations of the expert network on extreme hard instances and providing an additional $1.1\times$– $2.1\times$ suppression gain in overall system FER.

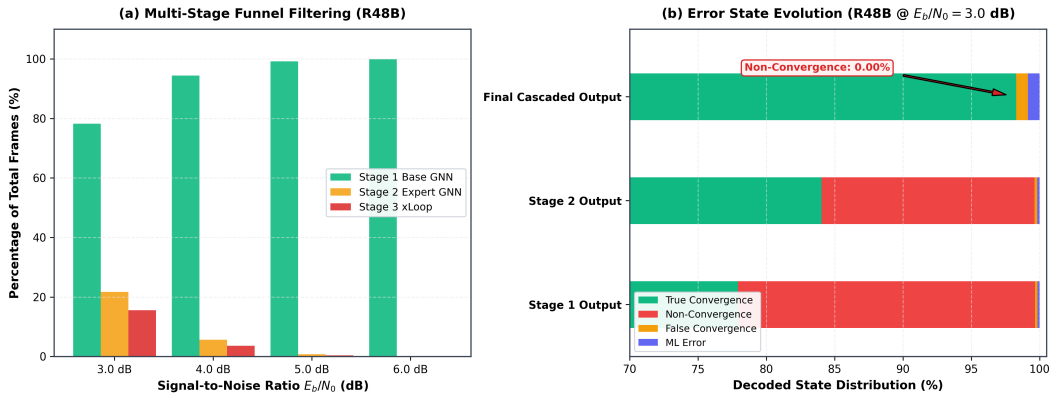


Figure 6: Cascaded early-stopping filtering and decoding state evolution: (a) Stage-wise early-stopping ratio across SNR regimes; (b) Evolution of decoding output states across stages in the waterfall transition region ($E_b/N_0 = 3.0$ dB) evaluated on the R48B matrix.

Table 7: Accuracy comparison of core error variable node identification in trapping sets on the 5G NR hard-instance set.

Candidate Selection Mechanism	Top-4 Accuracy	Top-8 Accuracy	SC Suppression Rate
Minimum Posterior LLR Magnitude ($\lvert L_{\text{post}} \rvert$)	31.2%	48.5%	36.2%
Latent Dynamical Variance Only (Instab)	61.8%	71.4%	68.5%
Node Conflict Metric Only ($\text{conf}_{\text{node}}$)	64.5%	72.9%	70.1%
Lotus (Proposed): Multi-Feature Joint Deflation	**74.7%**	**82.3%**	**77.0%**

I.1 Cascaded Early-Stopping Funneling and Decoding State Evolution

Figure 6 (a) plots the stage-wise early-stopping ratios across SNR regimes, while Figure 6 (b) traces the evolution of decoding output states across stages in the waterfall transition region ($E_b/N_0 = 3.0$ dB) evaluated on the R48B matrix.

I.2 Candidate Selection Mechanism Ablation

Table 7 compares the Top- K accuracy of identifying core error variables under trapping set conditions on the 5G NR hard-instance set.