Self-Organising Digital Circuits

Marcello Barylli[1]*, Gabriel Béna[2]*, Alexander Mordvintsev[3], Eleni Nisioti[1], and Sebastian Risi[1]

[1]IT University of Copenhagen
[2]Imperial College London
[3]Google, Paradigms of Intelligence Team
bary@itu.dk, g.bena21@imperial.ac.uk

Abstract

Fault tolerance in classical computing has traditionally relied on static strategies like hardware redundancy and error-correcting codes. Biological systems, in contrast, exhibit adaptive plasticity, maintaining function through dynamic re-organisation around damage. Inspired by this principle, we introduce Self-Organising Digital Circuits, framing functional logic generation and maintenance as a meta-learning problem on graphs. Our architecture employs a topology-masked Transformer that configures the Lookup Tables (LUT) of a circuit's Boolean gates. Extending the pattern-generation paradigm of Neural Cellular Automata (NCA), it navigates the degenerate Boolean search space to satisfy a computational task, rather than regenerating a fixed target state. We demonstrate that it can self-assemble functional circuits from scratch and rapidly re-route logic around permanent, previously unseen hardware faults. For soft errors, the policy achieves near-perfect recovery (>99.99% accuracy) from damage sizes far exceeding training conditions. We further observe generalisation across circuit scales: accuracy improves on graphs substantially wider than those seen during training. This work bridges the principles of biological self-organisation with the practical domain of digital hardware.

Submission type: **Full Paper**

Data/Code + **Circuit Visualization** available at: `https://github.com/GabrielBena/boolean_nca_cc/tree/gabi`

Introduction

Robustness is a central tenet of modern informatics. Modern systems are capable of impressive fault tolerance, using mechanisms such as Error Correcting Codes (ECC), modular redundancy, and sophisticated fallback protocols to ensure graceful degradation (Neumann, 1956; Woods and Lightbody, 2008). However, these engineered successes typically rely on *anticipation*: resources are pre-allocated and failure modes are modeled in advance. When a system encounters damage that exceeds its redundancy budget or defies its failure model, it often fails brittly.

Biological intelligence offers a complementary paradigm: *adaptive plasticity*. Natural automata, such as the mammalian cortex, do not rely solely on static backups. Instead, they maintain function by dynamically re-purposing surviving components to compensate for injury, a phenomenon known as cortical remapping (Nudo, 2013). This form of resilience is not pre-programmed but emergent, and it is enabled by the *degeneracy* of biological networks: the availability of structurally distinct yet functionally equivalent pathways (Edelman and Gally, 2001).

Inspired by these biological principles, and by von Neumann's early postulate that systems should "operate across errors" (von Neumann, 1966), we aim to provide digital hardware with similar capabilities. As emphasised by Woods et al. (Woods and Lightbody, 2008), true hardware autonomy is essential for remote deployments such as deep-space missions out of contact with Earth. This requires moving beyond simple redundancy toward organism-like self-healing, where a collective maintains function despite the loss of individual components. However, in current reconfigurable computing systems like Field-Programmable Gate Arrays (FPGAs), this error mitigation is limited by the 'reconfiguration penalty': the overhead of globally monitoring faults and remapping logic(Woods and Lightbody, 2008).

To address this, we introduce a framework for self-organising digital circuits that replaces global, supervised reconfiguration with a decentralised local policy. We extend the Neural Cellular Automata (NCA) paradigm from pattern formation to functional logic generation. In an NCA, simple agents communicate only with their local neighbours, yet collectively produce globally coordinated behaviour. We apply this principle to a substrate of programmable logic gates, where each agent corresponds to a gate whose truth table is a differentiable parameter. When a gate fails, either due to being reversibly corrupted or permanently damaged, the circuit must be reconfigured. The goal of reconfiguration is to recover the original input-output function (fig. 1).

*Denotes equal contribution.

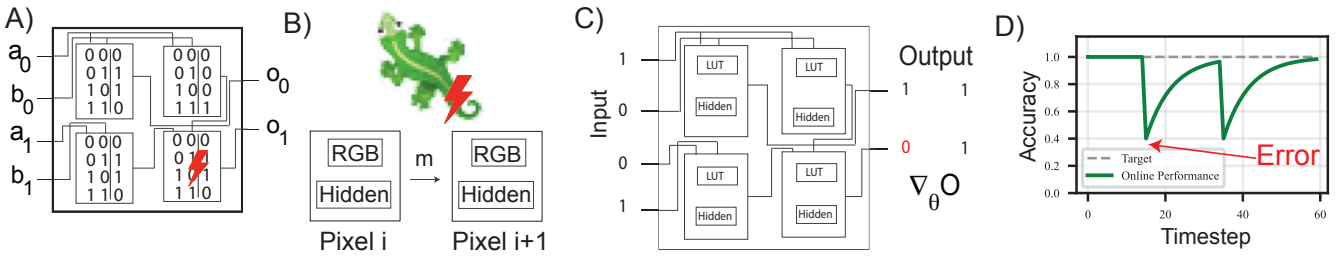


Figure 1: **From Self-Organising Images to Self-Organising Circuits.** A) A digital circuit of LUT-based gates can be corrupted by errors that flip LUT contents. B) NCAs achieve self-organising, self-repairing images via local message passing (Mordvintsev et al., 2020). C) We extend this paradigm to circuits: each LUT maintains a hidden state and updates its logic through neighbour communication. D) At deployment, the trained local policy repairs damage without global supervision.

Our architecture is meta-learned in two phases. The local update rule is trained offline via backpropagation through time, exploiting the differentiability of the continuous LUT relaxation. Once trained, it optimises circuits using only local forward passes, so the expensive global optimisation is paid once and the resulting policy can assemble and repair circuits without differentiable hardware components (Sunada et al., 2025).

Key Contributions

- We introduce the Topology-Masked Transformer (TMT), a shared-weight attention block applied recurrently under a binary wiring mask, amortising circuit optimisation into a deployment-time forward pass with no differentiable hardware required.
- The TMT achieves >99.99% recovery on soft-error patterns up to 5× larger than seen in training, with an edit size independent of perturbation size.
- Policies trained on random topologies generalise to unseen larger circuits of 1.7x the training width (264 → 450+ nodes).

Background and Related Work

Evolvable Hardware. Evolvable hardware (EHW) applies evolutionary algorithms to discover or repair circuit configurations on reconfigurable substrates such as FPGAs (Thompson, 1997; Whitley et al., 2021). EHW has demonstrated circuit evolution and online fault recovery, sharing our core objective of autonomous hardware adaptation. However, evolutionary search incurs its full computational cost for each individual circuit: chromosome lengths grow with circuit complexity, and the resulting search spaces demand substantial resources even for moderately sized designs (Haddow and Tyrrell, 2011). Our framework addresses this by amortising the cost of optimisation across all circuits seen during training, replacing stochastic population-based search with a single deterministic forward pass at deployment time.

From Pattern Formation to Functional Substrates. Neural Cellular Automata (NCA) parameterise local update rules with neural networks, enabling self-organising pattern formation on grids (Mordvintsev et al., 2020). Recent extensions have generalised NCAs to arbitrary graph topologies (Grattarola et al., 2021) and to dynamic computational tasks (Béna et al., 2025), but the automaton's state always remains the data itself.

We propose a fundamental shift: viewing the cellular state as the *functional substrate* that processes data. Each cell becomes a programmable logic gate whose truth table is optimised by a learned local policy, rendering the circuit a self-organising system. Because the gates converge to discrete Boolean logic, the resulting circuits remain amenable to formal verification (Kresse et al., 2025).

Concurrently, Miotti et al. (Miotti et al., 2025) use differentiable logic gates to parameterise NCA update rules for hardware-native implementations; our goal is orthogonal, as we use an NCA-like model to optimise the gates themselves.

Attention-Based Local Policies. Graph Attention Networks (Veličković et al., 2018) introduced learnable, content-dependent message weighting on graphs. Subsequent Graph Transformer architectures augment attention with structural encodings and edge features (Dwivedi and Bresson, 2021). Our approach is deliberately more minimal: we apply a single shared-weight Transformer block whose attention is restricted to wired neighbours via a binary topology mask, with no additional structural encoding. Applied recurrently, this topology-masked Transformer (TMT) internalises a local search policy that replaces explicit global optimisers, connecting to recent work on meta-trained in-context learners (Kirsch and Schmidhuber, 2022; Kirsch et al., 2024).

Approach

Problem Formulation

We model a digital circuit as a Directed Acyclic Graph (DAG) of programmable Look-Up Tables (LUTs) connected by fixed wires (see fig. 1A). The circuit receives a global binary input $\mathbf{x} \in \{0,1\}^{N_{in}}$ and produces an output $\hat{\mathbf{y}} \in \{0,1\}^{N_{out}}$. Each gate has arity k and is parameterised by

a LUT of 2^k entries that fully specify its Boolean function. Wires are integer indices selecting bits from the previous layer's outputs, allowing arbitrary connectivity and fan-out. Gates are arranged in feed-forward layers; for our 12-bit tasks with $k = 4$, this yields a three-hidden-layer architecture of sizes $(96, 96, 48)$.

During deployment, two categories of hardware fault may corrupt a gate's LUT: **(i) Recoverable (soft) errors**, which flip LUT entries but can be overwritten, and **(ii) Permanent (stuck-at) faults**, which clamp a gate's output irreversibly. Given a target Boolean function f, the goal is to learn a *decentralised local policy* that configures the LUTs such that $\hat{\mathbf{y}} = f(\mathbf{x})$, and that can autonomously restore this mapping after previously unseen faults, without global supervision or backpropagation at deployment time.

Circuits are initialised as noisy "soft wires": each gate's LUT acts as an identity pass-through on one of its inputs (assigned round-robin), with small additive noise to break symmetry.

Model: Topology-Masked Transformer (TMT)

To apply self-organising principles, we lift the circuit into a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where each node v_i corresponds to a gate or input pin (fig. 1C). We then learn a local message-passing policy that discovers communication protocols to satisfy the target logic.

Node State. Each node carries a state vector:

$$\mathbf{s}_i = [\boldsymbol{\ell}_i, \mathbf{m}_i, \mathbf{p}_i].$$

The LUT logits $\boldsymbol{\ell}_i \in \mathbb{R}^{2^k}$ defining the gate's logic, a latent memory $\mathbf{m}_i \in \mathbb{R}^{d_{hidden}}$ ($d_{hidden} = 64$) for recurrent state, and a sinusoidal positional encoding $\mathbf{p}_i$ of the normalised depth d_i/D, which is invariant to circuit scale. Optionally, a scalar feedback signal r_i (described below) is appended, yielding $\mathbf{s}_i = [\boldsymbol{\ell}_i, \mathbf{m}_i, \mathbf{p}_i, r_i]$.

Connectivity. The graph topology mirrors the circuit wiring. In the random-topology regime, connections are generated by randomly permuting previous-layer output indices, ensuring uniform fan-out while preserving the DAG structure. We enforce bidirectional edges: every forward wire $A \to B$ implies a backward message-passing edge $B \to A$.

Update Rule. We parameterise the update rule as a single-block Transformer operating on the N gate tokens simultaneously. We write $\hat{\mathbf{M}}$ for a layer-normalised (Ba et al., 2016) matrix $\mathbf{M}$. The node states $\mathbf{S}$ are projected into a latent space ($d_{attn} = 128$):

$$\mathbf{Z}^{(0)} = \hat{\mathbf{S}}\, W_{in}^T \in \mathbb{R}^{N \times d_{attn}} \tag{1}$$

A single attention block, whose attention is restricted to wired neighbours via a binary topology mask $M \in \{0,1\}^{N \times N}$, refines the latents. We adopt **Pre-LN** normalisation (Xiong et al., 2020) with separate LayerNorms for queries and keys/values, and **QK-normalisation** (Dehghani et al., 2023) to stabilise attention logits across recurrent steps. Residual branches are gated via **ReZero** (Bachlechner et al., 2020): learned scalars α initialised at zero so that the block initially acts as an identity:

$$\mathbf{Z}' = \mathbf{Z}^{(0)} + \alpha_{\text{attn}} \cdot \text{MSA}\Big(\hat{\mathbf{Z}}^{(0)},\ \hat{\mathbf{Z}}^{(0)},\ M\Big) \tag{2}$$

$$\mathbf{Z}_{out} = \mathbf{Z}' + \alpha_{\text{ffn}} \cdot \text{MLP}\Big(\hat{\mathbf{Z}}'\Big) \tag{3}$$

The two-layer MLP (with GeLU activation) acts independently per node; all cross-node communication is confined to the masked attention step. Output latents are decoded into residual parameter updates via α-gated linear heads:

$$\Delta\boldsymbol{\ell}_i = \alpha_\ell \cdot \hat{\mathbf{Z}}_{out,i}\, W_\ell^T, \quad \Delta\mathbf{m}_i = \alpha_m \cdot \hat{\mathbf{Z}}_{out,i}\, W_m^T \tag{4}$$

Positional encodings $\mathbf{p}_i$ remain static; the error feedback r_i is dynamically recomputed at every step. The updates are applied residually ($\mathbf{s}_i^{(t+1)} = \mathbf{s}_i^{(t)} + \Delta\mathbf{s}_i^{(t)}$) by applying this single shared-weight block recurrently for T steps, forming a T-layer weight-tied residual network whose expressivity comes through iterated refinement. The topology mask imposes a strict "speed of light": information propagates at most one hop per step, so global coordination emerges from iterated local interactions, faithful to the NCA paradigm.

Per-Node Error Feedback. For fixed-wiring experiments the architecture above suffices; for random wirings, we found explicit error signals essential. Each output gate receives a scalar r_i: the mean absolute residual over a task batch:

$$r_i = \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}} |y_i - \hat{y}_i(\mathbf{x})| \tag{5}$$

Non-output gates receive $r_i = 0$. This signal propagates upstream through recurrent attention, enabling interior gates to adjust in response to downstream error, forming a decentralised credit assignment.

Scale-Free Architecture. Because the Transformer operates at the node level with shared weights, its parameters are independent of circuit size: the same update rule applies to 20 or 200 gates, 3 or 10 layers. Only the topology mask M must be recomputed from the wiring. Combined with normalised positional encodings, this endows the architecture with *scale-freedom*: a trained policy can, in principle, be deployed on circuits of different size without retraining.

Training

Differentiable Circuit Execution. To train the update rule via backpropagation, we require gradients through the circuit's Boolean logic. Following the differentiable logic gate network (DLGN) paradigm (Petersen et al., 2022), we achieve this through a continuous relaxation: binary signals are represented as probabilities $x \in [0, 1]$, and each gate's 2^k LUT logits are passed through a sigmoid.

Unlike the categorical relaxation of DLGNs, which learns a softmax distribution over the 16 fixed two-input Boolean functions, our formulation directly parameterises the continuous LUT entries. The soft gate output is then computed as a multilinear interpolation of the LUT at the input point $\mathbf{x}$. For instance, with a 2-input gate, the first input x_1 interpolates between the two halves of the LUT to produce an intermediate tensor L':

$$L' = (1 - x_1) \cdot [l_{00}, l_{01}] + x_1 \cdot [l_{10}, l_{11}]$$

The second input x_2 does the same in L' and computes the final scalar output:

$$y = (1 - x_2) \cdot L'[0] + x_2 \cdot L'[1].$$

This interpolation is fully differentiable, and because each LUT entry influences multiple input combinations, gradients flow through the entire function. At inference, rounding to $\{0, 1\}$ recovers discrete Boolean logic.

Pool-Based Meta-Learning. We frame the circuit optimisation problem as a dynamic system trained via Backpropagation Through Time (BPTT). This is meta-learning in the sense of Andrychowicz et al. (2016): The TMT is trained to produce parameters for a *separate* computation (the circuit), with the outer objective being that computation's task loss, a two-level structure absent in standard NCAs, where the cell state *is* the target output. As a baseline, we compare against standard Backpropagation (BP) applied directly to the differentiable LUTs, which serves as an upper bound on circuit performance. Following Mordvintsev et al. (2020), we maintain a persistent graph Pool $\mathcal{P} = \{\mathcal{G}_1, \ldots, \mathcal{G}_K\}$. Circuits are periodically reset to unoptimised states (average lifespan of 128 steps) to enforce a dual curriculum: constant injection of fresh circuits demands rapid self-assembly, while surviving circuits enforce long-term homeostasis under damage injections.

Inner Loop (Inference): At each training step, a batch of circuits and Boolean input–output pairs $(\mathbf{X_{train}}, \mathbf{Y_{train}})$ are sampled. The TMT is applied iteratively for T steps via a differentiable `scan`. Crucially, the circuit is functionally executed at every tick t: updated LUT logits are extracted, the circuit processes the input data, and resulting per-node residuals r_i are written back into the graph state. This creates a closed real-time feedback loop. To balance recurrent expressivity with the computational constraints of scaling batch size and model capacity, we truncate the BPTT horizon to $T = 5$.

Outer Loop (Optimisation): Following the T-step scan, the circuit loss $\mathcal{L}$ (Binary Cross Entropy) is computed against $\mathbf{Y_{train}}$. We support both fixed evaluation at the final step ($t^* = T$) and stochastic evaluation ($t^* \sim \mathcal{U}(T_{\min}, T)$) to capture varying gradient depths and enable curriculum scheduling. Gradients of $\mathcal{L}$ with respect to the TMT parameters are computed via BPTT through both the recurrent message-passing and the functional circuit execution, followed by an Adam (Kingma and Ba, 2017) update.

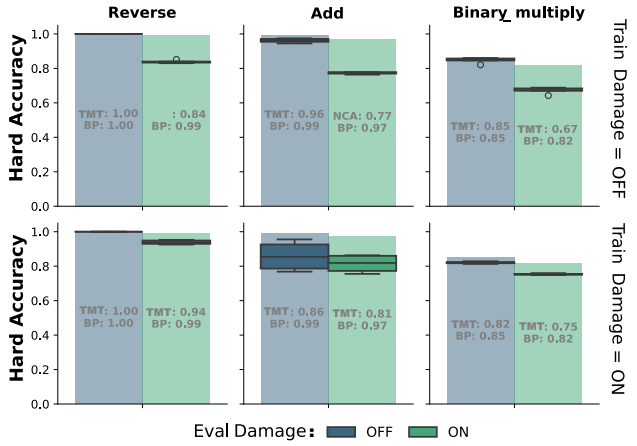


Figure 2: **TMT Performance on Fixed Topologies. Box Plots** represent the test-set accuracy distribution of circuits grown by the NCA policy after 256 steps. The **Faint Background Bars** represent the baseline performance of standard Global Backpropagation (BP) on the same task and regime. The **Blue** elements (Damages OFF during Eval) show performance in ideal conditions. The **Green** elements (Damages ON during Eval) show performance when subjected to stochastic failures. We compare two training regimes: The **Top Row** displays a policy that has never faced damage during training (Damage Training = False); thus, the Green elements represent an out-of-distribution evaluation. The **Bottom Row** displays a policy trained with active damages (Damage Training = True), demonstrating adaptive robustness.

Experiments and Results

Boolean Tasks. We evaluate on three 12-bit tasks (4,096 input–output pairs; 256 held out for testing). **Split Multiplication:** two 6-bit integers are multiplied, occupying the full 12-bit output. **Split Addition:** two 6-bit integers are added (7-bit result, 5 bits zero-padded). **Bit Reversal:** the input array is reversed, mapping bit i to position $11 - i$.

Regime I: Growth, Persistence, and Repair on Fixed Topologies

We first establish that the local TMT policy can grow functional circuits from unoptimised "soft wires" and maintain them indefinitely. To ensure the policy learns the underlying generative function rather than memorising training patterns, all reported accuracies are computed strictly on a held-out test split of 256 unseen inputs.

As shown in fig. 2 (Top Row, Blue), the TMT policy converges across all tasks without damage, achieving performance virtually identical to the global Backpropagation

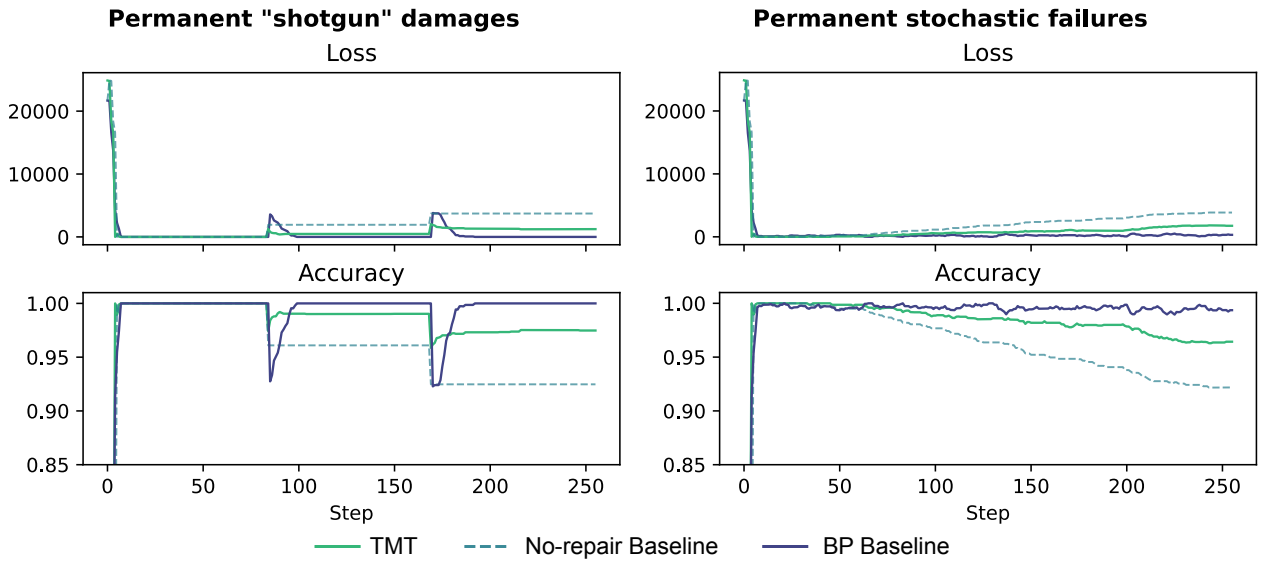


Figure 3: **Resilience to Damage.** Evolution of Loss (left) and Accuracy (right) under permanent damage. The **Solid Green** line represents the TMT-maintained circuit. The **Solid Blue** line represents standard Backpropagation (Upper Bound). The **Dashed** line represents the No-Repair baseline (Lower Bound).

(BP) baseline. It achieves perfect accuracy on Bit Reversal and mean accuracies of 0.96 and 0.84 on Split Addition and Multiplication, respectively.

To evaluate fault tolerance, we introduce stochastic damage (clamping 20% of gates to zero). When exposed to damage out-of-distribution (Zero-Shot Resilience, fig. 2 Top Row, Green), the TMT retains partial functionality, indicating inherent robustness in the distributed representation. When trained with active damage (Learned Resilience, Bottom Row), performance under damage approaches the BP baseline, demonstrating adaptive robustness, albeit with a slight degradation in the damage-free ceiling due to the noisy training environment.

Targeted catastrophic events introduce simultaneous destruction of 10% of its gates, termed "shotgun" in fig. 3. The TMT actively recovers and stabilises at ≈ 0.97 accuracy, vastly outperforming a passive (no-repair) baseline. While global BP (upper bound) recovers more fully, the TMT's decentralised repair minimises the initial impact drop.

Principal Component Analysis (PCA) of the optimisation trajectories (fig. 4) reveals the mechanism of this resilience: under recoverable damage, the TMT does not rewind to a memorised canonical state. Instead, it dynamically reroutes, fanning out into functionally equivalent but structurally distinct local minima.

Regime II: Self-Healing Circuits and the Degenerate Solution Space

To focus on practical considerations, we remove the need to discover functioning circuits and test the TMT purely as a

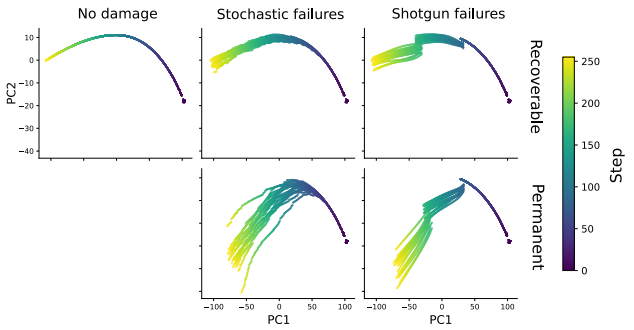


Figure 4: **PCA Trajectories of Circuit Optimisation. (Left)** Without damage, optimisation is deterministic. **(Middle/Right)** Under recoverable or permanent damage, trajectories diverge, driving circuits into a diverse fan of alternative configurations rather than rewinding.

maintenance mechanism on circuits preconfigured to perfect accuracy by BP.

We restrict ourselves to reversible perturbations, modelling the radiation-induced soft errors that dominate over hard faults in aerospace-grade SRAM-based FPGAs (Wang et al., 2024). Figure 5 compares TMT and BP performance upon damage recovery to convergence (26 message steps and 300 gradient steps, respectively), on the **Binary Addition** task. For TMT, it demonstrates near-perfect out-of-distribution recovery for "shotgun" damage sizes far exceeding training parameters, achieving >99.99% hard accuracy across all perturbations. This generalisation highlights

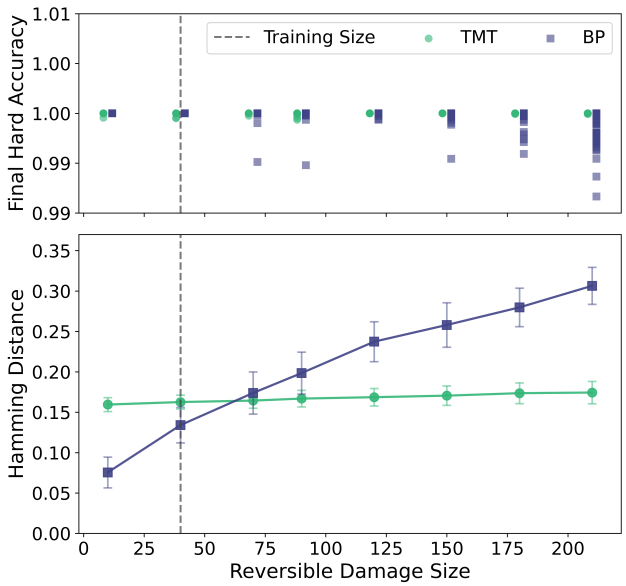


Figure 5: **Out-of-Distribution Perturbation Recovery. (Top)** Hard accuracy of recovered solutions from out-of-distribution damage patterns of increasing size, after training on 40-gate patterns only (dashed v-line). 100 patterns per damage size are shown (20 patterns across 5 seeds). **(Bottom)** Mean per-gate Hamming distance between base and recovered circuits.

the model's relevance for radiation-induced error scenarios. Hamming distances reveal a "signature edit" fraction independent of perturbation size, contrasting with the monotonically increasing response exhibited by BP.

To further stress-test the system, we introduce successive "shotgun" failures of 40 gates at once (17%). Figure 6 demonstrates recovery on the addition task following multiple damage events.

Solution Degeneracy. To map the degenerate solution space of these self-healing circuits (fig. 7), we employ a recursive Depth-First Search (DFS) exploration strategy.

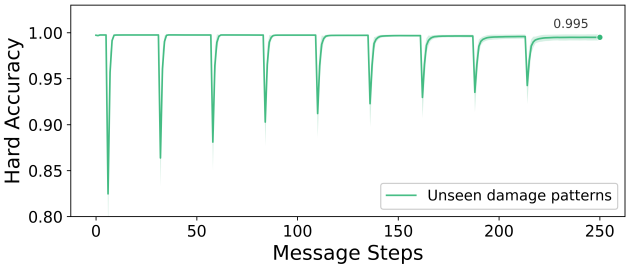


Figure 6: **Recovery Trajectory from Reversible Damage.** Recovery from successive 40-gate damage events, with standard deviation across 256 damage patterns indicated in shaded region.

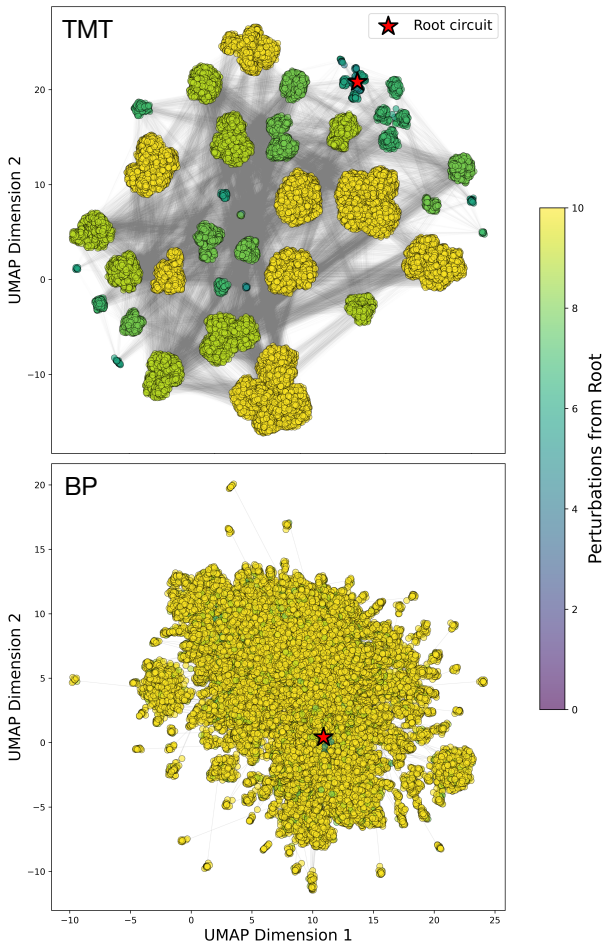


Figure 7: **UMAP of Circuit Solution Space.** 2D embedding of 50,000 functional configurations recovered by the TMT policy (Top) and BP (Bottom) computing binary addition. The red star indicates root circuit, grey edges indicate perturbation - recovery cycles. The TMT policy discovers discrete structural archetypes while BP showcases a single cluster.

Starting from a preconfigured circuit with perfect accuracy on the binary addition task (8 inputs/outputs, yielding a 2560-dimensional LUT vector), we apply a 40-gate recoverable perturbation. Because the damage is not permanent, the exact pre-damage state is theoretically recoverable. However, the system consistently finds alternate, functionally equivalent solutions (perfect accuracy).

By treating each recovered circuit as a new seed for subsequent perturbations, we generate a recursive tree of trajectories (search depth of 10, branching factor of 4). We apply this procedure identically to the TMT policy and a standard BP baseline, visualising the resulting 2560-dimensional configuration space via UMAP (Euclidean distance, $n_{\text{neighbors}} = 15$, min_dist $= 0.1$), where Euclidean distance preserves the neighbourhood ranking of Hamming

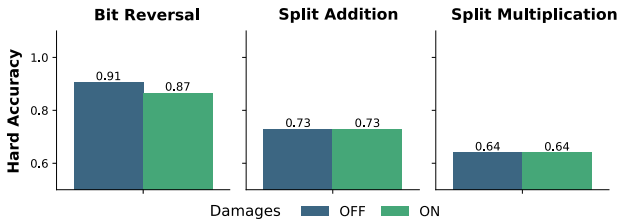


Figure 8: **TMT Performance on Random Topologies.** Test-set accuracy distributions after 256 steps. BP baseline accuracies (inherently wiring-agnostic) are equivalent to those in fig. 2.

distance for binary vectors.

The TMT policy organises recovered solutions into discrete functional clusters. The sparse connectivity between these clusters suggests a *neutral landscape*: multiple successive perturbations force the system to traverse fitness valleys, landing in entirely different structural arrangements that maintain identical global functionality. This reveals an ergodic, degenerate solution space.

In contrast, standard BP produces a single cluster centered on the root configuration, exploring more incremental deformations rather than traversing between distinct structural basins.

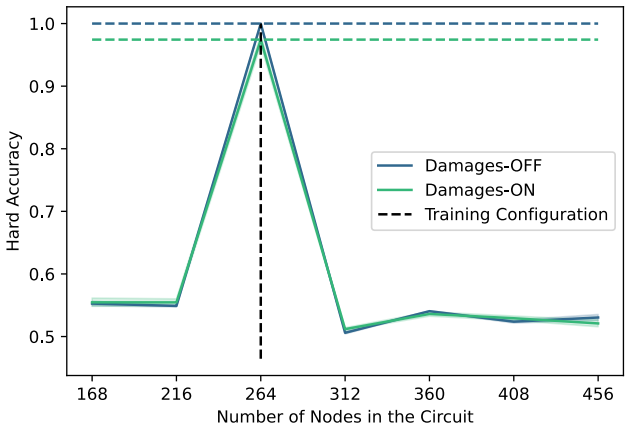
Regime III: Generalisation to Random Topologies

The most challenging setting removes the architectural constraints entirely. We transition to a Random Topology regime, where every circuit in the pool possesses a unique, randomly generated wiring diagram. Consequently, the meta-learner cannot overfit to "Gate A connected to Gate B." Instead, it must learn a truly topological, wiring-agnostic policy.

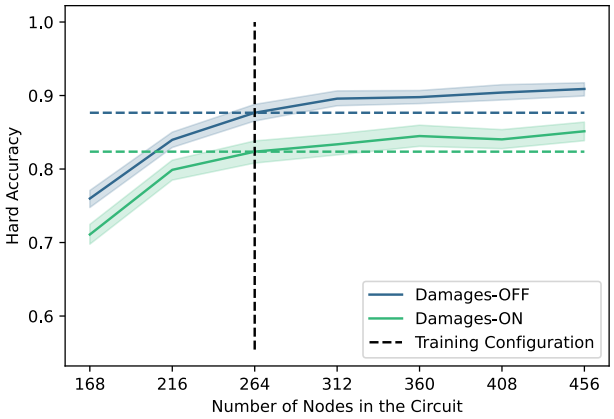
We find this learning regime to be significantly harder than Fixed Topology training. We successfully learned a high-performing, topology-agnostic policy for the **Bit Reversal** task, achieving good generalisation to unseen random graphs. However, for the arithmetic tasks (Addition and Multiplication), the policy struggles to fully converge. While it learns to approximate the output distribution, capturing coarse statistical patterns of the target function, it lacks the precision required for exact arithmetic operations. In this specific regime, standard BP (which is inherently topology-agnostic via re-training) still holds the advantage. However, the success on the Bit Reversal task provides a proof-of-principle that the TMT *can* learn generalisable routing algorithms. We display random-wiring training results in fig. 8.

Regime IV: Scale-Free Optimisation

Finally, we leverage the decentralised nature of our architecture to explore its scaling capabilities. Theoretically, be-



(a) Trained on Fixed Wires



(b) Trained on Random Wires

Figure 9: **Scale-Free Generalisation.** Test accuracy as a function of circuit width (number of nodes). V-line indicates training circuit size ($N = 264$). **(a)** Fixed topology training leads to overfitting: performance peaks at training size and collapses elsewhere. **(b)** Random topology training: performance *improves* as the circuit gets wider, leveraging the extra capacity unseen during training.

cause the TMT shares weights across all nodes and operates on local neighbourhoods, the learned update rule should be independent of the circuit size. We investigate this "Scale-Free" hypothesis by deploying a trained TMT policy on circuits significantly larger (or smaller) than those seen during training.

The results, displayed in fig. 9b, reveal a critical insight: scale-freedom is not just an architectural feature, but rather a *learned capability*.

Overfitting Size (Fixed Training): When the TMT is trained on a fixed topology (fig. 9a), it overfits the specific scale of the training graph. Performance peaks exactly at the training size ($N = 264$, vertical dashed line) and collapses for larger or smaller circuits.

Emergent Scalability (Random Training): In contrast, when trained on the curriculum of random topologies (fig. 9b), the policy generalises remarkably. Not only does it maintain function on larger circuits, but accuracy actually *increases* as we expand the width of the circuit (from 264 to 450+ nodes). The local policy is able to utilise the additional latent capacity of the wider layers to route signals more effectively, despite never having encountered graphs of this size during training.

We note that this scaling success is currently limited to circuit *width*. Scaling *depth* remains a challenge, likely because our positional encoding (normalised depth fraction) changes resolution as layers are added, disrupting the policy's depth perception. Nevertheless, the ability to train on more small, cheap circuits and successfully deploy on wider architectures is a potent validation of the TMT's "growth" paradigm.

Discussion

In summary, we have presented a framework that extends the NCA paradigm from grid-based pattern formation to functional logic generation on arbitrary graphs. By replacing global backpropagation with a decentralised, topology-masked Transformer, we demonstrated that digital circuits can autonomously self-assemble, self-repair, and generalise their routing policies across varying structural scales. For soft errors, the learned policy recovers perfectly even at damage scales several times beyond training, indicating a general repair strategy rather than memorisation of specific failure patterns.

Under such reversible damage, where the system could theoretically return to its exact prior state, the policy drives the circuit toward new, functionally equivalent local minima. This indicates that the meta-learner prioritises functional homeostasis over structural targets.

This use of degenerate solutions mirrors biological multi-scale competency (Pezzulo and Levin, 2016), where evolved systems maintain robust homeostasis under noisy, uncertain conditions by dynamically switching between structurally distinct but functionally equivalent pathways (e.g, aerobic vs. anaerobic metabolic pathways (Edelman and Gally, 2001)). While current industrial FPGA applications typically require deterministic logic configurations, the capacity of the TMT to harness structural degeneracy holds promise for autonomous systems facing unpredictable hardware failures in remote environments such as deep space missions.

Limitations and Future Directions

Despite these capabilities, several limitations present immediate avenues for future work. First, our current positional encoding captures only vertical depth within the circuit DAG, stripping the policy of local topological context. Incorporating Random Walk Structural Encodings (RWSE) (Rampášek et al., 2023), Laplacian eigenvectors (Shuman et al., 2013), or functional metrics like fan-out centrality would provide the meta-learner with richer structural maps while preserving the architecture's scale-freedom. Second, the per-node error feedback r_i is a coarse scalar residual. The meta-learner remains blind to the actual task data; augmenting the architecture with cross-attention to input-output pairs, akin to Perceiver IO (Jaegle et al., 2021), could enable genuine, data-informed in-context reasoning. Finally, while the circuit's physical connectivity is inherently sparse ($< 3\%$ density), the topology mask currently operates as a dense $N \times N$ matrix. Implementing native sparse attention is computationally critical to scale this framework to circuits containing thousands of gates.

Toward Self-Organising Computational Substrates

The true significance of this work lies beyond the Boolean domain on which we have validated it. What distinguishes this system from prior self-organising models is that the decentralised policy operates on a substrate that itself computes. The topology mask defines the physical wiring, while the learned attention policy governs the functional interactions between connected nodes. These two levels, structural connectivity and dynamic coupling, are independently addressable: one can remain fixed while the other adapts, or both can evolve on separate timescales. This dissociation is absent in conventional neural networks, where structure and function are collapsed into a single weight matrix.

The separation naturally suggests a richer architecture in which the same shared-weight mechanism first grows a sparse structural scaffold, then governs the functional dynamics within it, with no hard boundary between the two phases. Realising this structural half, endowing the TMT with the ability to add, prune, and rewire, remains the central open challenge.

By embracing adaptive plasticity over prescriptive redundancy, this work forms the basis for computational substrates that grow, learn, and heal themselves.

Acknowledgements

We thank Nicolas Bessone, Ismail Ceylan, Matthias Dellago, Benedikt Hartl, Kathrin Korte, Milton Montero, Elias Najarro, Joachim W. Pedersen, Fernando Rosas, and Florian Scheidl for fruitful and inspiring discussions.

Funded by the European Union (ERC, GROW-AI, 101045094). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and Freitas, N. d. (2016). Learning to learn by gradient descent by gradient descent. arXiv:1606.04474 [cs].

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer Normalization. arXiv:1607.06450 [stat].

Bachlechner, T., Majumder, B. P., Mao, H. H., Cottrell, G. W., and McAuley, J. (2020). ReZero is All You Need: Fast Convergence at Large Depth. arXiv:2003.04887 [cs].

Béna, G., Faldor, M., Goodman, D. F. M., and Cully, A. (2025). A Path to Universal Neural Cellular Automata. arXiv:2505.13058 [cs] version: 2.

Dehghani, M., Djolonga, J., Mustafa, B., Padlewski, P., Heek, J., Gilmer, J., Steiner, A., Caron, M., Geirhos, R., Alabdulmohsin, I., Jenatton, R., Beyer, L., Tschannen, M., Arnab, A., Wang, X., Riquelme, C., Minderer, M., Puigcerver, J., Evci, U., Kumar, M., Steenkiste, S. v., Elsayed, G. F., Mahendran, A., Yu, F., Oliver, A., Huot, F., Bastings, J., Collier, M. P., Gritsenko, A., Birodkar, V., Vasconcelos, C., Tay, Y., Mensink, T., Kolesnikov, A., Pavetić, F., Tran, D., Kipf, T., Lučić, M., Zhai, X., Keysers, D., Harmsen, J., and Houlsby, N. (2023). Scaling Vision Transformers to 22 Billion Parameters. arXiv:2302.05442 [cs].

Dwivedi, V. P. and Bresson, X. (2021). A Generalization of Transformer Networks to Graphs. arXiv:2012.09699 [cs].

Edelman, G. M. and Gally, J. A. (2001). Degeneracy and complexity in biological systems. *Proceedings of the National Academy of Sciences*, 98(24):13763–13768.

Grattarola, D., Livi, L., and Alippi, C. (2021). Learning Graph Cellular Automata. arXiv:2110.14237 [cs].

Haddow, P. C. and Tyrrell, A. M. (2011). Challenges of evolvable hardware: past, present and the path to a promising future. *Genetic Programming and Evolvable Machines*, 12(3):183–215.

Jaegle, A., Gimeno, F., Brock, A., Zisserman, A., Vinyals, O., and Carreira, J. (2021). Perceiver: General Perception with Iterative Attention. arXiv:2103.03206 [cs].

Kingma, D. P. and Ba, J. (2017). Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs].

Kirsch, L., Harrison, J., Sohl-Dickstein, J., and Metz, L. (2024). General-Purpose In-Context Learning by Meta-Learning Transformers. arXiv:2212.04458 [cs].

Kirsch, L. and Schmidhuber, J. (2022). Meta-Learning Backpropagation and Improving It. arXiv:2012.14905 [cs, stat].

Kresse, F., Yu, E., Lampert, C. H., and Henzinger, T. A. (2025). Logic Gate Neural Networks are Good for Verification. arXiv:2505.19932 [cs] version: 2.

Miotti, P., Niklasson, E., Randazzo, E., and Mordvintsev, A. (2025). Differentiable Logic Cellular Automata: From Game of Life to Pattern Generation. arXiv:2506.04912 [cs].

Mordvintsev, A., Randazzo, E., Niklasson, E., and Levin, M. (2020). Growing Neural Cellular Automata. *Distill*, 5(2):e23.

Neumann, J. V. (1956). Probabilistic Logics and the Synthesis of Reliable Organisms From Unreliable Components. In Shannon, C. E. and McCarthy, J., editors, *Automata Studies. (AM-34)*, pages 43–98. Princeton University Press.

Nudo, R. J. (2013). Recovery after brain injury: mechanisms and principles. *Frontiers in Human Neuroscience*, 7:887.

Petersen, F., Borgelt, C., Kuehne, H., and Deussen, O. (2022). Deep Differentiable Logic Gate Networks. arXiv:2210.08277 [cs].

Pezzulo, G. and Levin, M. (2016). Top-down models in biology: explanation and control of complex living systems above the molecular level. *Journal of The Royal Society Interface*, 13(124):20160555.

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. (2023). Recipe for a General, Powerful, Scalable Graph Transformer. arXiv:2205.12454 [cs].

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P. (2013). The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. *IEEE Signal Processing Magazine*, 30(3):83–98. arXiv:1211.0053 [cs].

Sunada, S., Niiyama, T., Kanno, K., Nogami, R., Röhm, A., Awano, T., and Uchida, A. (2025). Blending Optimal Control and Biologically Plausible Learning for Noise-Robust Physical Neural Networks. *Physical Review Letters*, 134(1):017301.

Thompson, A. (1997). An evolved circuit, intrinsic in silicon, entwined with physics. In Higuchi, T., Iwata, M., and Liu, W., editors, *Evolvable Systems: From Biology to Hardware*, pages 390–405, Berlin, Heidelberg. Springer.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph Attention Networks. arXiv:1710.10903 [stat].

von Neumann, J. (1966). Theory of Self-Reproducing Automata. *University of Illinois Press*.

Wang, W., Li, X., Chen, L., Sun, H., and Zhang, F. (2024). A Review on Soft Error Correcting Techniques of Aerospace-Grade Static RAM-Based Field-Programmable Gate Arrays. *Sensors*, 24(16).

Whitley, D., Yoder, J., and Carpenter, N. (2021). Resurrecting FPGA Intrinsic Analog Evolvable Hardware. MIT Press.

Woods, R. and Lightbody, G. (2008). Robustness in Digital Hardware. *Robust Intelligent Systems, ISBN 978-1-84800-260-9. Springer-Verlag London, 2008, p. 3*.

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., and Liu, T.-Y. (2020). On Layer Normalization in the Transformer Architecture. arXiv:2002.04745 [cs].