# CHIP-STAR: Spatial-Topological Attention for Pre-trained Generative Chip Routing

**Junfeng Liu**[1], **Xingquan Li**[2]*
[1] Department of Optical Fiber Communication, Pengcheng Laboratory
[2] School of Integrated Circuits, Southeast University
{liujf@pcl.ac.cn, fzulxq@gmail.com}

## Abstract

Routing is foundational in VLSI physical design, directly shaping timing, power, and area. Recent pre-trained Seq2Seq models recast routing as autoregressive generation over DFS-serialized trees, but inherit a text backbone blind to the physical space their tokens describe: DFS decouples sequence position from physical coordinate, breaking spatial reasoning at branching points, in load selection, and along the autoregressive trajectory. We name these failures the *Three Spatial Blindnesses* and propose **CHIP-STAR**, which augments a T5Gemma backbone with three lightweight modules: a dual-frame rotary encoding coupling sequence order with 3D proximity, a Fourier-factorized attention prior with memory linear in sequence length, and a cumulative coordinate-noise schedule closing the train–inference gap, together adding under $0.5\%$ parameters. We rebuild the AiEDA corpus into the largest geometry-annotated routing dataset to date ($6.4$M nets, $1.8$B tokens, $556$ GB). CHIP-STAR-Large surpasses iPCL-Large by $\sim 5\%$ on Leaf IoU and Connectivity, and commercial ECO signoff yields $3\%$ shorter wirelength and $11\%$ fewer vias. Our code is publicly available at `https://github.com/jfkey/iPCL-R`.

## 1 Introduction

Routing is a foundational step in VLSI physical design: given pin placements, it generates physical interconnect paths for each net subject to connectivity and Manhattan routing rules, with direct impact on timing, power, and area [5]. As process nodes scale, routing has evolved into a multi-objective optimization problem balancing wirelength, via count, timing slack, and power.

Existing approaches fall into two categories. *Traditional methods* rely on explicit mathematical formulations and hand-crafted rules, optimizing Rectilinear Steiner Minimum Trees per net [7] or solving Integer Linear Programs for concurrent routing and congestion [9, 10, 46]. *AI-augmented methods* learn routing heuristics from data, including reinforcement-learning global routers [19], neural Steiner-tree constructors [11, 22, 29], and deep generative routers [6, 38].

A new paradigm has recently emerged from pre-trained Seq2Seq models such as iPCL [17] and iPCL-M [13]: the routing tree is serialized via depth-first traversal (DFS) into a sequence of incremental direction tokens, e.g., `R10` (move right 10 units) or `T2` (ascend two metal layers), and an encoder–decoder model autoregressively generates complete paths conditioned on the net description [51]. The split is well-suited to routing: the encoder ingests the full net context (driver, loads, and their spatial relationships) once, while the decoder emits routing decisions step by step, matching the branch-structured nature of routing trees. Crucially, this reformulation lets large pre-trained models capture transferable routing knowledge across heterogeneous designs, while delivering a complete solution in a single forward pass at negligible latency (Figure 1, top).

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

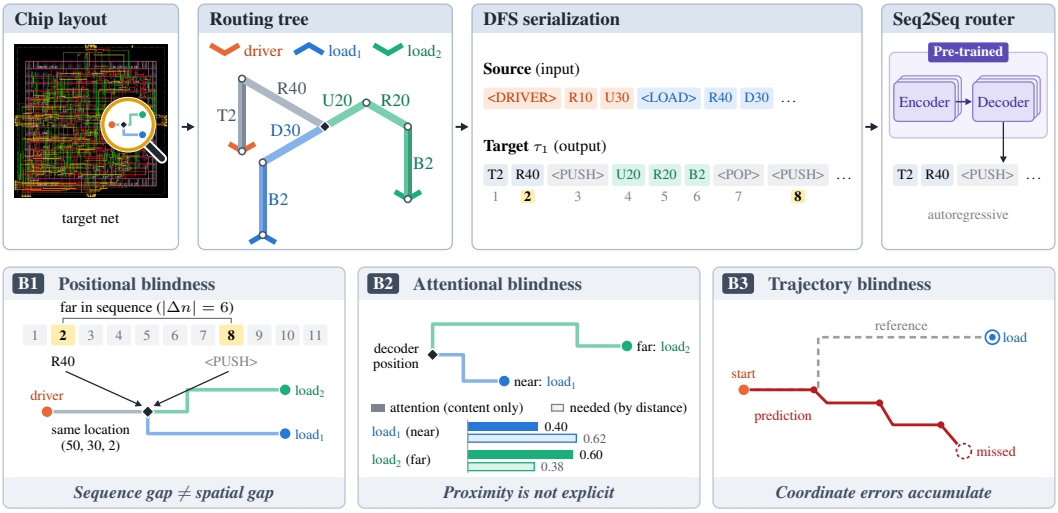



Figure 1: **Top: pre-trained Seq2Seq routing.** The routing tree of a target net is DFS-serialized into direction tokens and generated autoregressively by a pre-trained encoder–decoder. **Bottom: Three Spatial Blindnesses.** B1: RoPE confuses sequence gap for physical distance (positions 2 and 8 share one location). B2: Cross-attention lacks a spatial proximity channel, so a far load can draw more attention than a near one. B3: Coordinate errors compound along the inference trajectory.

Yet encoder–decoder backbones such as the T5Gemma used in iPCL are general-purpose text models, and operate on routing tokens without any intrinsic awareness of the physical space those tokens describe. The root cause is structural: DFS serialization decouples a token's *sequence position* from its *physical coordinate*, so spatially adjacent nodes may sit hundreds of steps apart in the sequence and vice versa. This single decoupling propagates through three layers of the model, manifesting as failures we collectively term the **Three Spatial Blindnesses** (Figure 1, bottom): the input encoding misreads geometry (B1), the attention mechanism cannot query it (B2), and once it is finally available, the training procedure cannot deliver it reliably (B3).

**(B1) Positional Blindness** arises at the *input encoding*. RoPE [39] treats sequence-index difference $|i - j|$ as a proxy for geometric distance. DFS serialization breaks this proxy: a single tree admits many valid traversals, so identical spatial relations receive varying sequence distances, and $\langle$POP$\rangle$ jumps separate adjacent nodes by hundreds of positions. The model thus cannot perceive proximity at branching points.

**(B2) Attentional Blindness** arises at the *attention mechanism*. Routing is fundamentally a reachability problem: at each decoding step the decoder needs to know how far it is from each unconnected load. Cross-attention scores tokens by Q/K content similarity alone, with no channel for spatial distance, so this geometric signal must be inferred from content rather than read directly.

**(B3) Trajectory Blindness** arises at the *training procedure*, once coordinates are introduced into the decoder to fix B1 and B2. Under teacher forcing, decoder coordinates accumulate from ground-truth tokens; at inference, from the model's own predictions, so per-step errors compound into drift. This is the spatial counterpart of exposure bias [2]: the stronger the geometric mechanisms, the deeper the dependence on coordinate precision, and the more damaging the mismatch.

To address the three blindnesses, CHIP-STAR augments a pre-trained T5Gemma backbone with three lightweight, targeted modules:

- **STPE** anchors attention in physical space by composing a quaternion-based 3D rotation onto RoPE, with the spatial component invariant to DFS traversal (§3.2).
- **GeoPrior** tracks distance to unconnected loads by adding a content-independent additive term to cross-attention, computed in memory linear in sequence length, via a Fourier factorization (§3.3).
- **TrajNoise** closes the coordinate drift gap by perturbing decoder coordinates with cumulative Gaussian noise that matches the inference-time error process (§3.4).

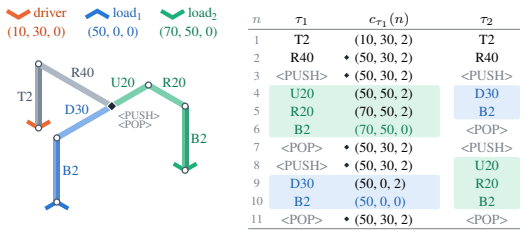
We evaluate CHIP-STAR on the largest geometry-annotated routing corpus to date, rebuilt from AiEDA [36] and iPCL [17]. The corpus contains 6.4M nets and 1.8B tokens across 32 designs (556 GB). Our contributions are three-fold. First, we identify the *Three Spatial Blindnesses*, a conceptual framework that describes how DFS serialization limits geometric reasoning in Seq2Seq models. The framework also applies to other spatially-embedded tree generation tasks. Second, we introduce three modules, each targeting one Blindness: STPE anchors attention in physical space, GeoPrior tracks distance to unconnected loads, and TrajNoise reduces the coordinate drift between training and inference. Third, with joint topological and spatial awareness, CHIP-STAR-Large improves Leaf IoU and Connectivity by about 5% over iPCL-Large. Commercial ECO signoff further shows shorter wirelength (3%) and fewer vias (11%).

## 2 Background

**Chip Routing and Its Physical Primitives** Chip routing connects each net's *driver* pin to its *load* pins on a multi-layer metal stack under Manhattan rules. A routed `.def` file is built from four primitives: **wires** carry signals within one layer and dominate per-net RC; **vias** stitch wires across adjacent layers and make via count a standard quality proxy; **pins** are the (possibly multi-layer) rectangles through which a cell exposes its terminals; and **patches** are auxiliary rectangles inserted during ECO for timing or signal-integrity fixes. Abstracting these primitives into adjacency relations and filling pin/patch shapes with virtual Manhattan segments, a routed net becomes a rectilinear Steiner tree $T = (V, E, p, v_0)$ rooted at the driver $v_0$, with loads as leaves (App. A.1, A.3).

**Routing as Sequence Generation.** Seq2Seq routers [13, 17] cast $T$ as a generation target via depth-first serialization into direction-displacement tokens. The encoder reads the driver coordinate and each load's offset relative to the driver (e.g. `R10 U30` for $(10, 30, 0)$); the decoder emits a DFS traversal $\tau \in \mathcal{S}(T)$, with `PUSH`/`POP` marking branch entry/exit and direction tokens encoding edge moves (as shown in Fig 2). To cover the four-decade coordinate range, iPCL [17] adopts *DecimalWordLevel* tokenization, decomposing each displacement by decimal place value while keeping its direction prefix (207 tokens; App. A.2).

**The Sequence–Coordinate Decoupling.** A tree $T$ admits combinatorially many valid serializations $\mathcal{S}(T)$, since child visit order at every branching node is free. The coordinate function $c_\tau$ (Def. A.3) tracks the DFS stack top for any $\tau$, but the sequence index $n$ does not: the same node sits at different positions across traversals, a POP return can place spatially adjacent nodes hundreds of steps apart, and consecutive tokens may be physically distant, e.g., `T2` at $(10, 30, 2)$ and `R40` at $(50, 30, 2)$ in Fig. 2. This decoupling between sequence index and physical coordinate $p$ is the single root of the Three Spatial Blindnesses (B1–B3), addressed in §3.




| $n$ | $\tau_1$ | $c_{\tau_1}(n)$ | $\tau_2$ |
|---|---|---|---|
| 1 | T2 | (10, 30, 2) | T2 |
| 2 | R40 | ♦ (50, 30, 2) | R40 |
| 3 | <PUSH> | ♦ (50, 30, 2) | <PUSH> |
| 4 | U20 | (50, 50, 2) | D30 |
| 5 | R20 | (70, 50, 2) | B2 |
| 6 | B2 | (70, 50, 0) | <POP> |
| 7 | <POP> | ♦ (50, 30, 2) | <PUSH> |
| 8 | <PUSH> | ♦ (50, 30, 2) | U20 |
| 9 | D30 | (50, 0, 2) | R20 |
| 10 | B2 | (50, 0, 0) | B2 |
| 11 | <POP> | ♦ (50, 30, 2) | <POP> |

Source Tokens: <DRIVER>, R10, U30, <LOAD>, R40, D30, <LOAD>, R60, U20

Figure 2: One routing tree $T$ and two of its DFS serializations $\tau_1, \tau_2 \in \mathcal{S}(T)$. The two branches (to load$_1$/load$_2$) swap order, while <PUSH>/<POP> stay at the branch point (♦); $c_{\tau_1}(n)$ gives the coordinate at position $n$.

**Position Encodings for Spatially-Embedded Inputs.** Position encodings inject order or geometry into permutation-invariant attention. Absolute schemes (sinusoidal [45] or learned) tag each index with a fixed vector; RoPE [39] encodes index differences through orthogonal rotations so attention depends only on $|i - j|$. For high-dimensional inputs, LiePE [32] learns continuous Lie-group elements, STRING [37] guarantees translation invariance in arbitrary dimensions, and GeoPE [49] restores 3D Euclidean geometry via quaternions. All assume a regular input space (1D sequence, grid, or point cloud) where positional and geometric arguments coincide. In DFS-serialized routing they do not: the same geometry admits $|\mathcal{S}(T)|$ valid orderings, and any index-bound encoding conflates serialization noise with physical signal, motivating the dual-frame design of STPE (§3.2). Extended related work is in App. B.

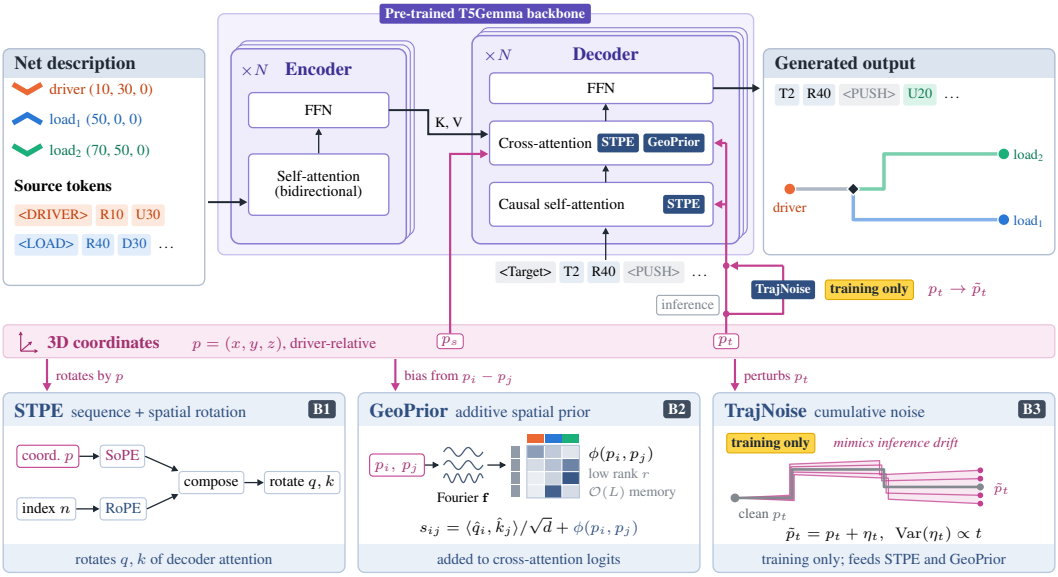



Figure 3: Overview of CHIP-STAR

# 3 Method: CHIP-STAR

## 3.1 Overall Pipeline

We formulate chip routing as a geometry-conditioned generative sequence task. A routed Steiner tree $T$ is serialized via DFS into a token sequence with PUSH/POP anchors marking branch entry and exit. An encoder–decoder backbone (T5Gemma [51]) reads the driver/load layout from the encoder side and autoregressively emits direction tokens from the decoder side. Each token $i$ carries two pieces of information: a sequence index $n_i$ used by RoPE, and a physical coordinate $p_i \in \mathbb{R}^3$ recovered from the cumulative DFS-stack-top function $c_\tau$ (§2). The three modules of CHIP-STAR (STPE, GeoPrior, TrajNoise) each consume $p_i$ to inject geometry into a specific layer of the model. The framework is illustrated in Fig. 3. A complete list of symbols used throughout this section is provided in App. C.

**Coordinate Normalization.** Raw coordinates are anisotropic: $(x, y)$ ranges over the chip span ($\sim 10^6$ nm at 28 nm) while $z$ indexes a small number of metal layers ($\sim 10^1$). Without rescaling, the five-order axis imbalance dominates any geometric encoding. We apply a fixed per-axis rescaling

$$p = \big(\lambda_{xy}\, x_{\text{raw}},\ \lambda_{xy}\, y_{\text{raw}},\ \lambda_z\, z_{\text{raw}}\big), \tag{1}$$

so that all three axes share a comparable range. The rescaling eliminates the cross-axis imbalance but preserves the four-order dynamic range within each axis grid pitch at $\sim 10^{-3}$, chip span at $\sim 10^1$, which is the multi-scale signal SoPE and GeoPrior need to resolve.

## 3.2 STPE: Anchoring Attention in Physical Space

To address Positional Blindness (B1), STPE composes a coordinate-driven 3D rotation, denoted SoPE, on top of standard RoPE, so that attention scores depend on both sequence order and physical proximity:

$$\hat{q}_i = \mathrm{SoPE}(p_i)\,\mathrm{RoPE}(q, n_i), \quad \hat{k}_j = \mathrm{SoPE}(p_j)\,\mathrm{RoPE}(k, n_j), \quad s_{ij} = \langle \hat{q}_i, \hat{k}_j \rangle / \sqrt{d}. \tag{2}$$

We motivate SoPE from three routing-side requirements (§3.2.1), realize them via a multi-scale quaternion rotation (§3.2.2), and analyze how the resulting score relates to B1 (§3.2.3).

### 3.2.1 Routing-Side Requirements

**(R1) Cross-axis coupling.** A diagonal displacement such as $(100, 10, 2)$ is not the sum of three independent moves: planar detour and via-up share the same offset budget and trade off against

each other. Axis-wise extensions of RoPE place $x$, $y$, $z$ in disjoint head sub-spaces, so joint $(x{:}y{:}z)$ directions are not directly representable [32]. SoPE should act on the full 3D sub-vector at once.

**(R2) No non-physical axis ordering.** The three axes are not equivalent: $z$ is discrete and comes with via cost and layer-parity rules. Still, no *a priori* ordering among them is justified. Composing 3D rotation as $R_z R_y R_x$ injects a non-commutativity bias that has no routing meaning. SoPE should therefore treat $x, y, z$ symmetrically, leaving relative axis weights to the data through $\lambda_z$ (Eq. 1) and the TrajNoise scales.

**(R3) Driver-relative anchoring.** Routing geometry is driver-relative: the driver pin serves as the origin of the net's frame, and load offsets are what the model reasons about. SoPE should reduce to identity when $p = 0$, so the driver imposes no spurious rotation.

These three requirements pin down SoPE up to a multi-scale frequency schedule (§3.2.2): quaternion rotation of the head sub-vector for (R1); $\mathfrak{so}(3)$ symmetric averaging, which takes the three per-axis rotations to a single rotation whose magnitude is their Euclidean norm, for (R2); and continuous degeneration to identity at $p = 0$ for (R3).

### 3.2.2 Multi-scale Quaternion Rotation

After normalization, all three axes occupy the same range of roughly $10^{-3}$ to $10^{1}$, covering metal pitch up to chip span. A single rotation frequency cannot resolve this four-order span: too high causes the chip-span phase to wrap, too low collapses adjacent grid cells onto the same phase. SoPE therefore uses a multi-scale frequency schedule.

Each head is partitioned into $K = \lfloor d_{\text{head}}/3 \rfloor$ 3D sub-vectors, with the $\ell$-th band assigned frequency $f_\ell = \theta_{\text{base}}^{-\ell/K}$ ($\theta_{\text{base}} = 1000$). The symmetric phase magnitude in band $\ell$ is

$$\Theta_\ell = \tfrac{1}{3}\, f_\ell \sqrt{x^2 + y^2 + z^2}, \tag{3}$$

which is the $\mathfrak{so}(3)$-averaged rotation magnitude required by (R2). The exponential map then gives the per-band unit quaternion

$$R_\ell = \Big( \cos(\Theta_\ell/2),\ \tfrac{\sin(\Theta_\ell/2)}{3\Theta_\ell}\, f_\ell\, (x, y, z) \Big), \tag{4}$$

applied to the head sub-vector via the standard quaternion sandwich product [49]. By construction $R_\ell$ acts on all three dimensions simultaneously (R1), is invariant to axis ordering (R2), and approaches identity as $p \to 0$ (R3). High-$\ell$ bands resolve grid-level relations; low-$\ell$ bands capture chip-span topology.

### 3.2.3 Geometric Decomposition

By Rodrigues' formula [37, 49], the rotated inner product $\langle \hat{q}_i, \hat{k}_j \rangle$ depends only on the relative rotation between $p_i$ and $p_j$, which yields translation invariance. Let $A_\ell$ and $\mathbf{a}_\ell$ denote the rotation angle and axis induced by $p_i - p_j$ in band $\ell$. The score then decomposes as

$$s_{ij} \ \propto\ \underbrace{\langle q, k \rangle \cos A_\ell}_{\text{(I) Projected Similarity}} \ +\ \underbrace{(q{\cdot}\mathbf{a}_\ell)(k{\cdot}\mathbf{a}_\ell)(1 - \cos A_\ell)}_{\text{(II) Axial Alignment}} \ +\ \underbrace{(\mathbf{a}_\ell \times q){\cdot}k\, \sin A_\ell}_{\text{(III) Torsional}}, \tag{5}$$

where we omit the RoPE sub-rotation for clarity [39]. Each term admits a routing-side reading:

- **Term (I)** reweights content similarity by physical distance through $\cos A_\ell$. Spatially close tokens retain their similarity, while distant tokens are attenuated. This term is most relevant to B1: two tree nodes that are far apart in DFS order (e.g., across a `POP`) but close in space can still attend to each other.
- **Term (II)** biases attention along the displacement direction $\mathbf{a}_\ell$, which can encourage the decoder to extend toward unconnected loads. The behavior is consistent with an A*-style heuristic, although no explicit goal token is provided [9].
- **Term (III)** captures the component perpendicular to $\mathbf{a}_\ell$, separating paths that share planar coordinates but differ in metal layer. It provides a geometric signal relevant to the planar-detour vs. via-up tradeoff.

The three terms follow from an algebraic identity rather than hand-crafted features; their routing-side reading follows from chip routing being a 3D Euclidean task.

### 3.2.4 Theoretical Guarantee

The STPE attention score admits a natural separation into a sequence-driven RoPE component and a geometry-driven SoPE component, and only the latter is invariant to the choice of DFS traversal. This separation is exactly what we want: the geometric part fixes B1 by giving the model a serialization-independent view of physical layout, while the residual sequence part is retained as a tie-breaker for ordering tokens during autoregressive decoding.

**Theorem 3.1** (Geometric Invariance of the SoPE Component)**.** *Let $T$ be a routing Steiner tree with coordinate function $c_\tau$ (Def. A.3), and let $\tau_1, \tau_2 \in \mathcal{S}(T)$. For any tree nodes $u, v \in T$ and any $q, k \in \mathbb{R}^d$, the SoPE-only score*

$$s_{\mathrm{SoPE}}(q, k;\, p_u, p_v) \;\triangleq\; \langle \mathrm{SoPE}(p_u)\, q,\; \mathrm{SoPE}(p_v)\, k \rangle / \sqrt{d} \tag{6}$$

*depends only on $(p_u, p_v)$ and is therefore identical under $\tau_1$ and $\tau_2$. In contrast, the RoPE-only score depends on the sequence indices and is in general not invariant: there exist $T, \tau_1, \tau_2$ for which $s_{\mathrm{RoPE}}$ takes different values.*

**Implication.** The theorem covers only the SoPE component of STPE: the SoPE score is invariant to the DFS traversal, whereas the full STPE score, which also contains the RoPE component, is not. It explains why the geometric signal stays stable across serializations; it does not make the full attention score, or the model, serialization-invariant. The residual sequence dependence is needed for autoregressive decoding. Fig. 9 shows this empirically: ROUGE-L $\approx 0.92$ on geometrically equivalent outputs, i.e., the tree is preserved even when the specific serialization is not.

### 3.3 GeoPrior: Tracking Distance to Unconnected Loads

Routing is fundamentally a reachability problem: the decoder's job is to extend the partial wire from the driver until every load is reached, and at each step the most useful information is how far the current decoder position $p_t$ still is from each unconnected load. Encoder tokens carry the load coordinates, decoder tokens carry the running position, so this "distance to remaining loads" signal can in principle be read off from cross-attention. STPE makes attention sensitive to spatial distance, but the signal is multiplicative on top of content similarity $\langle \hat{q}_i, \hat{k}_j \rangle$, which means a load whose token embedding happens to be content-dissimilar to the current decoder state may be missed even when it is the closest one. We therefore add a content-independent additive term $\phi(p_i, p_j)$ to cross-attention, so that proximity to remaining loads is visible to the decoder regardless of token content. The roles are complementary: STPE modulates content similarity by geometry, GeoPrior provides geometry on its own channel.

**Why factorize.** A direct implementation $\phi(p_i, p_j) = \mathrm{MLP}(p_i - p_j)$ enumerates every $(i, j)$ pair, with cost $\mathcal{O}(L^2 d)$ and a pairwise tensor of tens of GB at $L = 512$.

**Fourier factorization.** We use the Fourier identity

$$\sin(\omega x_i)\cos(\omega x_j) - \cos(\omega x_i)\sin(\omega x_j) = \sin\big(\omega(x_i - x_j)\big), \tag{7}$$

which recovers a relative-position term from a product of two per-token encodings, reducing the pairwise cost from $\mathcal{O}(L^2)$ to $\mathcal{O}(L)$ encoding plus one matmul:

$$\phi(p_i, p_j) \;=\; \big(\mathbf{W}_q\, \mathbf{f}_i\big)\big(\mathbf{W}_k\, \mathbf{f}_j\big)^\top, \qquad \mathbf{f}_i \;=\; \big[\sin(\mathbf{W}_\omega p_i),\; \cos(\mathbf{W}_\omega p_i)\big] \in \mathbb{R}^{2F}, \tag{8}$$

where $\mathbf{W}_\omega \in \mathbb{R}^{F \times 3}$ is a learnable frequency projection that mixes axes and selects multi-scale resolutions, and $\mathbf{W}_q, \mathbf{W}_k$ project into rank-$r$ per-head spaces. The cross-attention score becomes

$$s_{ij} \;=\; \langle \hat{q}_i, \hat{k}_j \rangle / \sqrt{d} \;+\; \phi(p_i, p_j), \tag{9}$$

with $\hat{q}, \hat{k}$ from STPE. The factorization carries the relative displacement $p_i - p_j$ inside its frequency components and adds $\mathcal{O}(Fr)$ parameters per head, well under $0.1\%$ of the backbone.

### 3.4 TrajNoise: Closing the Coordinate Drift Gap

The decoder coordinate is accumulated step by step: $p_t = p_{t-1} + \Delta_t$, where $\Delta_t$ is the displacement of the $t$-th token. STPE and GeoPrior both consume $p_t$, so any error in an earlier prediction propagates

Table 1: Main results on the validation set. CHIP-STAR outperforms baselines at both MEDIUM and LARGE scales across all topology-aware metrics. **Bold** marks the best value per column.

| | | Topology-aware | | | | Text | | |
|---|---|---|---|---|---|---|---|---|
| **Model** | **#Para (M)** | **Leaf IoU** | **Leaf Match** | **Conn.** | **Graceful** | **ROUGE-L** | **Branch** | **EM** |
| Llama | 407 | 0.870 | 0.833 | 0.834 | 0.834 | 0.575 | 0.868 | 0.134 |
| iPCL-Medium | 52 | 0.916 | 0.884 | 0.885 | 0.885 | 0.579 | 0.873 | 0.135 |
| iPCL-Large | 416 | 0.949 | 0.924 | 0.925 | 0.924 | 0.581 | 0.876 | **0.139** |
| CHIP-STAR-Medium | 53 | 0.948 | 0.922 | 0.923 | 0.922 | 0.579 | 0.875 | 0.137 |
| CHIP-STAR-Large | 418 | **0.984** | **0.971** | **0.972** | **0.971** | 0.580 | 0.874 | 0.137 |

into all subsequent geometric channels. Under teacher forcing $\Delta_t$ is taken from ground-truth tokens, but at inference each $\Delta_t$ is the model's own prediction, and the accumulation in $p_t$ amplifies per-step errors into a drift that grows along the trajectory. This is the standard exposure-bias gap, now expressed in coordinate space rather than token space [2].

To match the training distribution to this inference-time error, we perturb $p_t$ with a noise term $\boldsymbol{\eta}_t$ that is itself accumulated:

$$\tilde{p}_t \;=\; p_t + \boldsymbol{\eta}_t, \qquad \boldsymbol{\eta}_t \;=\; \frac{\sigma}{\sqrt{L}} \sum_{u=1}^{t} \boldsymbol{\epsilon}_u, \qquad \boldsymbol{\epsilon}_u \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{10}$$

where $L$ is the decoder length and $\sigma = (\sigma_{xy}, \sigma_{xy}, \sigma_z)$ broadcasts per-axis in the normalized coordinate space (Eq. 1). The cumulative form makes $\mathrm{Var}(\boldsymbol{\eta}_t) = \sigma^2 t/L$, so noise grows with $t$ in the same way that prediction error compounds at inference. The noised $\tilde{p}_t$ then replaces $p_t$ in STPE and GeoPrior.

A per-sample Bernoulli mask gates whether noise is applied, with its activation rate ramped from 0 to a target value during warmup. The mask is applied per-sample rather than per-step, because at inference the trajectory is either fully clean or fully driven by predictions, never alternating between the two within one decode.

# 4 Experiments

## 4.1 Experimental Setup

**Environment.** All experiments use a 28 nm node, with synthesis by Synopsys Design Compiler [40] and back-end evaluation (dataset construction, ECO, and quality reporting) by Cadence Innovus [3]. Training runs on 16 NVIDIA V100 GPUs (32 GB each). **Dataset.** We build on AiEDA [36] and iPCL to construct a corpus of 32 training designs comprising $\sim$6.4M nets (1.8B tokens) parsed from DEF files, with per-net spatial coordinates annotated (556 GB total). **Baselines.** We compare against an encoder–decoder (T5Gemma backbone) iPCL [17] and a decoder-only (Llama) [44] model. CHIP-STAR builds on T5Gemma with hidden size 1024, 12 layers, 16 heads, and 400M parameters. **Hyperparameters.** GeoPrior uses $F$=16 Fourier frequencies and rank $r$=8; coordinate scales are $\lambda_{xy}$=3$\times$10$^{-5}$, $\lambda_z$=1.0; TrajNoise sets $\sigma_{xy}$=5.0, $\sigma_z$=2.0. Details are in Appendix E.

## 4.2 Main Results

Table 1 reports two metric families: *topology-aware* metrics (Leaf IoU, Leaf Match, Connectivity, Graceful), which evaluate the routing tree as a geometric object, and *text* metrics (ROUGE-L, Branch, EM), which measure surface-level sequence agreement. **(i)** On topology-aware metrics, CHIP-STAR-Large lifts Leaf IoU from 0.949 to 0.984 (+3.5%) and Leaf Match / Connectivity by $\sim 5\%$ over iPCL-Large; notably, CHIP-STAR-Medium (53M) already matches iPCL-Large (416M), showing that the gains stem from geometric inductive bias rather than scale. **(ii)** Text metrics differ by $< 0.01$ across all models—an artifact of the many DFS serializations describing one physical layout, which validates evaluating routing through topology-aware metrics. **(iii)** Llama trails both encoder–decoder models on every topology metric, confirming that bidirectional encoding of the source pin set is critical for global net geometry. For supplementary experimental results, please refer to Appendix F.

Table 2: Ablation on CHIP-STAR-Medium (53M). Each row removes one component; Δ reports the relative drop on Leaf IoU versus the full model. iPCL-Medium serves as the no-geometry reference.

| Variant | Leaf IoU | Leaf Match | Conn. | Graceful | ΔIoU |
|---|---|---|---|---|---|
| CHIP-STAR-Medium (full) | **0.948** | **0.922** | **0.923** | **0.922** | — |
| w/o STPE | 0.922 | 0.892 | 0.894 | 0.893 | −2.52% |
| w/o GeoPrior | 0.933 | 0.903 | 0.905 | 0.904 | −1.41% |
| w/o TrajNoise | 0.938 | 0.910 | 0.912 | 0.911 | −0.91% |
| iPCL-Medium *(no geometry)* | 0.916 | 0.884 | 0.885 | 0.885 | −3.12% |

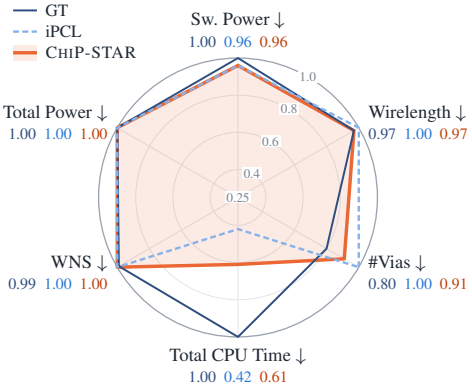



Figure 5: PPA metrics across 10 held-out designs (after ECO ).

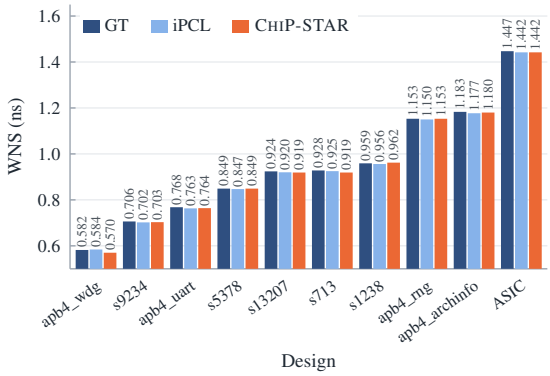



Figure 6: Per-design WNS; positive values indicate timing closure.

**Where do the gains come from?** Figure 4 stratifies Leaf IoU by net size: 1- and 2-load nets are saturated for both models, but on $3^+$-load nets CHIP-STAR improves Leaf IoU from 0.736 to 0.850 (+11.4 absolute points). The gains thus concentrate exactly where DFS serialization induces long-range cross-branch dependencies, providing direct evidence that STPE and GeoPrior recover the spatial signal pure sequence models lose.

**Ablation study.** Table 2 ablates each component on CHIP-STAR-Medium, using iPCL-Medium as the no-geometry reference. Removing STPE drops Leaf IoU by 2.52%. Removing GeoPrior drops it by 1.41%, and removing TrajNoise by 0.91%. This ordering matches their roles in the model. STPE restores spatial locality across DFS branches. GeoPrior provides a content-independent additive signal. TrajNoise narrows the train-inference gap, rather than adding new information. The three drops sum to about the full gap to iPCL-Medium (3.12%), which suggests that the components contribute largely complementary signals. We limit ablations to the Medium scale due to compute budget. A 3-seed study on CHIP-STAR-Medium (5 epochs per run; Appendix F.1) gives a seed standard deviation of at most 0.004 Leaf IoU, while the gap between the full model and iPCL-Medium ($0.895 \pm 0.004$ vs. $0.869 \pm 0.002$) is more than six times larger.

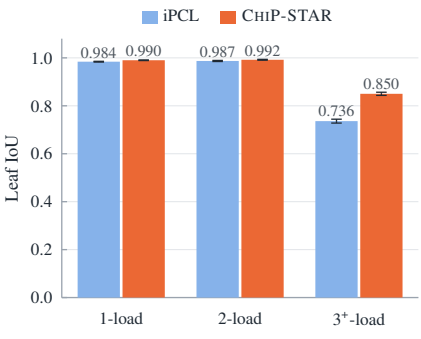



Figure 4: Leaf IoU by net size: gains concentrate on multi-load nets.

### 4.3 Layout-Level Downstream Evaluation

We pass CHIP-STAR's predicted routes through a standard back-end flow (DRC clean and ECO in Innovus) on 10 held-out designs, and report physical PPA metrics both before and after ECO.

**PPA (Fig. 5, Table 3).** After ECO, CHIP-STAR uses 8.7% fewer vias and 1.9% less wirelength than iPCL (totals over the 10 designs). These gains are already present in the raw model output: before

Table 3: Physical metrics and CPU time on the 10 held-out designs. Δ: CHIP-STAR vs. iPCL, on totals (Σ) and averaged per design; Wins: designs where CHIP-STAR is lower.

| Metric | Stage | iPCL | CHIP-STAR | Δ (Σ) | Δ (per design) | Wins |
|---|---|---|---|---|---|---|
| DRC violations | pre-ECO | 2972 | 1792 | −39.7% | −14.1% | 8/10 |
| | post-ECO | 0 | 0 | – | – | – |
| WL (µm) | pre-ECO | 7784 | 7409 | −4.8% | −4.6% | 10/10 |
| | post-ECO | 8077 | 7926 | −1.9% | −1.3% | 9/10 |
| #Vias | pre-ECO | 10486 | 9321 | −11.1% | −9.6% | 10/10 |
| | post-ECO | 10053 | 9177 | −8.7% | −6.0% | 10/10 |
| CPU time (× router) | inference | 0.11 | 0.24 | +124.7% | +124.3% | 0/10 |
| | ECO | 0.32 | 0.34 | +5.1% | +3.2% | 4/10 |
| | total | 0.43 | 0.58 | +34.9% | +24.3% | 1/10 |

ECO, CHIP-STAR uses 11.1% fewer vias and 4.8% less wirelength, on every design, and leaves 39.7% fewer DRC violations in total (fewer on 8 of 10 designs). ECO narrows the gap rather than creating it. Switching power drops by 4% relative to the commercial-tool ground truth. Total power and WNS stay close to the ground truth, so the wirelength and via savings do not come at the cost of timing. CHIP-STAR finishes the back-end flow in 58% of the commercial router's CPU time, versus 43% for iPCL. The extra time is spent in model inference (0.24× vs. 0.11× the router's CPU time), while ECO time is comparable (0.34× vs. 0.32×), so the overhead does not come from a heavier repair burden. The overhead is acceptable given the gains in topology quality.

**Timing (Fig. 6).** Timing depends on interactions across nets, which neither learned router models explicitly, so we measure it after the full signoff flow. On small and mid-sized designs, CHIP-STAR and iPCL stay close to the ground truth, and in some cases give slightly better WNS. On the largest design (ASIC), all three settings show similar degradation, including the commercial router. This suggests that timing on this design is limited by the back-end flow, not by the routing model, and that the wirelength and via savings of CHIP-STAR do not come at the cost of timing.

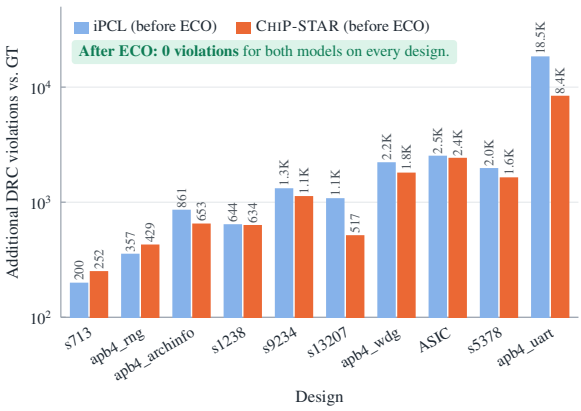
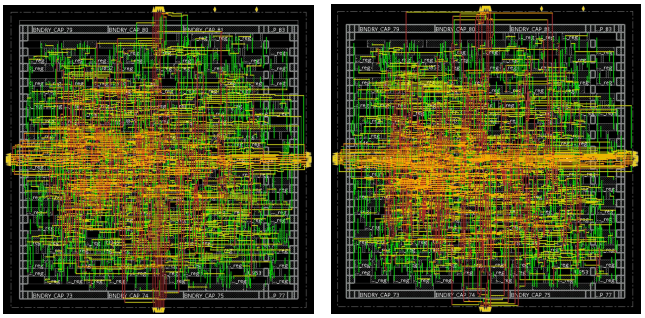



Figure 7: Distribution of DRC violation counts across held-out designs before and after ECO post-processing. After ECO, all designs reach zero DRC violations.

**DRC (Fig. 7).** Both learned routers leave DRC violations before ECO. They decode each net independently and cannot see the wires of neighbouring nets, so spacing and routing conflicts between nets accumulate as the design grows. CHIP-STAR leaves fewer violations than iPCL on 8 of 10 designs, with the largest reduction on the largest design, `apb4_uart` (8.4K vs. 18.5K), consistent with STPE and GeoPrior supplying the spatial context that single-net decoding otherwise lacks. The two exceptions, `s713` and `apb4_rng`, are the smallest designs, where both models leave few violations and the difference is small. After ECO, both models reach zero violations on every design, so all post-ECO metrics are measured on DRC-clean layouts.

**Case studies.** On `s5378` (Fig. 8), CHIP-STAR reproduces the ground-truth pattern almost identically on low metal layers (M3 and M4). Deviations on higher layers come from weak cross-net coupling, which a per-net generative formulation does not directly model. The 3D case in Fig. 9 is more illustrative. ROUGE-L is only 0.92, so the predicted sequence does not match the ground truth exactly. However, the predicted tree is geometrically equivalent to the ground truth. This is consistent with Theorem 3.1: STPE makes attention invariant to DFS traversal, so the model tends to learn the underlying tree rather than a specific serialization.



(a) Ground truth (b) CHIP-STAR

Figure 8: Routing layout for `s5378`, metal layers M3–M6. DRC is cleaned by ECO [3].

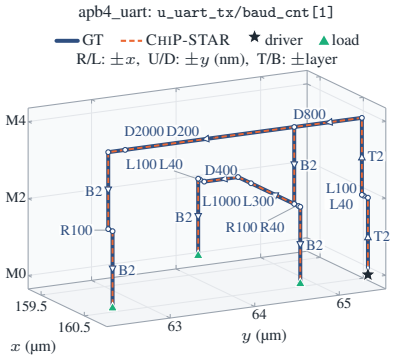



Figure 9: CHIP-STAR's 3D routing tree vs. the ground truth.

## 5 Discussion and Limitations

**Choice of paradigm.** We adopt the pre-trained Seq2Seq paradigm, which generates Steiner points autoregressively and leverages large backbones [13, 17]. We position CHIP-STAR as a topology-faithful initializer for a detailed router, not as a DRC-clean router: it generates the routing tree, and ECO then enforces the design rules. GNNs excel at scoring given node sets, often with explicit search [12, 24]. Diffusion and flow matching [21] have advanced placement [15, 47] and global routing [6, 38], but their routing outputs are fixed-resolution GCell maps rather than variable-length paths with discrete branches. Future directions include combining autoregressive topology decoding with flow-matching coordinate refinement, cross-design generalization [41], and multimodal fusion [14, 42].

**Limitations.** *Backbone scale.* Limited by available pre-training data, we fix the backbone at 0.4B; behavior at larger scale remains open. *ECO dependence.* DRC violations are measured after ECO; CHIP-STAR does not yet produce DRC-clean layouts directly. *Single-net scope.* Nets are routed independently, so multi-net congestion is not modeled. *Coarse vias.* Vias are encoded only by direction (B/T), omitting type and stack attributes. *Generalization.* All designs use one 28 nm node; transfer to other process nodes is untested. *Classical baselines.* We compare only against the commercial router (Innovus), not against open-source classical flows such as OpenROAD [43] and TritonRoute [10].

## 6 Conclusion

We trace three structural failures of pre-trained Seq2Seq routers, the *Three Spatial Blindnesses*, to a single root cause: DFS serialization decouples sequence position from physical coordinate. CHIP-STAR addresses each with a targeted, lightweight module: a dual-frame rotary encoding, a Fourier-factorized spatial prior, and a cumulative coordinate-noise schedule, together adding under $0.5\%$ parameters to a T5Gemma backbone. On the largest geometry-annotated routing corpus to date, CHIP-STAR-Large surpasses iPCL-Large by $\sim 5\%$ on Leaf IoU and Connectivity, while ECO signoff yields $3\%$ shorter wirelength and $11\%$ fewer vias. The framework extends to other DFS-serialized spatial-tree tasks such as molecular conformation [48], scene synthesis [34], and trajectory planning [1].

## Acknowledgments and Disclosure of Funding

This work was supported by the Major Key Project of Peng Cheng Laboratory, the NSF of Fujian Province under Grant 2024J09045, and the China Postdoctoral Science Foundation under Grant No. 2025M781480. We thank Weiguo Li for providing the initial code and for helpful discussions. The authors declare no competing interests.

## References


[1] Ahmed Alanazi, Duy Ho, and Yugyung Lee. Pathformer: A transformer with 3d grid constraints for digital twin robot-arm trajectory generation. *CoRR*, abs/2510.20161, 2025. doi: 10.48550/ARXIV.2510.20161. URL `https://doi.org/10.48550/arXiv.2510.20161`.

[2] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

[3] Cadence Design Systems. Innovus Implementation System, 2023. URL `https://www.cadence.com`.

[4] Dexiong Chen, Markus Krimmel, and Karsten Borgwardt. Flatten graphs as sequences: Transformers are scalable graph generators. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2502.02216.

[5] Huang-Yu Chen and Yao-Wen Chang. Global and detailed routing. In *Electronic Design Automation*, pages 687–749. Elsevier, 2009.

[6] Ruoyu Cheng, Xianglong Lyu, Yang Li, Junjie Ye, Jianye Hao, and Junchi Yan. The policy-gradient placement and generative routing neural networks for chip design. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, 2022.

[7] Chris Chu and Yiu-Chung Wong. FLUTE: Fast lookup table based rectilinear Steiner minimal tree algorithm for VLSI design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(1):70–83, 2008.

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[9] Jiang Hu and Sachin S. Sapatnekar. A survey on multi-net global routing for integrated circuits. *Integr.*, 31(1):1–49, 2001.

[10] Andrew B. Kahng, Lutong Wang, and Bangqi Xu. Tritonroute: The open-source detailed router. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 40(3):547–559, 2021.

[11] Andrew B. Kahng, Robert R. Nerem, Yusu Wang, and Chien-Yi Yang. NN-Steiner: A Mixed Neural-Algorithmic Approach for the Rectilinear Steiner Minimum Tree Problem. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(12):13022–13030, March 2024. doi: 10.1609/aaai.v38i12.29200.

[12] Robert Kirby, Saad Godil, Rajarshi Roy, and Bryan Catanzaro. CongestionNet: Routing congestion prediction using deep graph neural networks. In *IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pages 217–222. IEEE, 2019.

[13] Xinhua Lai, He Liu, Weiguo Li, Yihang Qiu, Miao Liu, Simin Tao, Xingquan Li, and Jungang Xu. iPCL-M1: Pre-training of chip layout for metrics evaluation and optimization. In *Proceedings of the 63rd ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2026.

[14] Xinhua Lai, He Liu, Miao Liu, Xingquan Li, and Jungang Xu. GT-Fusion: Synergizing LLM semantics, topology, and geometry for PPA prediction and optimization. In *Proceedings of the International Symposium of EDA (ISEDA)*. IEEE, 2026. Accepted for publication.

[15] Vint Lee, Minh Nguyen, Leena Elzeiny, Chun Deng, Pieter Abbeel, and John Wawrzynek. Chip placement with diffusion models. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pages 33499–33514. PMLR, 2025.

[16] Mingxiao Li, Tingyu Qu, Ruicong Yao, Wei Sun, and Marie-Francine Moens. Alleviating exposure bias in diffusion models through sampling with shifted time steps. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2305.15583.

[17] Xingquan Li, Weiguo Li, Xinhua Lai, Junfeng Liu, and Rui Wang. ipcl: Pre-training for chip layout (invited paper). In *Asia and South Pacific Design Automation Conference, ASP-DAC*, pages 1103–1109. IEEE, 2026. doi: 10.1109/ASP-DAC66049.2026.11420619. URL `https://doi.org/10.1109/ASP-DAC66049.2026.11420619`.

[18] Yangming Li and Mihaela van der Schaar. On error propagation of diffusion models. *International Conference on Learning Representations (ICLR)*, 2024.

[19] Haiguang Liao, Wentai Zhang, Xuliang Dong, Barnabas Poczos, Kenji Shimada, and Levent Burak Kara. A Deep Reinforcement Learning Approach for Global Routing. *Journal of Mechanical Design*, 142(061701), November 2019. doi: 10.1115/1.4045044.

[20] Yi-Lun Liao and Tess Smidt. Equiformer: Equivariant graph attention transformer for 3d atomistic graphs. In *International Conference on Learning Representations (ICLR)*, 2023.

[21] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

[22] Jinwei Liu, Gengjie Chen, and Evangeline F. Y. Young. REST: Constructing rectilinear Steiner minimum tree via reinforcement learning. In *Proceedings of the 58th ACM/IEEE Design Automation Conference (DAC)*, pages 1135–1140. IEEE, 2021. doi: 10.1109/DAC18074.2021.9586209.

[23] Junfeng Liu, Liwei Ni, Xingquan Li, Min Zhou, Lei Chen, Xing Li, Qinghua Zhao, and Shuai Ma. AiMap: Learning to improve technology mapping for ASICs via delay prediction. In *Proceedings of the 41st IEEE International Conference on Computer Design (ICCD)*, pages 344–347. IEEE, 2023. doi: 10.1109/ICCD58817.2023.00059.

[24] Junfeng Liu, Min Zhou, Shuai Ma, and Lujia Pan. MATA*: Combining learnable node matching with A* algorithm for approximate graph edit distance computation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1503–1512. ACM, 2023. doi: 10.1145/3583780.3614959.

[25] Junfeng Liu, Liwei Ni, Lei Chen, Xing Li, Qinghua Zhao, Xingquan Li, and Shuai Ma. A delay-driven iterative technology mapping framework. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 44(7), 2025. doi: 10.1109/TCAD.2024.3524463.

[26] Junfeng Liu, Qinghua Zhao, Liwei Ni, Jingren Wang, Biwei Xie, Xingquan Li, Bei Yu, and Shuai Ma. Bounded dynamic level maintenance for efficient logic optimization. *IEEE Transactions on Computers*, 75(7):2548–2562, 2026. doi: 10.1109/TC.2026.3688680.

[27] Miao Liu, Liwei Ni, Junfeng Liu, Xingyu Meng, Rui Wang, Xiaoze Lin, Xinhua Lai, Xingquan Li, and Jungang Xu. A survey of machine learning approaches in logic synthesis. *ACM Transactions on Design Automation of Electronic Systems*, 31(2):38:1–38:43, 2026. doi: 10.1145/3785362.

[28] Mingjie Liu et al. ChipNeMo: Domain-adapted LLMs for chip design. *arXiv preprint arXiv:2311.00176*, 2024.

[29] Ruizhi Liu, Zhisheng Zeng, Shizhe Ding, Jingyan Sui, Xingquan Li, and Dongbo Bu. NeuralSteiner: Learning Steiner Tree for Overflow-avoiding Global Routing in Chip Design. *Advances in Neural Information Processing Systems*, 37:127346–127368, December 2024.

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, 2021.

[31] Mang Ning, Enver Sangineto, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Input perturbation reduces exposure bias in diffusion models. In *International Conference on Machine Learning (ICML)*, 2023.

[32] Sophie Ostmeier, Brian Axelrod, Maya Varma, Michael Moseley, Akshay S Chaudhari, and Curtis Langlotz. Liere: Lie rotational positional encodings. In *International Conference on Machine Learning*, pages 47339–47355. PMLR, 2025.

[33] others. Multimodal chip physical design engineer assistant. *arXiv preprint arXiv:2510.15872*, 2025.

[34] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. ATISS: autoregressive transformers for indoor scene synthesis. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 12013–12026, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/64986d86a17424eeac96b08a6d519059-Abstract.html`.

[35] Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

[36] Yihang Qiu, Zengrong Huang, Simin Tao, Hongda Zhang, Weiguo Li, Xinhua Lai, Rui Wang, Weiqiang Wang, and Xingquan Li. Aieda: An open-source ai-aided design library for design-to-vector. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1–1, 2025. doi: 10.1109/TCAD.2025.3632188.

[37] Connor Schenck, Isaac Reid, Mithun George Jacob, Alex Bewley, Joshua Ainslie, David Rendleman, Deepali Jain, Mohit Sharma, Kumar Avinava Dubey, Ayzaan Wahid, et al. Learning the ropes: Better 2d and 3d position encodings with string. In *International Conference on Machine Learning*, pages 53295–53315. PMLR, 2025.

[38] Liangliang Shi, Shenhui Zhang, Xingbo Du, Nianzu Yang, and Junchi Yan. DSBRouter: End-to-end global routing via diffusion Schrödinger bridge. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pages 55065–55081. PMLR, 2025.

[39] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[40] Synopsys Inc. Design Compiler, 2023. URL `https://www.synopsys.com`.

[41] Xiaole Tang, Xiaoyi He, Jiayi Xu, Xiang Gu, and Jian Sun. Learning continuous Wasserstein barycenter space for generalized all-in-one image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

[42] Xiaole Tang, Jiayi Xu, Xiang Gu, Yan Yang, and Jian Sun. Binding multiple modalities via multimodal Wasserstein barycenter. *arXiv preprint arXiv:2609.33800*, 2026.

[43] The OpenROAD Project. OpenROAD, 2019. URL `https://github.com/The-OpenROAD-Project/OpenROAD`.

[44] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[46] Tai-Hsuan Wu, Azadeh Davoodi, and Jeffrey T. Linderoth. GRIP: Global routing via integer programming. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(1):72–84, 2011.

[47] Peng Xie, Ke Xue, Yunqi Shi, Ruo-Tong Chen, Chengrui Gao, Siyuan Xu, Chenjian Ding, Mingxuan Yuan, and Chao Qian. FlowPlace: Flow matching for chip placement. In *Proceedings of the 63rd ACM/IEEE Design Automation Conference (DAC)*, 2026.

[48] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

[49] Yupu Yao and Bowen Yang. Geope: A unified geometric positional embedding for structured tensors. *arXiv preprint arXiv:2512.04963*, 2025.

[50] Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton, and Jure Leskovec. GraphRNN: Generating realistic graphs with deep auto-regressive models. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 5694–5703, 2018.

[51] Biao Zhang, Fedor Moiseev, Joshua Ainslie, Paul Suganthan, Min Ma, Surya Bhupatiraju, Fede Lebron, Orhan Firat, Armand Joulin, and Zhe Dong. Encoder-decoder gemma: Improving the quality-efficiency trade-off via adaptation. *arXiv preprint arXiv:2504.06225*, 2025.

[52] Biao Zhang, Paul Suganthan, Gaël Liu, Ilya Philippov, Sahil Dua, Ben Hora, Kat Black, Gus Martins, Omar Sanseviero, Shreya Pathak, Cassidy Hardin, Francesco Visin, Jiageng Zhang, Kathleen Kenealy, Qin Yin, Xiaodan Song, Olivier Lacombe, Armand Joulin, Tris Warkentin, and Adam Roberts. T5gemma 2: Seeing, reading, and understanding longer. *CoRR*, abs/2512.14856, 2025. doi: 10.48550/ARXIV.2512.14856. URL `https://doi.org/10.48550/arXiv.2512.14856`.

[53] Lingxiao Zhao, Xueying Ding, and Leman Akoglu. PARD: Permutation-invariant autoregressive diffusion for graph generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2402.03687.

# A Detailed Preliminaries

## A.1 Routing Primitives: Extended Specification

We expand the four primitives summarized in §2.

**Wire.** Confined to a single metal layer. Width and layer choice set the per-segment RC, directly shaping power and timing.

**Via.** Connects wires across adjacent layers via a cut shape with top and bottom landings. EDA tools instantiate different via types with distinct dimensions and RC profiles depending on constraints, so via selection materially affects connectivity quality.

**Pin.** A cell's interface, composed of one or more rectangles that may span multiple layers. Intra-cell pins follow predefined logic; inter-cell pins are connected through nets formed by wires and vias.

**Patch.** A rectangular shape inserted during optimization (timing closure, IR-drop, signal integrity) to satisfy constraints with low routing-resource overhead.

From a routed `.def` file we build $G_{\text{topo}}$ from pairwise spatial adjacency among primitives, then fill pin and patch shapes with virtual Manhattan segments to obtain $G_{\text{wire}}$. The Steiner tree $T = (V, E, p, v_0)$ of Def. A.1 is the rooted-tree spanning subgraph of $G_{\text{wire}}$ for a given net, with $v_0$ the driver and leaves the loads.

## A.2 DecimalWordLevel Tokenization

Routing displacements span four orders of magnitude, from grid-pitch moves of a few units to chip-span moves of $10^6$ units. Standard BPE tokenizes such numbers by frequency rather than by magnitude: the displacement `2480` can split as `24|80` or `2|480`, severing the link to its preceding direction prefix and losing the place-value structure that distinguishes "move 80 units" from "move 8000 units."

**DecimalWordLevel** avoids this by reading each displacement the way a human reads a number: digit by digit, anchored to its place value. The edge `R2480` (move right 2480 units) decomposes into

$$\texttt{R2480} \;\longrightarrow\; \texttt{R2000 R400 R80},$$

one token per nonzero decimal place, each carrying its own direction prefix so that the geometric meaning survives every split. The vocabulary pairs each direction in $\mathcal{D} = \{\mathrm{R}, \mathrm{L}, \mathrm{U}, \mathrm{D}, \mathrm{T}, \mathrm{B}\}$ (right, left, up, down, top-via, bottom-via) with the nine nonzero digits $\{1, \ldots, 9\}$ at each decimal scale up to the chip span. This yields 207 tokens (the iPCL vocabulary) that cover the entire coordinate range of 28 nm designs while preserving two properties critical for geometric reasoning: every token is a self-contained direction-displacement primitive, and tokens at the same decimal scale share an embedding, giving the model a built-in multi-scale view of distance. CHIP-STAR adds 15 entries, used only to write the driver and load pin coordinates in absolute form when building the per-token coordinate annotation, for 222 in total; with the three modules removed they are never used (Appendix E.6).

## A.3 Formal Notation

**Definition A.1** (Routing Steiner Tree)**.** *A* routing Steiner tree *is a tuple $T = (V, E, p, v_0)$ where $V$ is a finite set of nodes, $E \subseteq V \times V$ is a set of undirected edges such that $(V, E)$ is a tree, $p : V \to \mathbb{R}^3$ assigns each node a 3D physical coordinate, and $v_0 \in V$ is a distinguished root (the driver).*

**Definition A.2** (DFS Serialization)**.** *A* DFS serialization *of $T$ rooted at $v_0$ is a sequence*

$$\tau = (t_1, t_2, \ldots, t_L), \qquad t_n \in V \cup \{\textsc{pop}\}, \tag{11}$$

*generated by the procedure:*

1. *Push $v_0$ and emit $t_1 = v_0$.*
2. *At each step, given the current node $v$, choose any unvisited neighbor $u$ of $v$ (the only source of nondeterminism), push $u$, and emit $u$.*
3. *When $v$ has no unvisited neighbors, emit* POP *and return to its parent.*

*4. Terminate when control returns to $v_0$ and $v_0$ has no unvisited neighbors.*

*We write $\mathcal{S}(T)$ for the set of all DFS serializations of $T$, and define $\pi_\tau(n) = t_n$.*

**Definition A.3** (Coordinate Function)**.** *For a serialization $\tau$, the* coordinate function $c_\tau : \{1, \dots, L\} \to \mathbb{R}^3$ *assigns to each sequence position $n$ the physical coordinate of the node currently at the top of the DFS stack after emitting $t_n$. Concretely, if $t_n = v \in V$ then $c_\tau(n) = p(v)$; if $t_n =$* POP *then $c_\tau(n) = p(\text{parent of last visited node})$.*

# B Extended Related Work

**Position encodings for high-dimensional inputs.** RoPE [39] encodes 1D index differences via orthogonal rotations and remains the default in modern LLMs. Several works extend it beyond 1D. LiePE [32] replaces the block-diagonal RoPE rotation with a learned dense rotation from $\mathfrak{so}(n)$. STRING [37] generalizes RoPE through Lie-group commuting generators and shows it is the most general separable encoding preserving translation invariance in arbitrary dimensions. GeoPE [49] recovers the 3D Euclidean manifold lost in 1D flattening using quaternionic sandwich products and $\mathfrak{so}(3)$ averaging. Equivariant models like Equiformer [20] achieve exact $\mathrm{SO}(3)$ equivariance through tensor-product features, but at heavy compute cost. STPE differs in the underlying problem: prior work assumes a regular input space where positional and geometric arguments coincide, while DFS-serialized routing decouples them, one tree admits $|\mathcal{S}(T)|$ valid serializations. STPE also avoids full equivariance, since the Manhattan grid gives routing physically privileged axes.

**Additive attention biases.** GeoPrior belongs to the family that adds positional or geometric information *additively* to attention logits, complementing the multiplicative rotary route. T5 [51, 52] learns a discrete relative-position bias; ALiBi [35] replaces it with a fixed linear penalty in $|i - j|$ and extrapolates well to longer sequences; Swin [30] looks up a 2D bias table by relative window position. All operate on regular index grids with content-independent biases. The Fourier factorization in Eq. 8 is the continuous 3D counterpart of Swin's table, swapping discrete lookups for learned frequency bands and reducing $\mathcal{O}(T^2)$ to $\mathcal{O}(T_q{+}T_k)$ encoding.

**Tree and graph autoregressive generation.** Sequence-linearized graph generation has progressed quickly. GraphRNN [50] emits nodes one by one under fixed BFS orderings. Transformer-based generators have followed: PARD [53] pairs autoregressive blocks with a discrete diffusion head and uses graph partial orders for permutation invariance, while AutoGraph [4] flattens graphs into reversible random sequences and trains a decoder-only Transformer end-to-end. AutoGraph is closest to our setting, its trail traversal mirrors DFS serialization, and it explicitly flags flattening ambiguity as a modeling challenge. The key difference is that graph generation is permutation-invariant by construction, whereas routing has a physically distinguished root (the driver) and anisotropic axes, so sequence order carries real physical meaning. STPE therefore keeps RoPE on top of the spatial channel rather than replacing it, and only quotients out residual ambiguity from DFS child-visit order.

**Exposure bias in autoregressive and diffusion generation.** Scheduled Sampling [2] addresses exposure bias in RNN sequence models by mixing teacher-forced and predicted tokens. Diffusion models hit the same problem in continuous space: training conditions on noisy inputs, inference on self-generated ones. ADM-IP [31] re-perturbs training samples to match the inference distribution; EB-DDPM [18] bounds the cumulative error and adds it as a regularizer; time-shift samplers [16] realign sampling timesteps without retraining. TrajNoise tackles the same gap in a third regime: the running coordinate $\mathbf{c}_t$ is recursively accumulated rather than denoised, so the natural perturbation is a cumulative random walk that matches the inference error process. Its variance schedule resembles DDPM [8] structurally but plays a different role-simulating drift, not driving generation.

**Machine learning for chip routing.** Classical routers like TritonRoute [10] rely on heuristic rip-up-and-reroute. Learning has been introduced for sub-tasks: congestion prediction [12], Steiner-point selection [11, 29], and RL-based global routing [19]. Broader LLM use in chip design includes ChipNeMo [28] for domain-adapted EDA scripting and multimodal assistants [33] for interpretable congestion analysis, but neither produces complete detailed routes end-to-end. Earlier stages of the flow have likewise benefited from learning-based and algorithmic advances, including logic synthesis [27], technology mapping [23, 25], and logic optimization [26]. The Seq2Seq routers

iPCL [17] and its successor [13] introduced the formulation that emits full direction-displacement sequences. CHIP-STAR builds on iPCL's tokenization and corpus, keeping the symbolic interface intact while addressing the geometric blindnesses from §2.

# C Notation

For reference, Table 4 lists the symbols used throughout §3 and the appendix proofs. Symbols are grouped by concept; their usage scope is indicated where relevant.

# D Proof of Theorem 3.1: DFS Permutation Invariance

This appendix provides the formal preliminaries, full statement, and proof of Theorem 3.1. We have algebraized the routing-task objects (trees, DFS serializations, coordinate functions in App. A.3) so that the claim has unambiguous meaning, then prove the two parts of the theorem in turn.

**Remark D.1.** *Two different DFS serializations of the same tree assign* different *sequence positions to the* same *physical node, but the coordinate function $c_\tau$ tracks the physical location regardless of which traversal is chosen. This is the core property exploited by SoPE.*

**Definition D.2** (SoPE Operator)**.** *For coordinate $p \in \mathbb{R}^3$ and frequency band index $k \in \{0, \dots, K-1\}$ with $K = \lfloor d_{\text{head}}/3 \rfloor$, the* SoPE operator $R_k(p) \in SO(3)$ *acts on the $k$-th 3D sub-vector of a query/key as*

$$R_k(p) = \exp\left( \sum_{a \in \{x,y,z\}} \tfrac{1}{3}\, \theta_{a,k}(p)\, J_a \right), \qquad \theta_{a,k}(p) = p_a \cdot f_k, \tag{12}$$

*where $\{J_x, J_y, J_z\}$ are the standard skew-symmetric generators of $\mathfrak{so}(3)$ and $f_k = \text{base}^{-k/K}$. The full operator $R(p)$ acts as $R_k(p)$ blockwise on each of the $K$ 3D sub-vectors of the head.*

**Definition D.3** (Attention Scores under STPE)**.** *Given queries $q$, keys $k$, sequence positions $n_i, n_j$, and coordinates $p_i, p_j$, the STPE attention score is*

$$s(q, k; n_i, n_j, p_i, p_j) = \big\langle R(p_i)\,\mathrm{RoPE}(n_i)\,q,\ R(p_j)\,\mathrm{RoPE}(n_j)\,k \big\rangle / \sqrt{d}. \tag{13}$$

*The* SoPE-only *and* RoPE-only *scores, used to isolate each component, are*

$$s_{\mathrm{SoPE}}(q, k; p_i, p_j) = \big\langle R(p_i)\,q,\ R(p_j)\,k \big\rangle / \sqrt{d}, \tag{14}$$

$$s_{\mathrm{RoPE}}(q, k; n_i, n_j) = \big\langle \mathrm{RoPE}(n_i)\,q,\ \mathrm{RoPE}(n_j)\,k \big\rangle / \sqrt{d}. \tag{15}$$

## D.1 Proof of Theorem 3.1

For convenience we restate the theorem in the form used in the main text.

**Theorem D.4** (Restatement of Theorem 3.1)**.** *Let $T$ be a routing Steiner tree, $\tau_1, \tau_2 \in \mathcal{S}(T)$, and $u, v \in T$ any two tree nodes. For all $q, k \in \mathbb{R}^d$,*

$$s_{\mathrm{SoPE}}\big(q, k;\, p_u, p_v\big) \;=\; \big\langle \mathrm{SoPE}(p_u)\,q,\ \mathrm{SoPE}(p_v)\,k \big\rangle / \sqrt{d} \tag{16}$$

*takes the same value under $\tau_1$ and $\tau_2$. The analogous identity for $s_{\mathrm{RoPE}}$ does not hold: there exist $T, \tau_1, \tau_2$ and $q, k$ for which the two sides differ.*

The proof has two parts. The bridge between them is a small lemma extracting the key property of the coordinate function $c_\tau$ (Def. A.3).

**Lemma D.5** (Coordinate consistency at tree nodes)**.** *For any tree node $u \in T$ and any DFS serialization $\tau \in \mathcal{S}(T)$, let $n_u^{(\tau)}$ denote the sequence position at which $u$ is emitted. Then*

$$c_\tau\big(n_u^{(\tau)}\big) \;=\; p_u, \tag{17}$$

*independently of $\tau$.*

*Proof.* By construction (Def. A.3), $c_\tau(n)$ returns the top of the DFS coordinate stack at position $n$. When $u$ is itself emitted, the stack top is $p_u$, regardless of the order in which $u$'s siblings or ancestors' siblings have been visited. Hence $c_\tau(n_u^{(\tau)}) = p_u$ for every $\tau$. □

Table 4: Symbols used in §3 and proofs.

| Symbol | Meaning |
|---|---|
| *Routing tree and serialization (§2)* | |
| $T = (V, E, p, v_0)$ | Routing Steiner tree: nodes $V$, edges $E$, coordinate map $p$, driver $v_0$. |
| $\tau, \mathcal{S}(T)$ | A DFS serialization of $T$, and the set of all valid DFS serializations. |
| $n_i$ | Sequence index of token $i$ in $\tau$. |
| $c_\tau(n)$ | DFS-stack-top coordinate function: the physical coordinate at sequence position $n$ under $\tau$. |
| $n_u^{(\tau)}$ | Sequence position of tree node $u$ under serialization $\tau$ (used in proofs). |
| *Physical coordinates (§3.1)* | |
| $p_i \in \mathbb{R}^3$ | Normalized 3D coordinate of token $i$, driver-relative. |
| $x, y, z$ | Components of $p$, with $x, y$ in-plane and $z$ indexing metal layer. |
| $\lambda_{xy}, \lambda_z$ | Per-axis rescaling factors; $p = (\lambda_{xy} x_{\text{raw}}, \lambda_{xy} y_{\text{raw}}, \lambda_z z_{\text{raw}})$. |
| $p_t$ | Decoder coordinate at step $t$, accumulated as $p_t = p_{t-1} + \Delta_t$. |
| $\Delta_t$ | 3D displacement of the $t$-th decoder token (e.g., `R10` $\to (10, 0, 0)$). |
| $\tilde{p}_t$ | TrajNoise-perturbed decoder coordinate; replaces $p_t$ in STPE and GeoPrior at training time. |
| *Attention (§3.2)* | |
| $q, k \in \mathbb{R}^d$ | Raw query and key vectors. |
| $\hat{q}_i, \hat{k}_j$ | STPE-rotated query/key: $\hat{q}_i = \mathrm{SoPE}(p_i)\,\mathrm{RoPE}(q, n_i)$. |
| $s_{ij}$ | Attention score between positions $i, j$. |
| $d, d_{\text{head}}$ | Model dimension and per-head dimension. |
| *SoPE: multi-scale quaternion rotation (§3.2.2)* | |
| $\ell \in \{1, \ldots, K\}$ | Frequency-band index. |
| $K = \lfloor d_{\text{head}}/3 \rfloor$ | Number of 3D sub-vectors per head. |
| $f_\ell, \theta_{\text{base}}$ | Frequency of band $\ell$: $f_\ell = \theta_{\text{base}}^{-\ell/K}$, with $\theta_{\text{base}} = 1000$. |
| $\Theta_\ell$ | $\mathfrak{so}(3)$-averaged phase magnitude in band $\ell$. |
| $R_\ell$ | Per-band unit quaternion implementing the SoPE rotation. |
| $A_\ell, \mathbf{a}_\ell$ | *(Used in §3.2.3 only.)* Relative rotation angle and axis induced by $p_i - p_j$ in band $\ell$. |
| *GeoPrior: factorized spatial prior (§3.3)* | |
| $\phi(p_i, p_j)$ | Additive spatial prior added to the cross-attention score. |
| $\mathbf{f}_i \in \mathbb{R}^{2F}$ | Per-token Fourier encoding $[\sin(\mathbf{W}_\omega p_i),\ \cos(\mathbf{W}_\omega p_i)]$. |
| $\mathbf{W}_\omega$ | Learnable frequency projection, $\mathbb{R}^{F \times 3}$. |
| $\mathbf{W}_q, \mathbf{W}_k$ | Per-head rank-$r$ projection matrices for the factorization. |
| $F, r$ | Number of Fourier bands; per-head rank. |
| *TrajNoise: spatial scheduled sampling (§3.4)* | |
| $L$ | Decoder sequence length. |
| $t$ | Decoder step index, $t \in \{1, \ldots, L\}$. |
| $\boldsymbol{\epsilon}_u$ | Per-step i.i.d. noise, $\mathcal{N}(\mathbf{0}, \mathbf{I})$. |
| $\boldsymbol{\eta}_t$ | Cumulative noise, $\boldsymbol{\eta}_t = (\sigma/\sqrt{L}) \sum_{u=1}^{t} \boldsymbol{\epsilon}_u$. |
| $\sigma = (\sigma_{xy}, \sigma_{xy}, \sigma_z)$ | Per-axis noise scale in normalized coordinate space. |

Note this consistency holds only at tree nodes; at POP tokens, $c_\tau$ depends on the stack history and does vary with $\tau$. The restatement above therefore restricts $u, v$ to tree nodes.

**Part (a): SoPE invariance.** Applying Lemma D.5 to both $u$ and $v$,

$$c_{\tau_1}\big(n_u^{(\tau_1)}\big) = c_{\tau_2}\big(n_u^{(\tau_2)}\big) = p_u, \qquad c_{\tau_1}\big(n_v^{(\tau_1)}\big) = c_{\tau_2}\big(n_v^{(\tau_2)}\big) = p_v. \tag{18}$$

Substituting into the SoPE score (Def. D.3),

$$s_{\mathrm{SoPE}}\big(q, k;\, c_{\tau_i}(n_u^{(\tau_i)}),\, c_{\tau_i}(n_v^{(\tau_i)})\big) = \big\langle \mathrm{SoPE}(p_u)\, q,\ \mathrm{SoPE}(p_v)\, k \big\rangle / \sqrt{d} \tag{19}$$

for both $i \in \{1, 2\}$. The right-hand side references neither $\tau$ nor the sequence indices, so the two sides are equal.

This argument applies only to the SoPE-only score; the full STPE score retains a RoPE factor that does depend on $\tau$, as Part (b) shows.

**Part (b): RoPE non-invariance, by counterexample.** Consider a three-node tree $T$ with driver $v_0$ at $p_{v_0} = (0, 0, 0)$ and two loads $u, v$ connected directly to $v_0$ at $p_u = (a, 0, 0)$, $p_v = (0, a, 0)$ for some $a > 0$. Since $v_0$ has two children, $T$ admits two DFS orderings:

$$\tau_1 = \big(v_0,\ u,\ \textsc{pop},\ v,\ \textsc{pop}\big), \quad n_u^{(\tau_1)} = 2,\ n_v^{(\tau_1)} = 4, \tag{20}$$

$$\tau_2 = \big(v_0,\ v,\ \textsc{pop},\ u,\ \textsc{pop}\big), \quad n_u^{(\tau_2)} = 4,\ n_v^{(\tau_2)} = 2. \tag{21}$$

The relative offset between $u$ and $v$ thus flips sign across the two serializations: $\Delta^{(1)} := n_v^{(\tau_1)} - n_u^{(\tau_1)} = +2$ versus $\Delta^{(2)} := n_v^{(\tau_2)} - n_u^{(\tau_2)} = -2$. Since RoPE attention depends on the index difference,

$$s_{\mathrm{RoPE}}\big(q, k;\, n_u^{(\tau_i)}, n_v^{(\tau_i)}\big) = \big\langle q,\ \mathrm{RoPE}(\Delta^{(i)})\, k \big\rangle / \sqrt{d}. \tag{22}$$

We exhibit explicit $q, k$ making the two sides unequal. Take $d = 2$ for clarity (the argument lifts to higher dimensions by embedding into the first $2 \times 2$ block). Let $\theta_1$ be the RoPE base frequency for this block, and set $q = (1, 0)^\top$, $k = (0, 1)^\top$. A direct computation gives

$$\big\langle q,\ \mathrm{RoPE}(\Delta)\, k \big\rangle = -\sin(\Delta\, \theta_1), \tag{23}$$

so

$$s_{\mathrm{RoPE}}^{(1)} - s_{\mathrm{RoPE}}^{(2)} = \frac{1}{\sqrt{d}}\Big( -\sin(2\theta_1) + \sin(-2\theta_1)\Big) = -\frac{2\sin(2\theta_1)}{\sqrt{d}}. \tag{24}$$

For standard RoPE base frequencies $\theta_m = \theta_{\text{base}}^{-2(m-1)/d}$ with $\theta_{\text{base}} \gg 1$, $\theta_1 \notin (\pi/2)\mathbb{Z}$, hence $\sin(2\theta_1) \neq 0$ and the two scores differ. □

### D.2 Discussion

The counterexample uses a minimal three-node star with two DFS orderings; the same construction extends to arbitrary trees with $\geq 2$ branching choices, where the number of distinct serializations grows combinatorially. Theorem 3.1 therefore certifies that the SoPE component of STPE eliminates an entire class of spurious order-dependent attention signals — signals that would otherwise have to be averaged out by training-time data augmentation over DFS orderings, at substantial cost.

## E Details of Experimental Setup

### E.1 Architecture and Implementation Details

Table 5 summarizes which attention modules receive geometric extensions.

### E.2 Metric Definitions

We evaluate generated routing trees along two complementary dimensions: (i) *topology-level* metrics that compare the generated tree against the ground-truth tree as a discrete structure, and (ii) *physical-design* metrics that measure the downstream quality of the routed layout after the tree is realized in the back-end flow.

Table 5: **Module placement across attention layers.** STPE and GeoPrior are inference-time architectural extensions; TrajNoise is a training-only regularizer (denoted "—" in inference rows). The encoder ingests net descriptions where token order is canonical and driver-relative coordinates are unnecessary, so no geometric extension is applied; all geometric mechanisms act on the decoder, where DFS-induced spatial blindness arises. GeoPrior is applied only in cross-attention, where the decoder position meets the load coordinates carried by encoder tokens (§3.3). All coordinates are relative to the driver and coordinate-scaled to $O(1)$–$O(10)$. The last three rows are the remaining places where coordinates could enter the model; none of them reads the coordinate field, so with the three modules removed the field is unused (Appendix E.6).

| Component | STPE | GeoPrior | TrajNoise (train only) |
|---|---|---|---|
| Encoder self-attn | × | × | — |
| Decoder self-attn | ✓ | × | ✓ |
| Cross-attn | ✓ | ✓ | ✓ |
| Token embedding | × | × | × |
| Sequence length / attention mask | × | × | × |
| Training objective (CE on targets) | × | × | × |

**Topology-level metrics.**

- **Connectivity** (%): fraction of nets in which *all* driver–load pairs are connected in the generated routing tree.
- **Graceful** (%): a partial-connectivity score that credits nets with at least some loads connected; formally, $\frac{1}{|\mathcal{N}|}\sum_{n\in\mathcal{N}}\frac{|\text{connected loads in } n|}{|\text{total loads in } n|}$.
- **Leaf IoU** (%): Intersection-over-Union of the set of leaf-level (load) nodes between the generated tree and the ground-truth tree, averaged over all nets.
- **Exact Match** (%): fraction of nets whose generated token sequence is *byte-identical* to the ground-truth sequence; dominated by single-load nets where no branching ambiguity exists.
- **Branch Match** (%): fraction of branch decisions that match the ground truth; measures structural correctness at branch points independently of the exact token sequence.
- **ROUGE-L** (%): longest-common-subsequence based overlap; included for ablation completeness but less interpretable than the routing-specific metrics above.

**Physical-design metrics.** These metrics are reported by the back-end router/STA flow after the generated tree is committed, and reflect the realized quality of the layout rather than its symbolic agreement with the ground truth. Lower values are better for all six.

- **Wirelength (WL)** (µm): total routed length across all metal layers, summed over all nets. A primary proxy for routing efficiency, congestion pressure, and dynamic-power cost on signal nets.
- **#Vias**: total number of vias inserted by the router across all nets. Vias incur resistance, manufacturing-yield penalties, and parasitic capacitance, so a topologically faithful tree should not inflate this count relative to the commercial flow.
- **Switching Power (Sw. Power)** (µW): dynamic power dissipated by signal-net charging and discharging, computed as $\frac{1}{2}\sum_n \alpha_n C_n V_{dd}^2 f$ where $C_n$ aggregates wire and via capacitance on net $n$. Directly tracks routing-induced capacitance and is the most sensitive electrical metric to tree topology.
- **Total Power (Pwr)** (µW): sum of switching, internal, and leakage power for the full design as reported by signoff power analysis. Routing changes mainly perturb the switching component, so Total Power is reported as a sanity check that gains on Sw. Power do not regress elsewhere.
- **Worst Negative Slack (WNS)** (ps): the largest setup-timing violation across all paths reported by static timing analysis; values closer to zero (or positive) indicate a faster-closing design. Sensitive to wire delay on critical nets and therefore to tree depth and load balance.
- **CPU Time** (s): wall-clock runtime of generating the routing tree, including model inference and any ECO refinement, measured on identical hardware. Reported to characterize the practical cost of replacing the commercial routing engine with a learned model.

### E.3 Training Hyperparameters

Table 6 lists the full hyperparameter configuration used to train CHIP-STAR-Medium, shared across all training runs in the main paper unless explicitly stated otherwise.

Table 6: Training hyperparameters for CHIP-STAR-Medium. Coordinate scales $\lambda_{xy}, \lambda_z$ map physical µm to the normalized range used by STPE; trajectory-noise standard deviations $\sigma_{xy}, \sigma_z$ are reported in the same normalized units.

| **Hyperparameter** | **Value** | **Notes** |
|---|---|---|
| *Backbone (T5Gemma)* | | |
| Parameters | 0.4B | shared with baselines |
| Hidden size | 1024 | |
| Layers / Heads | 12 / 16 | encoder–decoder |
| *Spatio-Temporal Positional Encoding (STPE)* | | |
| Frequency base $\theta_{\text{base}}$ | 1,000 | |
| Coordinate scale $\lambda_{xy}$ | $3\times10^{-5}$ | in-plane normalization |
| Coordinate scale $\lambda_z$ | 1.0 | metal-layer index, no scaling |
| *Geometric Prior (GeoPrior)* | | |
| Fourier frequencies $F$ | 16 | |
| Rank per head $r$ | 8 | low-rank attention bias |
| *Trajectory Noise (TrajNoise)* | | |
| $\sigma_{xy}$ | 5 | in normalized units |
| $\sigma_z$ | 2 | one layer per step |
| Max corruption ratio $\rho_{\max}$ | 2 | 20% clean samples retained |
| Noise schedule | cumulative | Brownian-motion process |
| *Optimization* | | |
| Optimizer | Adafactor | |
| Learning rate | $1\times10^{-4}$ | constant |
| Batch size | 36 | |
| Epochs | 12 | |

### E.4 Dataset Statistics

Designs are drawn from ISCAS'89, OpenROAD [43], OpenCores, and tape-out projects including *ysyx* and iEDA [36].[2]

We provide per-case statistics for the training and validation corpora used throughout the experiments. Tables 7 and 8 report the total number of nets (**#Nets**) and the number of routable nets (**#Routable Nets**) for each design. Routable nets exclude power, ground, and clock nets, as well as nets whose driver or loads fall outside the routable region; only routable nets contribute to the training loss and the evaluation metrics in the main paper.

### E.5 Inference Procedure

At inference, CHIP-STAR decodes autoregressively with greedy decoding. Decoder coordinates $\mathbf{c}_t$ are accumulated from the driver origin: $\mathbf{c}_t = \mathbf{c}_{t-1} + \Delta_t$, where $\Delta_t$ is the spatial displacement implied by the predicted token at step $t$. TrajNoise is disabled at inference (it is training-only). Generated sequences are terminated at ⟨EOS⟩; any mismatched PUSH/POP pairs that remain open are forcibly closed before the sequence is passed to the downstream EDA flow. Residual DRC violations are resolved by Innovus ECO post-processing; we report DRC counts both before and after ECO.

[2] ISCAS'89: `http://www.cbl.ncsu.edu`; OpenCores: `http://www.opencores.org`; *ysyx*: `https://ysyx.oscc.cc/`.

Table 7: Training design statistics. The corpus spans three orders of magnitude in net count, from 132-net microcontroller blocks to 1.2M-net SoCs, exposing the model to a wide range of routing-tree complexities.

| Case | #Nets | #Routable Nets |
|---|---|---|
| s44 | 132 | 127 |
| gcd | 286 | 268 |
| s1488 | 348 | 324 |
| apb4_ps2 | 505 | 429 |
| apb4_i2c | 731 | 671 |
| apb4_timer | 791 | 650 |
| apb4_pwm | 892 | 834 |
| apb4_clint | 1035 | 964 |
| s15850 | 2063 | 1911 |
| s38417 | 5628 | 5521 |
| s35932 | 5813 | 5776 |
| s38584 | 6832 | 6539 |
| PPU | 8904 | 8811 |
| picorv32 | 9148 | 8953 |
| BM64 | 9467 | 9462 |
| blabla | 16,661 | 15,631 |
| aes_core | 17,285 | 17,280 |
| aes | 18,013 | 18,007 |
| salsa20 | 20,366 | 20,298 |
| jpeg_encoder | 29,045 | 28,996 |
| yadan_riscv_sopc | 33,753 | 29,628 |
| eth_top | 38,275 | 38,178 |
| shanghai_MS | 305,438 | 244,496 |
| iEDA_2023 | 373,858 | 325,756 |
| ZJU_asic_top | 395,015 | 312,143 |
| ysyx4_SoC2 | 504,844 | 445,340 |
| AIMP | 574,755 | 506,299 |
| AIMP_2.0 | 618,201 | 504,941 |
| ysyx6 | 1,121,535 | 998,374 |
| ysyx4_SoC1 | 1,217,738 | 1,097,095 |
| ysyx4_2 | 1,257,005 | 1,139,131 |
| **Total** | 6,624,531 | 5,808,063 |

Table 8: Validation design statistics. Validation cases are held out from the training corpus and span two orders of magnitude in net count, with no overlap with the training set.

| Case | #Nets | #Routable Nets |
|---|---|---|
| s713 | 132 | 120 |
| apb4_rng | 207 | 167 |
| s1238 | 294 | 289 |
| apb4_archinfo | 383 | 343 |
| s9234 | 624 | 573 |
| ASIC | 804 | 784 |
| s5378 | 812 | 767 |
| apb4_wdg | 948 | 906 |
| apb4_uart | 5722 | 5513 |
| **Total** | 9926 | 9462 |

### E.6 Controlled Comparison with iPCL

iPCL and CHIP-STAR are not two separately tuned pipelines, but one pipeline with the three modules switched on or off. Table 9 lists the configuration of both: they share the nets and target sequences, the backbone and its pretrained checkpoint, the optimizer and training budget, the decoding procedure

and the downstream ECO flow, and differ only in the three modules and the coordinate annotation that the modules read.

**Only the three modules read the coordinate field.** Table 5 lists every component of the model that could read the per-token coordinate field. With STPE, GeoPrior and TrajNoise removed, none of them reads it, the data loader drops it, and what remains is the unmodified T5Gemma encoder–decoder trained on the same target sequences with the same objective, *i.e.,* iPCL. iPCL-Medium in Table 2 is therefore CHIP-STAR-Medium with all three modules switched off, and the only variable across the rows of that table is which modules are active.

**Corpus size (82 GB vs. 556 GB).** Both corpora contain the same 6.4M nets and the same token sequences. The 474 GB difference is the per-token coordinate field, which is a deterministic function of the token sequence (Definition A.3) and can be recomputed by a short parser without any external source. The coordinates are stored as a parallel field, not as additional tokens, so encoder and decoder lengths, padding, truncation and attention-mask shapes are identical in both settings.

**Vocabulary (207 vs. 222 tokens).** The 15 additional entries are introduced by preprocessing to write the driver and load pin coordinates in absolute form, from which the per-token annotation is built (Appendix A.2). They serve only the annotation: with the three modules removed they are never used, and the model falls back to the 207 tokens of iPCL.

**Initialization and decoding.** Both models start from the same official T5Gemma checkpoint, inherit all encoder–decoder transformer weights, and re-initialize the embedding matrix for the routing vocabulary. For the new modules, GeoPrior is zero-initialized, STPE has no learnable weights and reduces to RoPE at the driver origin, and the TrajNoise magnitude is ramped up from zero. Both models decode with the same procedure (Table 9).

# F Supplementary Experimental Results

## F.1 Training Variance Across Seeds

The main results (Table 1) and the ablation (Table 2) each come from a single training run. To measure the variance due to training stochasticity, we retrain three Medium configurations with three random seeds each: iPCL-Medium, CHIP-STAR-Medium without STPE, and the full CHIP-STAR-Medium. Apart from the seed and the number of epochs, every setting follows Table 9. To fit the compute budget, each run trains for 5 epochs instead of 12, so the absolute values are lower than in Table 2 and should be compared only within Table 10.

The seed standard deviation of Leaf IoU is at most $0.004$ for every configuration. The full model exceeds iPCL-Medium by $0.026$, more than six times this deviation, and every full-model run scores above every run without STPE. Without STPE, the remaining two modules lift Leaf IoU by only $0.005$ over iPCL-Medium, and the runs of the two configurations overlap; three seeds are not enough to separate this gain from noise.

## F.2 Detailed Physical-Design Quality

We assess whether routing trees generated by CHIP-STAR produce competitive back-end layouts when committed to the commercial place-and-route flow. For each validation case, we replace the commercial router's tree with the one predicted by iPCL or CHIP-STAR, run the standard signoff flow, and report wirelength, via count, switching power, total power, and worst negative slack from the post-route reports.

Table 11 summarizes the results. CHIP-STAR matches the commercial flow within 1–3% on every electrical metric (WL, Pwr, Sw. Pwr, WNS), and improves over iPCL on #Vias by a relative 6% ($1.18\times$ vs. $1.25\times$ of Com.) without regressing any other metric. The remaining gap on #Vias reflects the cost of replacing a hand-tuned router with a learned model, but is substantially smaller than that of the no-geometry baseline—confirming that the geometric inductive bias translates topology-level improvements (Table 1) into measurable layout-quality gains.

Table 9: Configuration of iPCL and CHIP-STAR. "Same" means identical to iPCL. Hidden size, layers and heads are those of the Large models.

| **Item** | **iPCL** | **CHIP-STAR** |
|---|---|---|
| *Data* | | |
| Training nets | 6.4M nets | same nets and target sequences |
| Tokenizer | DecimalWordLevel, 207 tokens | DecimalWordLevel, 222 tokens (15 for the coordinate annotation) |
| Corpus on disk | 82 GB | 556 GB (adds the per-token coordinate field) |
| *Model* | | |
| Backbone | T5Gemma encoder–decoder | same, plus STPE, GeoPrior, TrajNoise |
| Hidden / layers / heads | 1024 / 12 / 16 | same |
| Parameters (Medium / Large) | 52M / 416M | 53M / 418M |
| Checkpoint | official T5Gemma release | same |
| Inherited weights | all encoder–decoder transformer weights | same |
| Embedding matrix | re-initialized for the routing vocabulary | same |
| New-module init | — | GeoPrior zero-initialized; STPE weight-free, reduces to RoPE at the driver origin; TrajNoise ramped from 0 |
| *Training* | | |
| Optimizer | Adafactor, lr $1 \times 10^{-4}$ | same |
| Batch / epochs / steps | 36 / 12 / $\sim$2.1M | same |
| Hardware | 16$\times$ V100 (32 GB) | same |
| *Inference and evaluation* | | |
| Decoding | greedy (beam size 1), deterministic; 1 sample per net, no reranking | same |
| Max decode length | 512 | same |
| Termination | EOS; unmatched PUSH/POP forcibly closed | same |
| ECO / signoff | Cadence Innovus, identical scripts | same |

Table 10: Leaf IoU of three Medium configurations over three random seeds, 5 training epochs per run. Mean $\pm$ Std is the mean and the sample standard deviation ($n-1$) over the three seeds.

| **Variant** | **Seed 1** | **Seed 2** | **Seed 3** | **Mean $\pm$ Std** |
|---|---|---|---|---|
| CHIP-STAR-Medium (full) | 0.895 | 0.891 | 0.898 | $0.895 \pm 0.004$ |
| w/o STPE | 0.874 | 0.877 | 0.870 | $0.874 \pm 0.004$ |
| iPCL-Medium *(no geometry)* | 0.868 | 0.871 | 0.869 | $0.869 \pm 0.002$ |

Table 11: Comparison of design metrics on benchmark circuits. Arrows indicate the preferred direction (↓: lower is better). The bottom row reports the geometric-mean ratio normalized to the commercial flow (Com.); **bold marks the better of iPCL and CHIP-STAR per metric.**

| | **#Vias** ↓ | | | **WL** (µm) ↓ | | | **Sw. Pwr** (µW) ↓ | | | **Pwr** (µW) ↓ | | | **WNS** (ps) ↓ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Case** | Com. | iPCL | CHIP. | Com. | iPCL | CHIP. | Com. | iPCL | CHIP. | Com. | iPCL | CHIP. | Com. | iPCL | CHIP. |
| ASIC | 5525 | 7629 | 6891 | 6170 | 6505 | 6200 | 40 | 39 | 39 | 193 | 193 | 192 | 1438 | 1442 | 1442 |
| apb4_archinfo | 2606 | 3254 | 2970 | 2420 | 2465 | 2422 | 15 | 14 | 14 | 199 | 198 | 198 | 1175 | 1177 | 1180 |
| apb4_rng | 1090 | 1375 | 1303 | 1287 | 1308 | 1318 | 10 | 10 | 10 | 115 | 115 | 115 | 1146 | 1150 | 1153 |
| apb4_uart | 43 013 | 56 569 | 50 169 | 45 897 | 47 437 | 46 479 | 330 | 322 | 320 | 3251 | 3243 | 3241 | 757 | 763 | 764 |
| apb4_wdg | 7245 | 8911 | 8465 | 6404 | 6527 | 6460 | 46 | 45 | 45 | 358 | 357 | 357 | 558 | 584 | 570 |
| s1238 | 2664 | 3523 | 3322 | 1958 | 2050 | 2034 | 19 | 18 | 18 | 80 | 80 | 80 | 953 | 956 | 962 |
| s13207 | 4293 | 4985 | 4788 | 3781 | 3831 | 3768 | 27 | 26 | 26 | 484 | 483 | 483 | 911 | 920 | 919 |
| s5378 | 6513 | 7958 | 7591 | 6287 | 6412 | 6359 | 38 | 38 | 38 | 418 | 417 | 417 | 836 | 847 | 849 |
| s713 | 713 | 852 | 800 | 584 | 588 | 587 | 6 | 5 | 5 | 46 | 46 | 46 | 915 | 925 | 919 |
| s9234 | 4534 | 5478 | 5475 | 3573 | 3649 | 3629 | 25 | 24 | 24 | 316 | 315 | 315 | 693 | 702 | 703 |
| **Ratio** | 1.00 | 1.25 | **1.18** | 1.00 | 1.03 | **1.01** | 1.00 | 0.96 | **0.96** | 1.00 | 1.00 | **1.00** | 1.00 | 1.01 | **1.01** |

### F.3 Physical Metrics Before and After ECO

To separate the contribution of the routing model from that of ECO, we also measure the physical metrics on the raw model output, before the ECO pass. Figure 10 plots the change of CHIP-STAR relative to iPCL on each design, before and after ECO. Before ECO, CHIP-STAR uses fewer vias ($-11.1\%$ in total) and less wirelength ($-4.8\%$) than iPCL on every design, and leaves fewer DRC violations on 8 of 10 designs ($-39.7\%$ in total, $-14.1\%$ on average per design; the total is dominated by `apb4_uart`). ECO then adds more wire to the output of CHIP-STAR than to that of iPCL ($+517$ vs. $+293$ µm per design on average) and removes fewer vias ($-144$ vs. $-433$), which narrows the gap to $-1.9\%$ in wirelength and $-8.7\%$ in vias. On one design (`apb4_rng`) the wirelength advantage turns into a $+0.8\%$ deficit after ECO. The advantage therefore originates in the model output, and ECO reduces rather than creates it.

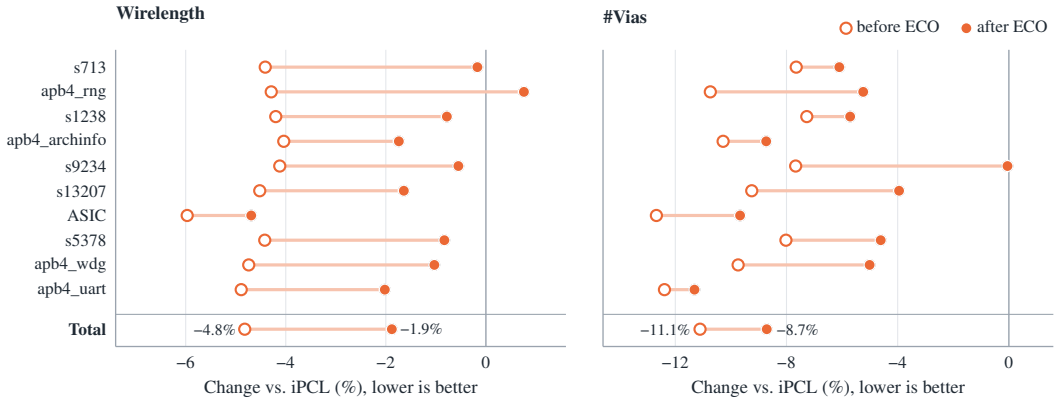



Figure 10: Change of CHIP-STAR relative to iPCL per held-out design, before ECO (○) and after ECO (●); negative is better. The last row is the change of the total. Every design starts with fewer vias and less wirelength before ECO, and ECO narrows the gap.

### F.4 CPU Time Breakdown

Table 12 splits the CPU time of the back-end flow into model inference and ECO, and Figure 11 plots both relative to the CPU time of the commercial router on each design. Inference of CHIP-STAR costs about $2.2\times$ that of iPCL on every design ($0.24\times$ vs. $0.11\times$ the router's CPU time in total), whereas ECO time is comparable ($0.34\times$ vs. $0.32\times$; lower for CHIP-STAR on 4 of 10 designs). The extra CPU time of CHIP-STAR thus comes from the model forward pass, not from a heavier repair burden. In total, CHIP-STAR needs $58\%$ of the router's CPU time and iPCL $43\%$; CHIP-STAR exceeds the router only on the largest design, `apb4_uart` ($1.02\times$), where inference dominates.

Table 12: CPU time (s) of the back-end flow per held-out design, split into model inference and ECO. Router: CPU time of the commercial router on the same design. Total / Router divides inference + ECO by the router's CPU time; the last row divides the column sums by the router's sum.

| | | Inference | | ECO | | Total | | Total / Router | |
|---|---|---|---|---|---|---|---|---|---|
| **Case** | **Router** | iPCL | ChiP. | iPCL | ChiP. | iPCL | ChiP. | iPCL | ChiP. |
| s713 | 141 | 2.6 | 5.8 | 41.0 | 39.1 | 43.6 | 44.9 | 0.31 | 0.32 |
| apb4_rng | 143 | 4.1 | 9.1 | 40.3 | 42.2 | 44.4 | 51.3 | 0.31 | 0.36 |
| s1238 | 168 | 5.8 | 13.0 | 57.6 | 51.5 | 63.4 | 64.5 | 0.38 | 0.38 |
| apb4_archinfo | 156 | 7.5 | 16.9 | 46.5 | 50.8 | 54.0 | 67.7 | 0.35 | 0.43 |
| s9234 | 173 | 12.2 | 27.5 | 56.9 | 55.3 | 69.1 | 82.8 | 0.40 | 0.48 |
| s13207 | 170 | 15.3 | 34.4 | 51.8 | 58.7 | 67.1 | 93.1 | 0.39 | 0.55 |
| ASIC | 196 | 15.8 | 35.4 | 85.3 | 56.9 | 101.1 | 92.3 | 0.52 | 0.47 |
| s5378 | 203 | 15.9 | 35.8 | 61.2 | 72.1 | 77.1 | 107.9 | 0.38 | 0.53 |
| apb4_wdg | 190 | 18.6 | 41.8 | 64.7 | 66.7 | 83.3 | 108.5 | 0.44 | 0.57 |
| apb4_uart | 413 | 112.2 | 252.1 | 126.2 | 170.2 | 238.4 | 422.3 | 0.58 | 1.02 |
| Sum | 1953 | 210.0 | 471.8 | 631.5 | 663.5 | 841.5 | 1135.3 | | |
| Sum / Router | 1.00 | 0.11 | 0.24 | 0.32 | 0.34 | 0.43 | 0.58 | | |

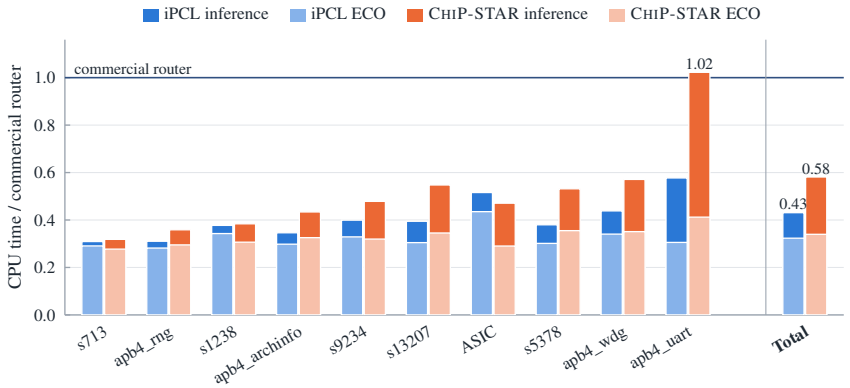
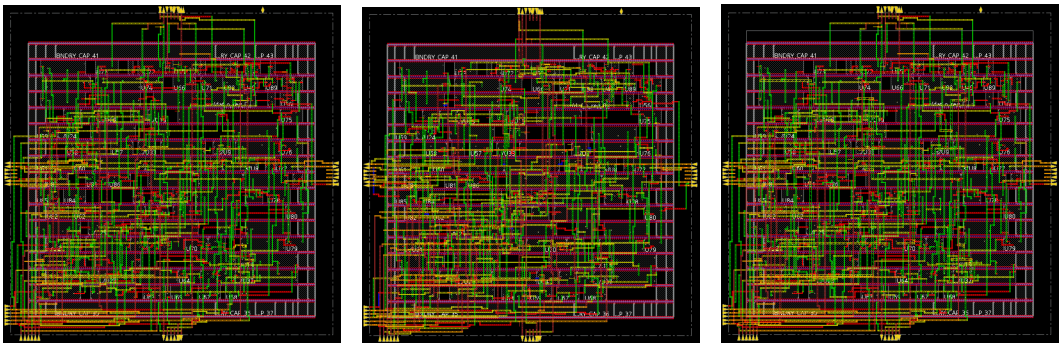



Figure 11: CPU time of iPCL and ChiP-STAR per held-out design, split into model inference (dark) and ECO (light), relative to the commercial router's CPU time on the same design (line at 1). The extra time of ChiP-STAR is inference; ECO time is comparable.

### F.5 Additional Case Study: `apb4_rng`

Figure 12 shows a second case study on `apb4_rng`, an unseen design with lower net density than `s5378`. The reduced density limits cross-net interaction, so ChiP-STAR already produces few DRC violations before ECO and the post-ECO layout (b) closely matches both the commercial result (c) and the ground truth (a) at the global routing level—confirming that the geometric patterns learned by ChiP-STAR generalize to lower-complexity designs as well.



(a) Ground truth (b) CHIP-STAR after ECO (c) Commercial router ECO

Figure 12: Layout comparison for `apb4_rng`. CHIP-STAR (center) produces routing comparable to the commercial tool result (right) and geometrically consistent with the ground truth (left).

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state three main claims: (i) identifying the *Three Spatial Blindnesses* as a conceptual framework for DFS-serialized routing; (ii) proposing three lightweight modules (STPE, GeoPrior, TrajNoise) that add under $0.5\%$ parameters; and (iii) achieving $\sim 5\%$ improvement on Leaf IoU and Connectivity over iPCL-Large, with $3\%$ shorter wirelength and $11\%$ fewer vias on commercial ECO signoff. These claims are directly supported by the analysis in Section 1 and Section 2, the method design in Section 3, the main results in Table 1, the ablation in Table 2, and the layout-level evaluation in Figures 5 and 6.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 discusses four limitations: (i) the backbone is fixed at $0.4$B parameters due to limited pre-training data, leaving scaling behavior open; (ii) ChiP-STAR does not produce DRC-clean layouts on its own and relies on commercial ECO; (iii) nets are routed independently, so multi-net effects such as congestion and cross-net timing are not captured; and (iv) vias are encoded only with a coarse B/T direction tag, omitting type and stack attributes.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper states one formal result, Theorem 3.1 (Geometric Invariance of the SoPE Component). The assumptions (the routing Steiner tree $T$, the coordinate function $c_\tau$ defined in Def. A.3, and the set of valid DFS traversals $\mathcal{S}(T)$) are explicitly stated in Section 2 and in the theorem statement. The complete proof is provided in the appendix, with an intuitive sketch given in Section 3.2 and empirical verification in Figure 9.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4 specifies the process node (28 nm), synthesis tool (Synopsys Design Compiler), back-end tool (Cadence Innovus), training hardware (16 NVIDIA Tesla V100 GPUs, 32 GB each), dataset construction (32 designs, $\sim 6.4$M nets, 556 GB), backbone (T5Gemma, hidden size 1024, 12 layers, 16 heads, 400M parameters), and key hyperparameters (GeoPrior $F$=16, $r$=8; $\lambda_{xy}$=$3\times10^{-5}$, $\lambda_z$=1.0; TrajNoise $\sigma_{xy}$=5.0, $\sigma_z$=2.0). Appendix E provides the full training configuration, and Appendices A.1, A.3, and A.2 document the dataset construction pipeline.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We release the code, model checkpoints, and dataset construction scripts at `https://github.com/jfkey/iPCL-R`. The repository includes instructions for environment setup, data preparation from AiEDA, training commands, and evaluation scripts that reproduce the main results in Table 1 and Table 2.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 specifies the train/test split (32 training designs and 10 held-out designs for downstream evaluation), the backbone configuration (T5Gemma, hidden size 1024, 12 layers, 16 heads, 400M parameters), and the key hyperparameters (GeoPrior $F$=16, $r$=8; coordinate scales $\lambda_{xy}$=3×10$^{-5}$, $\lambda_z$=1.0; TrajNoise $\sigma_{xy}$=5.0, $\sigma_z$=2.0). The downstream evaluation protocol is described in Section 4, including DRC cleaning by

Cadence Innovus ECO and PPA reporting. Full optimizer, learning-rate schedule, batch size, and training-budget details are provided in Appendix E.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Due to the substantial computational cost of pre-training a 400M-parameter encoder–decoder on a 1.8B-token routing corpus across 16 V100 GPUs, the main results in Table 1 and the ablations in Table 2 are reported from a single training run. However, each reported metric is averaged over many nets or designs: Table 1 over the 9,462 routable nets of the validation designs (Table 8), and Figures 5 and 6 over the 10 held-out designs, which provides reasonable estimation stability. We quantify the variance from training stochasticity on the Medium model: iPCL-Medium, CHIP-STAR-Medium without STPE and the full CHIP-STAR-Medium are each retrained with three random seeds for 5 epochs (Appendix F.1, Table 10). The error bars there are one sample standard deviation over seeds, at most $0.004$ Leaf IoU, against a gap of $0.026$ between the full model and iPCL-Medium. The main results in Table 1 carry no error bars.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Section 4 states that training is conducted on 16 NVIDIA Tesla V100 GPUs (32 GB each). The dataset is 556 GB in size. Additional details on training time and memory consumption are provided in Appendix E.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work studies VLSI routing using publicly available open-source EDA infrastructure (AiEDA, iPCL) and standard process design kits. No human subjects, personal data, or sensitive information are involved. The research conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This work focuses on improving routing quality in VLSI physical design using machine learning, an established engineering subroutine in chip design flows. It does not involve human subjects, sensitive data, or applications with foreseeable direct negative societal impact such as surveillance or autonomous weapons. Improvements in chip design efficiency benefit the broader semiconductor and computing ecosystem.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks,

mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The released assets (the geometry-annotated routing corpus and model weights) are specific to VLSI physical-design routing and pose no meaningful risk of misuse beyond standard EDA research. The model produces routing topologies that still require commercial signoff tools to be physically realized.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The existing assets are properly cited: AiEDA [36] and iPCL [13, 17] for the dataset and baseline; T5Gemma [51] and Llama [44] as the backbone and baseline architectures.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

Justification: The paper releases the largest geometry-annotated routing corpus to date, rebuilt from AiEDA and iPCL: 6.4M nets, 1.8B tokens, 556 GB across 32 designs with per-net spatial coordinate annotations. Corpus construction is described in Section 2 and Section 4, with full statistics, the routing-primitive abstraction, the Steiner-tree pre-processing, and the DecimalWordLevel tokenization documented in Appendices A.1, A.3, and A.2. The released assets are publicly available at `https://github.com/jfkey/iPCL-R`.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: This paper involves no crowdsourcing or human subjects. All data are routed netlists obtained from publicly available EDA design benchmarks and processed through commercial signoff tools.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: No human subjects research is conducted in this work.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs were not used as a core methodological component; any use was limited to writing assistance only.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.