APEC-Route: Guiding Analog Routing with On-the-Fly Parasitic Extraction Based on a Neural Capacitance Model

HAOYI ZHANG*, School of Integrated Circuits, Peking University, China
JIECHEN HUANG*, Department of Computer Science and Technology, BNRist, Tsinghua University, China
BINGYANG LIU, School of Electronics Engineering and Computer Science, Peking University, China
WENJIAN YU, Department of Computer Science and Technology, BNRist, Tsinghua University, China
YIBO LIN, School of Integrated Circuits, Peking University, China
RUNSHENG WANG, School of Integrated Circuits, Peking University, China

Analog routing must balance design rules and layout constraints while managing interconnect resistance and capacitance. Many analog-routing frameworks combine wirelength objectives with constraint handling and evaluate detailed parasitic effects after routing. This separation motivates investigating whether low-latency parasitic estimates can guide candidate expansions during routing. We present APEC-Route, an analog-routing framework that combines interval-tree-based wire segmentation, a multilayer perceptron (MLP)-based neighbor-aware total-capacitance model, and analytical wire-resistance evaluation. The resulting estimates guide the search in parasitic-minimization and parasitic-matching modes. On four finalized 65 nm layouts, the extractor attains a geometric-mean relative error of 5.54–6.77% and a mean absolute percentage error of 7.41–9.93% against the field-solver reference; each in-search query takes 251–345 μs. For each of three 65 nm block-level routed-circuit cases—an operational transconductance amplifier (OTA), a comparator, and a capacitive digital-to-analog converter (CDAC)—SAGERoute and the w/o-R/C, R-only, CC-only, and full APEC-Route variants use the same fixed placement. Across both controlled capacitance-term comparisons, enabling the capacitance term improves eight of nine independent circuit metrics, with OTA phase margin as the sole exception. The derived ENOB increases consistently with SINAD. Relative to w/o R/C, the full objective increases OTA gain and CMRR by 2.7 and 1.0 dB and UGB by 0.01 MHz, reduces comparator delay, hysteresis voltage, and offset voltage by 20.5%, 16.0%, and 26.6%, and reduces CDAC delay by 10.4% while increasing SINAD by 16.4 dB; OTA phase margin decreases by 1.1°.

CCS Concepts: • **Hardware → Electronic design automation**; *Analog and mixed-signal circuits.*

Additional Key Words and Phrases: parasitic extraction, neural capacitance model, parasitic-aware analog routing

1 Introduction

Analog routing is a time-consuming process that must balance design rules, symmetry, wire width, electromigration, and other constraints while managing interconnect resistance and capacitance, all of which affect circuit performance [25]. Representative analog-layout frameworks include ALIGN [10], MAGICAL [5, 6], SAGERoute [42, 43], BAG [9], and LAYGEN [23, 24]. Many such approaches combine a wirelength objective with constraint-driven heuristics. When interconnect parasitics are not represented directly in the routing objective, their electrical effects are assessed after routing and may require further design iteration.

*Haoyi Zhang and Jiechen Huang contributed equally to this research.

Authors' Contact Information: Haoyi Zhang, School of Integrated Circuits, Peking University, Beijing, China; Jiechen Huang, Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing, China; Bingyang Liu, School of Electronics Engineering and Computer Science, Peking University, Beijing, China; Wenjian Yu, yu-wj@tsinghua.edu.cn, Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing, China; Yibo Lin, yibolin@pku.edu.cn, School of Integrated Circuits, Peking University, Beijing, China; Runsheng Wang, School of Integrated Circuits, Peking University, Beijing, China
For correspondence, please contact Wenjian Yu (yu-wj@tsinghua.edu.cn) or Yibo Lin (yibolin@pku.edu.cn).

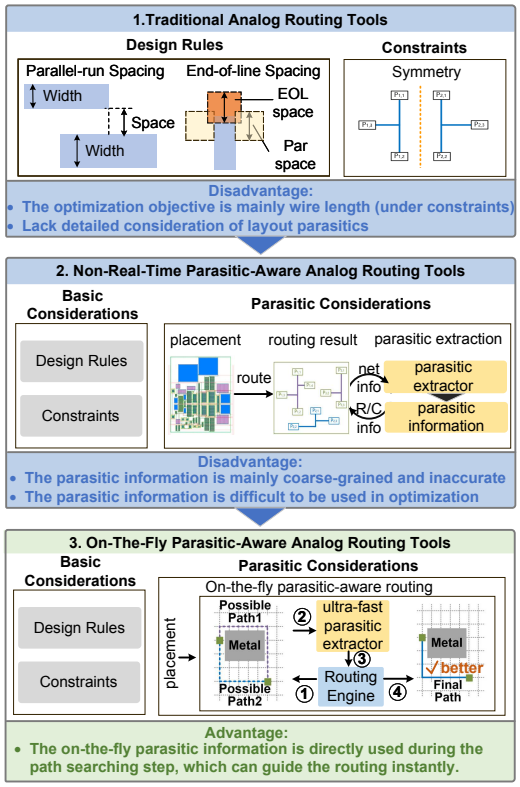

Fig. 1. Comparison of APEC-Route (3) with two representative analog-routing workflows (1 and 2).

Learning-guided routers use neural models in different ways. GeniusRoute [46] trains a variational autoencoder on manual layouts to generate routing-region guidance that biases detailed routing, whereas other learning-based approaches [22, 30, 35] predict parasitic or performance-related guidance before routing. Other works [3, 31, 44, 45] attempt to guide layout generation with parasitic information computed from approximate equations or field solvers. In addition, solutions such as LayoutCopilot [11, 21] and IAPARoute [7] incorporate human intervention to better account for interconnect parasitics. Most prior parasitic-aware approaches obtain parasitic information after an initial route or predict it before routing; they do not reevaluate the local conductor environment for every candidate expansion. Consequently, pre-routing predictions may not reflect the final local geometry, and once a routing solution is finalized, the design space for rip-up and re-routing becomes limited. IAPARoute [7] computes real-time estimates from approximate equations but does not explicitly reevaluate neighbor-induced capacitance for each candidate geometry. This limitation motivates investigating whether low-latency estimates that include the nearby conductor environment can provide additional guidance during routing.

The neural components of GeniusRoute and APEC-Route serve different purposes. GeniusRoute learns a pre-routing probability map from manually routed layouts and uses that map to bias a guidance-aware detailed router. In contrast, APEC-Route trains a field-solver-calibrated capacitance model on synthetic conductor cross sections and queries it on candidate geometries during routing. Consequently, the $(\lambda_R, \lambda_C) = (0, 0)$ configuration evaluated in Section 4 is an ablation of APEC-Route's own base router, not a reproduction of GeniusRoute.

On-the-fly parasitic extraction requires an ultra-fast extractor, which is particularly challenging for capacitance. Interconnect resistance can be modeled as proportional to wire length and inversely proportional to cross-sectional area [40], making it straightforward to evaluate during routing. Parasitic capacitance, however, is a coupling phenomenon strongly influenced by the surrounding conductors and dielectrics. Two principal capacitance-extraction approaches are three-dimensional (3D) field solvers and pattern matching. Accurate field solvers [13, 14, 26, 41] are computationally expensive and therefore impractical for frequent in-search queries. Pattern matching [8, 39] divides interconnects into segments whose capacitances are estimated from precharacterized cross-sectional patterns. It offers two advantages for this setting: (1) low-latency estimates whose accuracy can be evaluated before deployment, and (2) local, incremental evaluation without repeatedly solving a global field problem. Its main engineering challenge is that pattern libraries and matching rules are empirical, require extensive characterization and tuning, and are often proprietary. A compact and accurate learned model can replace part of this handcrafted estimation process.

In this work, we propose APEC-Route, an analog routing tool that exposes on-the-fly parasitic estimates to the routing search. Its ultra-fast extractor combines interval-tree-based wire segmentation with a multilayer perceptron (MLP) that estimates the capacitance of the resulting wire slices. Combined with analytically evaluated resistance, the extractor supplies candidate-path parasitic estimates during search. We use these estimates in two modes: parasitic minimization and parasitic matching. We evaluate the resulting routing behavior and circuit metrics on the block-level cases described in Section 4, subject to the comparison limitations stated below. The main contributions of this paper can be summarized as follows:

- Inspired by pattern-matching extraction, we propose an ultra-fast parasitic extractor that uses interval-tree-based segmentation to partition routed wires into slices and an MLP to estimate the neighbor-aware total capacitance of each slice.
- We propose a parasitic-aware algorithm that incorporates on-the-fly resistance and capacitance estimates in two modes: parasitic minimization and parasitic matching.
- On four finalized layouts, the extractor attains a geometric-mean relative error of 5.54–6.77% against a 3D field solver and runs faster than the evaluated commercial pattern-matching flow. On three routed-circuit cases, SAGERoute and the four evaluated APEC-Route configurations use the same fixed placement for each benchmark. In both controlled capacitance-term contrasts, enabling the capacitance term improves eight of nine independent circuit metrics, with OTA phase margin as the only exception; the derived ENOB increases consistently with SINAD, and the resistance term produces smaller, benchmark-dependent changes.

The empirical scope remains limited to the evaluated 65 nm block-level circuits and does not establish broader applicability to other technologies or larger irregular blocks.

2 Preliminaries

2.1 Grid-Based Analog Routing

Grid-based routing is widely used in analog and mixed-signal design [4, 16, 17, 29]. As illustrated in Figure 2, it discretizes the layout into a three-dimensional routing graph whose nodes can have up to six neighbors: four in the same layer and one in each adjacent layer. Analog routers can also employ nonuniform grids or permit wire placement beyond the fixed tracks used in digital routing [28]. A* is a widely used heuristic for searching this graph. It ranks candidate nodes by combining the accumulated path cost from the source (COSTG in Figure 2) with a heuristic estimate of the remaining cost to the target (COSTF), and these terms can be extended to represent analog-routing objectives and constraints.

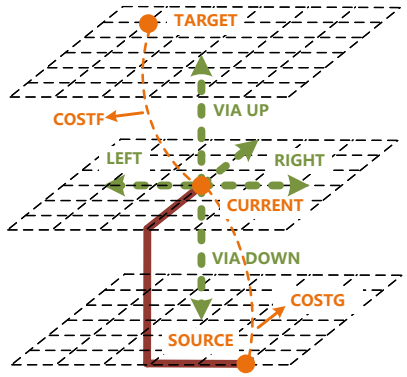

Fig. 2. Illustration of 3D grid-based routing.

Given a placement PL, a constraint set $CT = \{ct_i \mid 1 \le i \le |CT|\}$, and routing objectives $OB = \{OB_i \mid 1 \le i \le |OB|\}$, the analog-routing problem is to construct routing solutions $RS = \{RS_i \mid 1 \le i \le |RS|\}$ that satisfy every $ct_i \in CT$ while optimizing the objectives in OB.

2.2 Layout Parasitic Extraction

Layout parasitics include the resistance (R) and capacitance (C) of interconnect wires, which affect signal delay and other performance metrics [39]. Conventionally, parasitic extraction is performed after placement and routing to support post-layout simulation and analysis, thereby guiding design iteration. Interconnect resistance can be estimated from its length L, width W, and thickness T:

$$R = \frac{\rho L}{WT} \propto \frac{L}{W}, \tag{1}$$

which makes resistance extraction relatively straightforward [40]. In APEC-Route, resistance is evaluated with this analytical expression and is not predicted by a neural network; only capacitance is estimated by the MLP.

Parasitic capacitance arises from complex coupling among the surrounding conductors and dielectrics. For a system of conductors $\{i \mid 1 \le i \le N\}$, the *capacitance matrix* $\mathbf{C} \in \mathbb{R}^{N \times N}$ relates the electric charge vector $\mathbf{Q}$ and voltage vector $\mathbf{V}$ via

$$\mathbf{Q} = \mathbf{C}\mathbf{V}, \tag{2}$$

where each diagonal entry C_{ii} denotes total or self capacitance and, for $i \neq j$, $-C_{ij}$ denotes coupling capacitance. If C_{i0} denotes the capacitance from conductor i to the reference conductor or substrate, then

$$C_{ii} = C_{i0} - \sum_{1 \le j \le N,\, j \neq i} C_{ij}, \quad \forall\, 1 \le i \le N. \tag{3}$$

The total capacitance C_{ii} is the principal quantity of interest in many applications. Some studies [7, 42] approximate it as proportional to wire length and width:

$$\hat{C}_{ii} \propto LW, \tag{4}$$

which is a rough approximation of wire-to-substrate capacitance that ignores coupling effects.

2.3 Capacitance Extraction Techniques

Capacitance extractors fall into two broad categories: 3D field solvers and pattern-matching extractors. A 3D field solver computes the electric charge vector $\mathbf{Q}$ for an applied voltage vector

$\mathbf{V}$, from which the capacitance matrix $\mathbf{C}$ is obtained [40]. This computation is formulated as a boundary-value problem governed by Laplace's equation:

$$\begin{cases} \nabla \cdot (\varepsilon \nabla \phi) = 0, & \text{in } \Omega, \\ \phi = \phi_0, & \text{on } \partial\Omega_D, \\ \frac{\partial \phi}{\partial \mathbf{n}} = 0, & \text{on } \partial\Omega_N, \end{cases} \tag{5}$$

where ε is the dielectric permittivity, ϕ is the electric potential, and $\partial\Omega_D$ and $\partial\Omega_N$ are the Dirichlet and Neumann boundaries, respectively. Numerical methods for solving Equation (5) include the boundary element method (BEM) [26] and the floating-random-walk (FRW) method [13, 14, 41]. These methods can provide high accuracy but are usually computationally expensive.

In the pattern-matching approach, a set of cross-sectional patterns is defined for a process technology, and their two-dimensional capacitances are computed with a field solver to form a lookup table or polynomial-fitting rules [8, 39]. For an actual layout, wires are segmented so that each segment's surrounding geometry is invariant along its longitudinal direction. The 3D capacitances are then approximated from 2D cross-sectional capacitances by matching or interpolating predefined patterns. Commercial pattern-matching tools provide scalable performance even for chip-scale extraction, but their pattern libraries and matching rules are empirical and often proprietary. These limitations have motivated data-driven models of 2D cross-sectional capacitance [1, 2, 18–20, 36, 38]. To our knowledge, many reported models use a fixed metal-layer combination, often three selected layers, and have not been evaluated as part of a complete pattern-matching workflow on finalized layouts.

3 APEC-Route Framework

Figure 3 presents an overview of APEC-Route. The framework takes technology files, a circuit netlist, placement information, and a trained MLP-based capacitance model as input. The technology files include the layer map and the design-rule-checking (DRC) and layout-versus-schematic (LVS) rules. The capacitance model is trained on a layout-independent synthetic dataset labeled by a field solver (see Sec. 3.2). Once model training converges, the routing procedure requires no further fine-tuning for designs in the same technology and modeled geometry range.

APEC-Route integrates on-the-fly parasitic information into single-net routing. For a particular net, it considers wirelength, routing constraints, and parasitics. Resistance is evaluated analytically using Equation (1) and does not use machine learning; only capacitance is predicted by the MLP. For on-the-fly *capacitance extraction*, APEC-Route adopts the pattern-matching paradigm. It first partitions the wires into segments using the interval-tree-based method in Sec. 3.1, and then estimates the capacitance of those segments using the trained MLP in Sec. 3.2. The parasitic-aware algorithm in Sec. 3.3 uses these on-the-fly resistance and capacitance estimates during routing. APEC-Route applies the full process to each user-specified critical net; other nets are routed using basic wirelength and constraint considerations to balance runtime and quality.

3.1 Interval-Tree-Based Wire Segmentation

Figure 4 illustrates the interval-tree-based wire-segmentation method used by the capacitance extractor. For a specified master conductor, we first identify the relevant surrounding conductors. We locate the nearest non-empty layers above and below and search for conductors within a horizontal window of length $L_{\text{win}} = 40 \max_{\ell \in \mathcal{L}} \{W_{\text{min},\ell}\}$, where $\mathcal{L}$ is the set of routing layers and $W_{\text{min},\ell}$ is the minimum wire width on layer ℓ. We then partition the master conductor wherever its surrounding geometry changes, so that each resulting segment has a constant cross section.

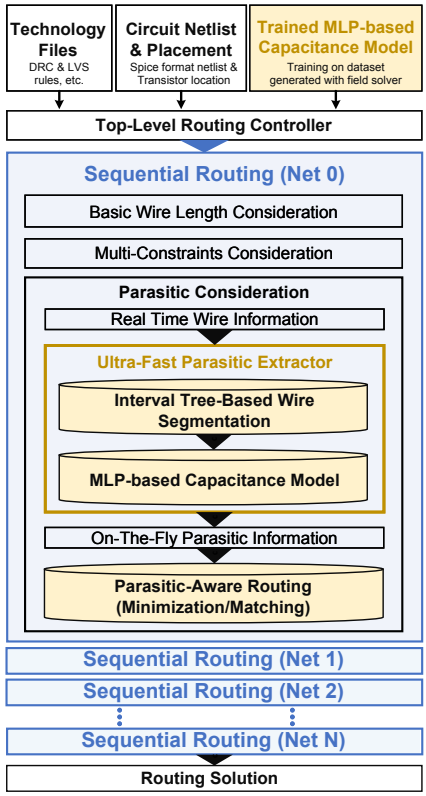

Fig. 3. Overview of APEC-Route.

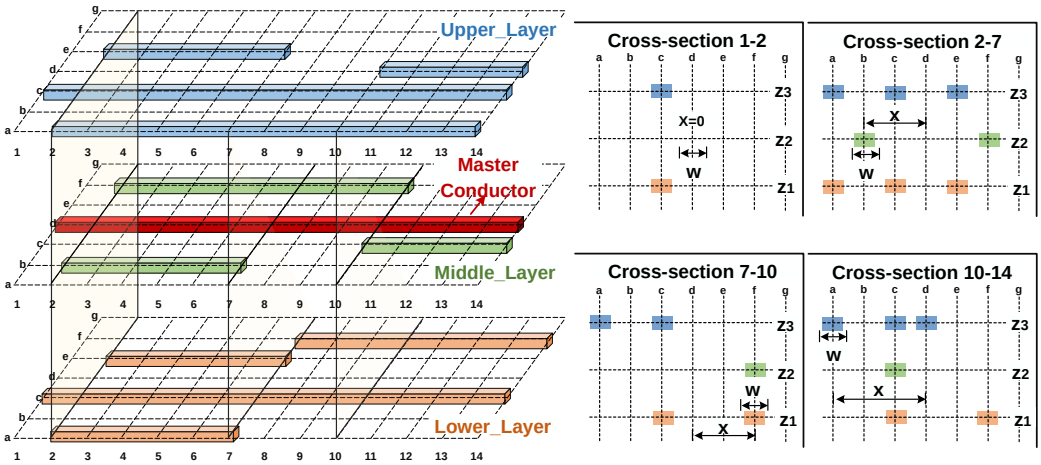

Fig. 4. Illustration of interval-tree-based wire segmentation and the resulting cross sections.

In Figure 4, the master conductor is divided into four segments (1–2, 2–7, 7–10, and 10–14); the corresponding cross sections appear on the right.

Within each cross section, the model considers up to nine conductors: the master conductor, its immediate left and right neighbors, and the three nearest conductors in each of the upper and lower layers. Conductors beyond this scope are omitted under the local-neighborhood approximation. Section 4 evaluates the resulting approximation on finalized layouts. This is a multi-conductor *capacitance-estimation* model: it estimates the total capacitance of one master conductor in the presence of nearby conductors. It should not be interpreted as a capability to synthesize multiple parallel wires for one electrical net. Each conductor is characterized by three parameters (W_i, X_i, Z_i): its width W_i, the horizontal distance X_i from its center to the master conductor's center, and its layer index Z_i. The resulting cross section is represented by 27 parameters and supplied to the MLP capacitance model. The capacitance of the master conductor is estimated by multiplying each two-dimensional cross-sectional capacitance by its segment length and summing over all segments.

The detailed implementation is shown in Algorithm 1. The algorithm takes the environment conductor boxes, the master conductor box, and the check length as input. It first initializes an interval tree based on the master conductor (line 2). Then, it projects the environment conductor boxes onto the master conductor box to find the clip points (line 3). Each clip point on the master conductor is inserted into the interval tree (lines 4–6). Next, the algorithm builds an R-tree based on the environment conductor boxes (line 7) for fast intersection checks. Then, each master conductor interval I is expanded with a check length L_{win} (line 9). The R-tree returns the conductor boxes that overlap the check box (line 10), from which the algorithm constructs the cross section (line 11). Finally, it merges intervals with the same cross section and returns the wire segments and cross sections (lines 13–15).

APEC-Route uses interval-tree-based segmentation to approximate a three-dimensional interconnect as a set of two-dimensional cross-sectional extraction problems. Section 4 evaluates the resulting accuracy and runtime against the field-solver reference.

Algorithm 1 Interval-Tree-Based Wire Segmentation

Require: Environment conductor boxes EnvBoxes, master conductor box M, check length L_{win}
Ensure: Wire segments WireSegs, cross-sections CrossSecs

```
1:  function WireSegmentation(EnvBoxes, M, L_win)
2:      ITree ← IntervalTreeInit(M)
3:      ClipPts ← ProjectBoxes(EnvBoxes, M)
4:      for all points p ∈ ClipPts do
5:          IntervalTreeInsert(ITree, p)
6:      end for
7:      RTree ← RTreeInit(EnvBoxes)
8:      for all intervals I ∈ ITree do
9:          CheckBox ← ExpandBox(I, L_win)
10:         Nbrs ← OverlapQuery(RTree, CheckBox)
11:         CrossSecs[I] ← ComputeCrossSec(M, Nbrs)
12:     end for
13:     ITree ← MergeIntervals(ITree, CrossSecs)
14:     WireSegs ← ExtractSegments(ITree)
15:     return (WireSegs, CrossSecs)
16: end function
```

3.2 MLP-Based Capacitance Model

Mapping cross-sectional geometry to capacitance is a key step in pattern-matching extraction and is often performed using a lookup table (LUT) [1, 2]. LUT interpolation can require extensive characterization because of finite table resolution. We instead develop a data-driven mapping from cross-sectional geometry to 2D capacitance; segment-level estimates are aggregated to estimate the capacitance of the original 3D conductor. Although many neural-network architectures are available, including convolutional neural networks (CNNs) [12], graph neural networks (GNNs) [33], and transformers [34], we choose a compact multilayer perceptron (MLP) for efficient function approximation at inference time. Section 4 evaluates its accuracy and runtime.

Figure 5 outlines the model. Because wire segmentation can produce tens of thousands of cross sections, evaluating every one with a field solver would be too slow for in-search use. We therefore generate synthetic structures with randomly placed conductors within the modeled layer, conductor-count, and geometry ranges, label them with a two-dimensional field solver, and train a compact MLP for pattern-matching extraction.

Prior work trains models using either samples from real circuit layouts [1, 2] or synthetic cross sections with a fixed number of conductors [20, 36]. Our sampling process instead varies the conductor count and layer combination within the modeled geometry range. It is designed to cover diverse cross sections without depending on a specific circuit layout. The sampling process is as follows. For each sample, we uniformly select a combination of three layers (e.g., M1, M3, and M6) and place the master conductor at the center of the middle layer. To represent cases in which no conductor or routing layer exists above or below the master, we add two dummy (empty) layers as the topmost and bottommost layers. This construction yields $\binom{m+2}{3}$ layer combinations, where m is the number of routing layers. The master-conductor width is sampled from the exponential-like distribution described by Yang et al. [36, Sec. 3.4]. Each left or right conductor is independently absent with probability 50%. If present, it is placed at a distance $d \sim \mathrm{Exp}(\frac{1}{4W_{\min}})$ from the master, where $W_{\min}$ is the minimum wire width of the middle layer. Its width is sampled uniformly within the design-rule-compliant range. For each upper or lower layer, the conductor count is sampled uniformly from $\{1, 2, 3\}$, and positions are sampled uniformly within the design-rule-compliant range. The horizontal window size is $40 \max_{\ell \in \mathcal{L}}\{W_{\min,\ell}\}$, for which the coupling capacitance of a conductor at the window boundary does not exceed 5% of the master conductor's total capacitance. We compute the reference capacitance of each synthetic structure using a 2D field solver.

The MLP input represents the conductor geometry of a cross-sectional pattern using 27 values, $\{(W_i, X_i, Z_i) \mid 1 \le i \le 9\}$ (see Sec. 3.1); absent conductors are zero-padded. The output is the total cross-sectional capacitance of the master conductor, C_{total} (i.e., C_{ii} in Equation (3)), in units of fF/μm. It includes the influence of the modeled neighboring conductors but does not separately predict every pairwise matrix entry C_{ij}. Figure 5 shows three hidden layers with 128, 64, and 32 neurons and Leaky ReLU activations. Once training converges, routing a design in the same technology and modeled geometry range requires no further fine-tuning. For a different technology, the synthetic dataset and the MLP model must be regenerated and retrained, as the specific design rules and dielectric stack fundamentally determine the conductor distributions and capacitance values. This technology-porting flow is a one-time procedure: generate rule-compliant synthetic cross-sections, label them with the 2D field solver, and retrain the MLP; no circuit-specific layout data or fine-tuning is required.

3.3 On-the-Fly Parasitic-Aware Analog Routing Algorithm

The on-the-fly parasitic-aware routing algorithm incorporates parasitic information into the analog routing process. APEC-Route applies parasitic-aware routing only to critical nets identified by the

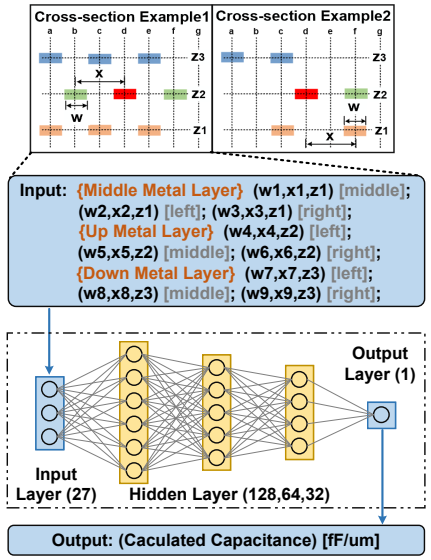

Fig. 5. The MLP for cross-sectional capacitance extraction.

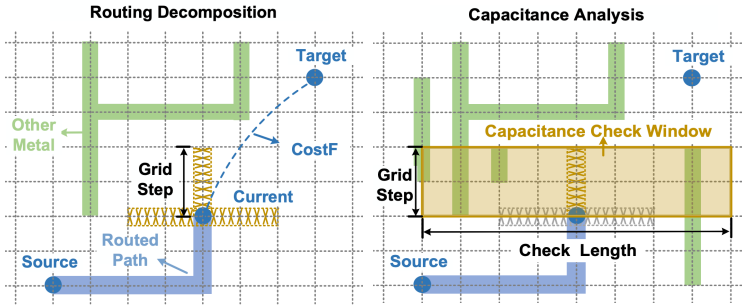

Fig. 6. Illustration of the on-the-fly capacitance extraction.

analog designer; other nets are routed using basic wirelength and constraint considerations to balance runtime and quality. These basic considerations are implemented using methods similar to SAGERoute [42]. For each net processed by the parasitic-aware routing kernel, APEC-Route adds a soft obstacle around the net to reduce capacitive interaction with other nets. The protection distance is defined by the user-specified routing grid and is usually two to three times the minimum design-rule width. The soft obstacle is then included in the routing graph. This soft obstacle is a common routing technique for mitigating design-rule violations and congestion. The associated penalty guides the routing process away from sensitive regions. The current router represents each electrical net as one routed tree and decomposes a multi-pin net into two-pin subnets. It does not jointly synthesize parallel conductors, current-sharing bundles, or redundant multi-wire structures within one net, and it does not provide complete electromigration (EM) or IR-drop closure for such structures.

Figure 6 illustrates the on-the-fly parasitic-aware routing algorithm. Parasitic information is incorporated directly into the fine-grained routing process. In the *Routing Decomposition* panel,

the current node has several candidate expansions during path search. Given the user-defined grid step size, candidate wire segments connect the current node to its neighboring nodes. For each candidate, APEC-Route estimates the parasitic capacitance inside the check window shown in the *Capacitance Analysis* panel using interval-tree-based wire segmentation and the MLP-based capacitance model. Resistance is calculated analytically from wire geometry using Equation (1). The on-the-fly parasitic information is then incorporated into the A* cost function:

$$J(u) = g(u) + h(u) + \lambda_R R_{\text{path}}(u) + \lambda_C C_{\text{est}}(u) \tag{6}$$

Here, $g(u)$ and $h(u)$ are the conventional A* path and heuristic costs, respectively; $R_{\text{path}}(u)$ is the cumulative path resistance evaluated analytically with Equation (1); and $C_{\text{est}}(u)$ is the on-the-fly total-capacitance score associated with the candidate expansion. The coefficients λ_R and λ_C convert the resistance and capacitance values, expressed in Ω and fF, into routing-cost units. Their numerical values are therefore implementation-specific weights rather than a physical ratio. This notation separates the analytical resistance term from the conventional path cost. When all routed segments have an identical cross section, the analytical resistance term is proportional to wirelength and therefore does not, by itself, demonstrate an advantage beyond wirelength optimization. The present formulation sums wire-segment resistance and does not include a separate via-resistance term.

Table 5 reports four APEC-Route configurations: w/o R/C, R only, CC only, and full, which use $(\lambda_R, \lambda_C) = (0, 0)$, $(0.1, 0)$, $(0, 1)$, and $(0.1, 1)$, respectively. The w/o-R/C configuration uses the same APEC-Route engine with both auxiliary parasitic terms disabled; SAGERoute is included as a separate reference evaluated with the same placement. Here, "CC only" denotes enabling $\lambda_C C_{\text{est}}$, where C_{est} estimates the master conductor's neighbor-aware total capacitance C_{ii}; it does not denote a separately predicted pairwise capacitance-matrix entry C_{ij}.

Algorithm 2 describes the parasitic-aware routing algorithm. It takes a routing net with pins P, resistance and capacitance scaling factors λ_R and λ_C, a set of matching costs MC, and a matching tolerance mp as inputs. The scaling factors balance the parasitic terms in Equation (6); the experiments use ohms (Ω) for resistance and femtofarads (fF, 10^{-15} F) for capacitance. Setting either scaling factor to zero disables its corresponding parasitic term; setting both to zero reduces Equation (6) to the common base cost $g(u) + h(u)$. After the first net of a matched pair is routed, its cost establishes MC_i for each corresponding two-pin subnet. In the current implementation, MC_i and its mismatch measure refer to the weighted composite routing cost in Equation (6), not to resistance or capacitance alone. The parameter mp is the allowed absolute deviation from MC_i, so a smaller numerical value of mp denotes higher matching precision. There is no universal optimal value because the appropriate tolerance depends on the matching specification and congestion. In practice, the designer begins with a permissive tolerance and tightens it until the required match is achieved without an unacceptable loss of routability; reporting $mp/|MC_i|$ also provides a scale-independent relative tolerance. An overly strict value may eliminate every path satisfying the tolerance in a congested region. In that event, the current implementation falls back to minimization mode instead of terminating without a route.

For each multi-pin net, APEC-Route first partitions the net into two-pin subnets using a minimum spanning tree and invokes A* search for each subnet. The search maintains the standard best-score, predecessor, closed-set, and priority-queue bookkeeping and returns the traced path when the target is removed from the priority queue. For every legal candidate expansion, it sums the conventional A* cost, the analytical resistance component, and the MLP-estimated capacitance component. In matching mode, only candidates whose absolute cost difference from MC_i is within mp are inserted into the queue; in minimization mode, this mismatch filter is disabled. If the matching-mode queue is exhausted, the algorithm records the fallback, clears the matching target by setting $MC_i = \varnothing$,

Algorithm 2 Parasitic-Aware Routing

Require: Net n with pin set P, resistance scaling factor λ_R, capacitance scaling factor λ_C, a set of matching costs MC, and matching tolerance mp
Ensure: Routing solution RS_n for net n

function ParasiticAwareRouting($P, \lambda_R, \lambda_C, MC, mp$)
  TwoPinNets ← PartitionSubnets(P)
  for all $tn_i \in$ TwoPinNets **do**
    $RS_{n,i} \leftarrow$ A*Search($tn_i, \lambda_R, \lambda_C, MC_i, mp$)
  end for
  $RS_n \leftarrow \bigcup_i RS_{n,i}$
  return RS_n
end function
function A*Search($tn_i, \lambda_R, \lambda_C, MC_i, mp$)
  Initialize priority queue $Q \leftarrow \varnothing$, closed set $S \leftarrow \varnothing$
  Initialize best-score map $B[x] \leftarrow \infty$ and predecessor map π
  $v_{src} \leftarrow$ GetSource(tn_i)
  $v_{tgt} \leftarrow$ GetTarget(tn_i)
  $B[v_{src}] \leftarrow 0$; Q.push($v_{src}, 0$)
  while $Q \neq \varnothing$ **do**
    $v \leftarrow Q$.pop()
    if $v \in S$ **then**
      continue
    end if
    $S \leftarrow S \cup \{v\}$
    if $v = v_{tgt}$ **then**
      return TraceBack(π, v)
    end if
    for all $u \in$ LegalNeighbors(v) **do**
      cost ← PathCost(u) + HeuristicCost(u)
      cost ← cost + $\lambda_R \times$ PathResistance(u)
      cost ← cost + $\lambda_C \times$ CapExtraction(u)
      if $MC_i = \varnothing$ **then**
        mismatch ← 0
      else
        mismatch ← ABSDiff(cost, MC_i)
      end if
      if mismatch $\leq mp$ **and** cost $< B[u]$ **then**
        $B[u] \leftarrow$ cost; $\pi[u] \leftarrow v$
        PushOrDecrease(Q, u, cost)
      end if
    end for
  end while
  if $MC_i \neq \varnothing$ **then**
    RecordMatchingFallback(tn_i)
    return A*Search($tn_i, \lambda_R, \lambda_C, \varnothing, mp$)
  end if
  return $\varnothing$
end function

and restarts the search in minimization mode. If that unconstrained search also exhausts the queue, it reports that no legal route exists.

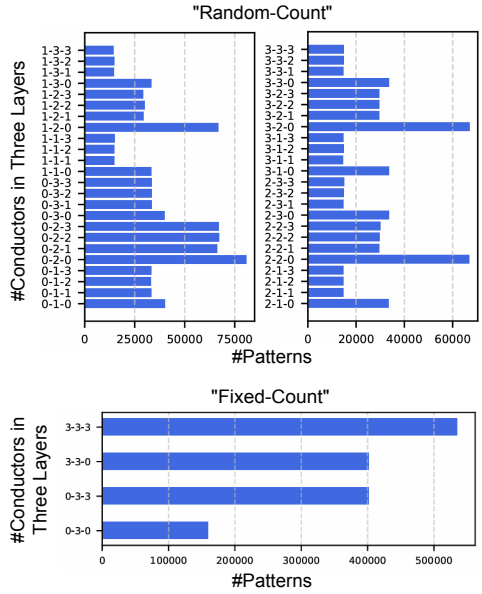

Fig. 7. Conductor-count distributions of the two datasets for the selected three layers.

4 Experimental Results

The wire-segmentation algorithm in Sec. 3.1 and the analog-routing engine in Sec. 3.3 are implemented in C++. The MLP described in Sec. 3.2 is trained with PyTorch on an NVIDIA RTX A6000 GPU, and inference is integrated into the routing engine through the PyTorch C++ API. Capacitance labels for the synthetic dataset are generated with an in-house two-dimensional floating-random-walk (FRW) solver. All experiments target a 65 nm process and run on a Linux server equipped with Intel Xeon Silver 4214 CPUs operating at 2.20 GHz.

4.1 Training the MLP-Based Capacitance Model

We generated two datasets to evaluate the effect of pattern diversity. The "Random-Count" dataset follows the sampling procedure in Sec. 3.2 and allows a variable number of conductors per pattern. The "Fixed-Count" dataset uses the same sampling distributions but fixes three conductors on every selected layer, following the setup used in prior work [20, 36]. Each dataset contains 1.5 million synthetic cross-sectional patterns. Figure 7 shows their conductor-count distributions.

The 2D FRW solver requires about 1 s to compute the reference capacitance for each cross-section pattern. For 1.5 million patterns per dataset and 12 parallel processes, the idealized arithmetic time is therefore about 35 hours per dataset. The end-to-end wall-clock labeling time, including data preparation, I/O, and scheduling overhead, was not recorded separately. Each dataset is divided into a 90% training set and a 10% test set. We train the MLP with mean-squared-error (MSE) loss and the Adam optimizer. The initial learning rate is 10^{-4} and is multiplied by 0.5 every 75 epochs; the batch size is 1024. For the selected 128–64–32 model trained on the Random-Count dataset, the 400-epoch training took 4033 s (about 67 minutes) on the RTX A6000. Dataset generation and model training are one-time costs for a given technology and geometry range, rather than per-routing-run costs.

Table 1. Evaluation of the trained MLP capacitance models on held-out synthetic test sets. MAPE is the mean absolute percentage error over the corresponding test set; ARE denotes absolute relative error, and Ratio is the percentage of test samples whose ARE exceeds 5%.

Dataset	Training Time	Test Loss	MAPE	Ratio (ARE>5%)
"Fixed-Count"	3867 s	5.6×10^{-6}	1.10%	0.86%
"Random-Count"	4033 s	5.9×10^{-6}	1.34%	1.52%

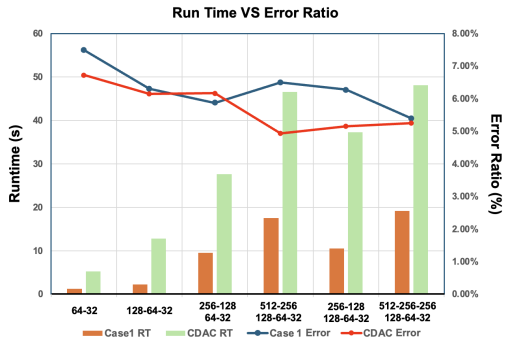

Fig. 8. Extraction-runtime and geometric-mean relative-error tradeoffs for the evaluated MLP architectures.

Table 1 reports held-out test-set results after 400 training epochs. Only total capacitance is evaluated. The Random-Count model has a slightly higher held-out error than the Fixed-Count model, consistent with the greater diversity of its training distribution.

4.2 Evaluation of the Neural Capacitance Extractor

To evaluate the accuracy–runtime tradeoff of the MLP-based capacitance extractor, we train six architectures: 64–32, 128–64–32, 256–128–64–32, 512–256–128–64–32, 256–128–128–64–32, and 512–256–256–128–64–32. We evaluate each model on the Case1 and CDAC layouts; Figure 8 reports the resulting extraction runtime and geometric-mean relative error. The larger models increase runtime substantially while providing only modest and non-monotonic accuracy changes. Because on-the-fly extraction is invoked repeatedly during routing, we select the 128–64–32 model as a favorable observed Pareto tradeoff between accuracy and runtime, rather than as the architecture with the minimum error alone.

The MLP-based pattern-matching model can also operate as a stand-alone capacitance extractor. We evaluate its accuracy on four 65 nm layouts. **Case1** contains 175 extracted nets from an industry-provided digital circuit layout, as shown in Figure 9. The OTA, COMP, and CDAC cases are finalized layouts produced by our routing engine (see Sec. 4.3). We use the 3D FRW-AGF field solver [15, 27] to compute reference values and also evaluate the commercial LUT-based pattern-matching tool StarRC [32].

Tables 2 and 3 report the relative errors and runtimes of StarRC, CNN-Cap [37], and our extractor with MLPs trained on the two datasets in Sec. 4.1. We train CNN-Cap on the Random-Count dataset using the settings in its original paper, except that the input is expanded to six channels for the six metal layers in our experiments. Training runs for 20 epochs; Table 2 evaluates the resulting model on the finalized layouts rather than on synthetic training patterns.

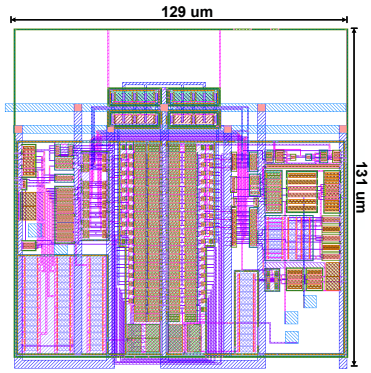

Fig. 9. Visualization of Case1.

Table 2. Total-capacitance extraction accuracy against 3D FRW-AGF. GME is the geometric mean of relative errors and MAPE is the mean absolute percentage error.

	StarRC		MLP–Random		MLP–Fixed		CNN–Random	
Case	GME	MAPE	GME	MAPE	GME	MAPE	GME	MAPE
Case1	4.46%	8.55%	6.31%	9.93%	301%	408%	28.6%	41.8%
OTA	1.92%	8.21%	5.54%	7.76%	708%	833%	43.98%	58.9%
COMP	3.35%	3.95%	6.77%	8.79%	501%	697%	54.9%	58.8%
CDAC	5.19%	5.19%	6.15%	7.41%	923%	946%	10.4%	11.2%

Table 3. Total-capacitance extraction runtime, including initialization and extraction of all nets. Boldface marks the shortest runtime in each row, including ties.

Case	StarRC	MLP–Random	MLP–Fixed	CNN–Random
Case1	29 s	**2.29 s**	2.36 s	98.6 s
OTA	31 s	**0.50 s**	0.54 s	21.3 s
COMP	26 s	**0.12 s**	**0.12 s**	1.8 s
CDAC	33 s	**12.8 s**	13.1 s	407.7 s

The runtimes in Table 3 include initialization and extraction of all nets. The Fixed-Count model generalizes poorly to the four finalized layouts, whereas Random-Count improves robustness within the evaluated geometry range. Here, MAPE $= \frac{100}{N}\sum_{i=1}^{N}\left|\left(\widehat{C}_i - C_i\right)/C_i\right|$ is the mean absolute percentage error, whereas the geometric-mean error (GME) is the geometric mean of the relative errors; the two summaries should not be conflated. For the Random-Count MLP, MAPE ranges from 7.41% to 9.93%, GME ranges from 5.54% to 6.77%, and the measured extraction runtimes are shorter than those of StarRC on all four cases. For CDAC, the measured MLP initialization time is 7.0 s. CNN-Cap shows higher real-layout errors and longer extraction runtimes on these cases. This gap may reflect its fixed-layer architectural assumptions and the variable three-of-six-layer patterns evaluated here.

The three evaluation levels in this section measure different quantities. Table 1 evaluates capacitance predictions on held-out synthetic patterns (Random-Count test loss 5.9×10^{-6}, MAPE 1.34%,

Table 4. Benchmark circuit statistics.

Benchmark	#Devices	Critical Nets / Total Nets	Die Size
Case1	–	– / 175	129 μm × 131 μm
OTA	17	4 / 17	50 μm × 56 μm
COMP	11	2 / 8	31 μm × 29 μm
CDAC	231	12 / 132	47 μm × 131 μm

and 1.52% of samples above 5% relative error). Tables 2 and 3 evaluate the complete segmentation-plus-MLP extraction flow on finalized real layouts against FRW-AGF. In contrast, the circuit metrics in Table 5 come from independent post-layout extraction of resistance, ground capacitance, and coupling capacitance (R+C+CC), followed by circuit simulation; they are not measurements of the MLP prediction error. The neural extractor is used only as an on-the-fly routing guide.

4.3 Circuit-Level Evaluation of APEC-Route

We integrate wire segmentation and the MLP-based capacitance model into the on-the-fly parasitic-aware routing engine and evaluate it on three routing benchmarks: OTA, COMP, and CDAC. Case1 is an industry-provided digital sub-block whose device-level details and function cannot be disclosed; it is used only to evaluate capacitance extraction. The OTA is a two-stage circuit with 17 devices; COMP is a comparator with 11 devices; and the CDAC benchmark is a 231-device capacitive digital-to-analog converter used in a successive-approximation-register analog-to-digital converter (SAR ADC). Table 4 summarizes the four extraction cases and three routing benchmarks. All use a 65 nm technology, and the largest reported routing-benchmark die size is 47 μm × 131 μm.

For each benchmark, all reported routing configurations use the same fixed placement.

Table 5 reports unity-gain bandwidth (UGB), common-mode rejection ratio (CMRR), phase margin (PM), hysteresis voltage (HV), offset voltage (OV), signal-to-noise-and-distortion ratio (SINAD), and effective number of bits (ENOB). The four APEC-Route columns are w/o R/C, R only, CC only, and full, corresponding to $(\lambda_R, \lambda_C) = (0, 0)$, $(0.1, 0)$, $(0, 1)$, and $(0.1, 1)$, respectively; SAGERoute is included as a separate reference evaluated with the same placement. Here, "CC only" denotes enabling $\lambda_C C_{\text{est}}$, where C_{est} is the learned estimate of the master conductor's neighbor-aware total capacitance C_{ii}; it is not a separately predicted pairwise capacitance-matrix entry C_{ij}. ENOB is calculated from SINAD as ENOB = (SINAD − 1.76)/6.02 and, as a deterministic transform of SINAD, is not counted as an independent circuit metric.

For each benchmark, the same fixed placement was supplied to SAGERoute and all four APEC-Route configurations: the OTA and COMP placements were obtained from manual layouts, whereas the CDAC placement was generated by a CDAC-specific template. The APEC-Route w/o-R/C, R-only, CC-only, and full-objective runs use $(\lambda_R, \lambda_C) = (0, 0)$, $(0.1, 0)$, $(0, 1)$, and $(0.1, 1)$, respectively. All four APEC-Route configurations use parasitic-minimization mode, i.e., $MC_i = \varnothing$, so the matching-tolerance filter is inactive. Apart from λ_R and λ_C, the four runs use the same APEC-Route engine and routing/evaluation setup. Capacitance values are expressed in fF, and path resistance is evaluated analytically with Equation (1). These coefficients scale the two parasitic cost terms rather than representing a physical ratio. Setting either coefficient to zero disables its corresponding term. The present implementation sums wire-segment resistance but does not include a separate via-resistance contribution.

Table 5. Routing runtimes and post-layout circuit metrics for SAGERoute and four APEC-Route configurations using the same fixed placement for each benchmark.

Benchmark		Schematic	SAGERoute [42, 43]	APEC-Route			
				w/o R/C	R only	CC only	Full
OTA	Gain (dB) ↑	53.1	52.1	52.0	53.4	53.9	54.7
	UGB (MHz) ↑	0.91	0.87	0.88	0.88	0.89	0.89
	CMRR (dB) ↑	50.1	48.7	48.8	49.4	48.9	49.8
	PM (°) ↑	69.0	72.1	72.0	71.4	71.1	70.9
	Runtime (s)	–	3.1	3.1	3.4	7.8	6.6
COMP	Delay (ns) ↓	5.8	7.9	7.8	7.7	6.3	6.2
	HV (mV) ↓	177	267	275	256	234	231
	OV (mV) ↓	28.6	40.6	43.2	38.4	32.1	31.7
	Runtime (s)	–	0.5	0.5	0.5	3.8	3.3
CDAC (SAR ADC)	Delay (ns) ↓	25.9	29.8	29.8	28.7	27.1	26.7
	SINAD (dB) ↑	65.6	46.9	45.8	44.3	57.6	62.2
	ENOB (bits) ↑	10.6	7.5	7.3	7.1	9.3	10.0
	Runtime (s)	–	52.4	50.9	60.5	230.8	160.9

APEC-Route applies the parasitic-aware objective to designer-identified critical nets and retains the design-rule and symmetry mechanisms used by the underlying analog router. Each routed layout was evaluated by post-layout circuit simulation following R+C+CC extraction. Because the same fixed placement is used for every routing column on each benchmark, placement does not confound the numerical differences in Table 5. Figure 10 shows the routed OTA and CDAC layouts.

The w/o-R/C, R-only, CC-only, and full entries in Table 5 represent all four combinations of the two auxiliary parasitic terms. Comparisons of R only against w/o R/C and full against CC only assess the conditional contribution of the resistance term; comparisons of CC only against w/o R/C and full against R only assess the conditional contribution of the learned neighbor-aware total-capacitance term. Across the latter two controlled capacitance contrasts, enabling this term improves eight of nine independent circuit metrics; OTA phase margin is the sole exception. The derived ENOB values increase consistently with SINAD. For COMP, CC only versus w/o R/C reduces delay, HV, and OV by 19.2%, 14.9%, and 25.7%, respectively, while full versus R only yields reductions of 19.5%, 9.8%, and 17.4%. For CDAC, the same two contrasts reduce delay by 9.1% and 7.0% and increase SINAD by 11.8 and 17.9 dB. For OTA, they increase gain by 1.9 and 1.3 dB, UGB by 0.01 MHz in both cases, and CMRR by 0.1 and 0.4 dB, while reducing phase margin by 0.9° and 0.5°.

The resistance term has a smaller and more benchmark-dependent effect. It improves all three COMP metrics in both conditional contrasts. On CDAC, however, R only versus w/o R/C reduces delay by 3.7% but also reduces SINAD by 1.5 dB; with the capacitance term enabled, adding resistance provides a further 1.5% delay reduction and a 4.6 dB SINAD increase. Relative to w/o R/C, the full objective improves eight of nine independent circuit metrics: OTA gain and CMRR increase by 2.7 and 1.0 dB, and UGB increases by 0.01 MHz; COMP delay, HV, and OV decrease by 20.5%, 16.0%, and 26.6%; and CDAC delay decreases by 10.4% while SINAD increases by 16.4 dB. The derived ENOB increases consistently with SINAD. The exception is OTA phase margin, which decreases by 1.1°. These controlled comparisons indicate that the learned capacitance term is the principal

Table 6. Runtime breakdown of the full-objective APEC-Route runs.

Benchmark	Pure Routing Time	Capacitance Query Time	#Queries	Time per Query
OTA	2.9 s	3.7 s	13482	274 μs
COMP	1.2 s	2.1 s	8351	251 μs
CDAC	73.7 s	87.2 s	252673	345 μs

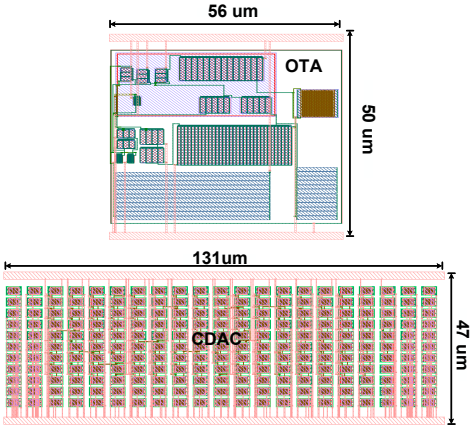

Fig. 10. The final routing solutions for OTA and CDAC.

source of the COMP and CDAC improvements, whereas the resistance term is secondary and is not uniformly beneficial.

For the full-objective APEC-Route configuration, the largest runtime is 160.9 s for CDAC, comprising 73.7 s of pure routing and 87.2 s of capacitance-query time in Table 6. This overhead reflects repeated on-the-fly extraction. The reported implementation targets block-level analog circuits; no runtime or quality claim is made here for larger designs.

Table 6 decomposes the APEC-Route runtime. The variation in pure routing time reflects both benchmark size and the search paths explored under parasitic guidance. The "#Queries" column reports the number of capacitance queries issued during routing. Each query takes a few hundred microseconds (251–345 μs), keeping the total runtime within the range reported for these block-level designs. The approximately 1 s per-cross-section reference-labeling time reported in Sec. 4.1 is more than three orders of magnitude longer than a neural query and is therefore unsuitable for repeated in-search evaluation. The aggregate query time in Table 6 exceeds the finalized-layout extraction time in Table 2 because step-by-step routing invokes the extractor many times.

5 Conclusion

This work presents APEC-Route, an on-the-fly parasitic-aware analog routing framework that combines interval-tree-based wire segmentation, an MLP capacitance model, and analytical wire-resistance evaluation. On four 65 nm real-layout extraction cases, the selected Random-Count model attains a geometric-mean relative error of 5.54–6.77% and a mean absolute percentage error (MAPE) of 7.41–9.93% against the field-solver reference, with substantially shorter extraction times

than the evaluated commercial pattern-matching flow. These results support the feasibility of using the lightweight extractor as an in-search routing guide.

SAGERoute and the w/o-R/C, R-only, CC-only, and full APEC-Route configurations use the same fixed placement for each benchmark. In both controlled capacitance-term contrasts, enabling the capacitance term improves eight of nine independent circuit metrics, with OTA phase margin as the sole tradeoff; the derived ENOB increases consistently with SINAD, and the resistance term produces smaller, benchmark-dependent changes. Relative to w/o R/C, the full objective reduces COMP delay, HV, and OV by 20.5%, 16.0%, and 26.6%, reduces CDAC delay by 10.4%, and increases CDAC SINAD by 16.4 dB, while OTA phase margin decreases by 1.1°. These results support attributing the principal COMP and CDAC improvements to the learned capacitance guidance rather than to an additional resistance-related wirelength weight alone.

The empirical scope is limited to the evaluated 65 nm block-level designs. The MLP predicts the master conductor's neighbor-aware total capacitance C_{ii} rather than the complete pairwise capacitance matrix. The current router represents each net as one routed tree, does not synthesize parallel current-sharing conductors, does not provide complete EM or IR-drop closure, and omits a separate via-resistance term. These limitations define the scope of the conclusions that can be drawn from the present study.

Acknowledgments

This work was supported in part by the Jiangsu National Science and Technology Major Project (Grant No. SBG20250000204), the MIND project (MINDXZ202406), and the Natural Science Foundation of Beijing, China (Grant No. Z230002).

References

[1] Mohamed Saleh Abouelyazid, Sherif Hammouda, and Yehea Ismail. 2022. Accuracy-based hybrid parasitic capacitance extraction using rule-based, neural-networks, and field-solver methods. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41, 12 (2022), 5681–5694.

[2] Mohamed Saleh Abouelyazid, Sherif Hammouda, and Yehea Ismail. 2022. Fast and accurate machine learning compact models for interconnect parasitic capacitances considering systematic process variations. *IEEE Access* 10 (2022), 7533–7553.

[3] S. Bhattacharya, N. Jangkrajarng, and C.-J.R. Shi. 2005. Template-driven parasitic-aware optimization of analog integrated circuit layouts. In *Proceedings of the 42nd Design Automation Conference*. ACM, 644–647. doi:10.1145/1065579.1065748

[4] Gengjie Chen, Chak-Wa Pui, Haocheng Li, and Evangeline F. Y. Young. 2020. Dr. CU: Detailed Routing by Sparse Grid Graph and Minimum-Area-Captured Path Search. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 9 (2020), 1902–1915. doi:10.1109/TCAD.2019.2927542

[5] Hao Chen, Mingjie Liu, Biying Xu, Keren Zhu, Xiyuan Tang, Shaolan Li, Yibo Lin, Nan Sun, and David Z. Pan. 2021. MAGICAL: An Open-Source Fully Automated Analog IC Layout System from Netlist to GDSII. *IEEE Design & Test* 38, 2 (2021), 19–26. doi:10.1109/MDAT.2020.3024153

[6] Hao Chen, Keren Zhu, Mingjie Liu, Xiyuan Tang, Nan Sun, and David Z. Pan. 2020. Toward silicon-proven detailed routing for analog and mixed-signal circuits. In *Proceedings of the 39th International Conference on Computer-Aided Design* (Virtual Event USA, 2020-11-02). ACM, 1–8. doi:10.1145/3400302.3415660

[7] Yu-Tzu Chen, Chin-Fu Nien, Chin Hsia, and Chung-Yi Li. 2024. Interactive Analog IC Layout Tool with Real-time Parasitic-aware Automatic Routing Assistance. In *2024 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*. 149–153. doi:10.1109/APCCAS62602.2024.10808823

[8] Jason Cong, Lei He, Andrew B. Kahng, David Noice, Nagesh Shirali, and Steve H.-C. Yen. 1997. Analysis and justification of a simple, practical 2 1/2-D capacitance extraction methodology. In *Proceedings of the 34th Annual Design Automation Conference*. 627–632.

[9] J. Crossley, A. Puggelli, H.-P. Le, B. Yang, R. Nancollas, K. Jung, L. Kong, N. Narevsky, Y. Lu, N. Sutardja, E. J. An, A. L. Sangiovanni-Vincentelli, and E. Alon. 2013. BAG: A designer-oriented integrated framework for the development of AMS circuit generators. In *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)* (2013-11). 74–81. doi:10.1109/ICCAD.2013.6691100 ISSN: 1558-2434.

[10] Tonmoy Dhar, Kishor Kunal, Yaguang Li, Yishuang Lin, Meghna Madhusudan, Jitesh Poojary, Arvind K. Sharma, Steven M. Burns, Ramesh Harjani, Jiang Hu, Parijat Mukherjee, Soner Yaldiz, and Sachin S. Sapatnekar. 2020. The ALIGN open-source analog layout generator: v1.0 and beyond. In *Proceedings of the 39th International Conference on Computer-Aided Design* (Virtual Event USA, 2020-11-02). ACM, 1–2. doi:10.1145/3400302.3415784

[11] Xiaohan Gao, Haoyi Zhang, Mingjie Liu, Linxiao Shen, David Z. Pan, Yibo Lin, Runsheng Wang, and Ru Huang. 2023. Interactive Analog Layout Editing With Instant Placement and Routing Legalization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 42, 3 (2023), 698–711. doi:10.1109/TCAD.2022.3190234

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385 [cs.CV] https://arxiv.org/abs/1512.03385

[13] Jiechen Huang, Shuailong Liu, and Wenjian Yu. 2025. A Parallel Floating Random Walk Solver for Reproducible and Reliable Capacitance Extraction. In *2025 Design, Automation & Test in Europe Conference (DATE)*. IEEE, 1–7. doi:10.23919/DATE64628.2025.10992848

[14] Jiechen Huang, Ming Yang, and Wenjian Yu. 2024. The Floating Random Walk Method With Symmetric Multiple-Shooting Walks for Capacitance Extraction. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* 43, 7 (2024), 2098–2111.

[15] Jiechen Huang and Wenjian Yu. 2024. Enhancing 3-D Random Walk Capacitance Solver with Analytic Surface Green's Functions of Transition Cubes. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*. ACM, 6 pages. doi:10.1145/3649329.3658263

[16] Andrew B. Kahng, Lutong Wang, and Bangqi Xu. 2018. TritonRoute: an initial detailed router for advanced VLSI technologies. In *Proceedings of the International Conference on Computer-Aided Design* (San Diego California, 2018-11-05). ACM, 1–8. doi:10.1145/3240765.3240766

[17] Haocheng Li, Gengjie Chen, Bentian Jiang, Jingsong Chen, and Evangeline F. Y. Young. 2019. Dr. CU 2.0: A Scalable Detailed Routing Framework with Correct-by-Construction Design Rule Satisfaction. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)* (Westminster, CO, USA, 2019-11). IEEE, 1–7. doi:10.1109/ICCAD45719.2019.8942074

[18] Haoyuan Li, Dingcheng Yang, and Wenjian Yu. 2024. Training Better CNN Models for 3-D Capacitance Extraction with Neural Architecture Search. In *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 1–2.

[19] Ping Li and Zhong Guan. 2025. Compact Geometric Feature Representation for Improved Capacitance Pattern-Matching in Parasitic Extraction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 44, 8 (2025), 3085–3098. doi:10.1109/TCAD.2025.3536380

[20] Zhixing Li and Weiping Shi. 2020. Layout capacitance extraction using automatic pre-characterization and machine learning. In *2020 21st International Symposium on Quality Electronic Design (ISQED)*. IEEE, 457–464.

[21] Bingyang Liu, Haoyi Zhang, Xiaohan Gao, Zichen Kong, Xiyuan Tang, Yibo Lin, Runsheng Wang, and Ru Huang. 2025. LayoutCopilot: An LLM-Powered Multi-Agent Collaborative Framework for Interactive Analog Layout Design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 44, 8 (2025), 3126–3139. doi:10.1109/TCAD.2025.3529805

[22] Mingjie Liu, Walker J. Turner, George F. Kokai, Brucek Khailany, David Z. Pan, and Haoxing Ren. 2021. Parasitic-Aware Analog Circuit Sizing with Graph Neural Networks and Bayesian Optimization. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)* (Grenoble, France, 2021-02-01). IEEE, 1372–1377. doi:10.23919/DATE51398.2021.9474253

[23] N. Lourenco, M. Vianello, J. Guilherme, and N. Horta. 2006. LAYGEN - Automatic Layout Generation of Analog ICs from Hierarchical Template Descriptions. In *2006 Ph.D. Research in Microelectronics and Electronics* (Otranto, Italy, 2006). IEEE, 213–216. doi:10.1109/RME.2006.1689934

[24] Ricardo Martins, Nuno Lourenço, and Nuno Horta. 2013. LAYGEN II—Automatic Layout Generation of Analog Integrated Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 32, 11 (2013), 1641–1654. doi:10.1109/TCAD.2013.2269050

[25] Ricardo M. F. Martins and Nuno C. C. Lourenço. 2023. Analog Integrated Circuit Routing Techniques: An Extensive Review. *IEEE Access* 11 (2023), 35965–35983. doi:10.1109/ACCESS.2023.3265481

[26] Keith Nabors and Jacob White. 1991. FastCap: A multipole accelerated 3-D capacitance extraction program. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* 10, 11 (Nov. 1991), 1447–1459.

[27] NUMBDA Group, Tsinghua University. n.d. FRW-AGF. Online software package. https://numbda.cs.tsinghua.edu.cn/packages/rwcap_agf.tar.gz Accessed July 19, 2026.

[28] Hung-Chih Ou, Hsing-Chih Chang Chien, and Yao-Wen Chang. 2014. Nonuniform Multilevel Analog Routing With Matching Constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 33, 12 (2014), 1942–1954. doi:10.1109/TCAD.2014.2363394

[29] Muhammet Mustafa Ozdal and Renato Fernandes Hentschke. 2012. Maze routing algorithms with exact matching constraints for analog and mixed signal designs. In *Proceedings of the International Conference on Computer-Aided Design* (San Jose California, 2012-11-05). ACM, 130–136. doi:10.1145/2429384.2429409

[30] Haoxing Ren, George F. Kokai, Walker J. Turner, and Ting-Sheng Ku. 2020. ParaGraph: Layout Parasitics and Device Parameter Prediction using Graph Neural Networks. In *2020 57th ACM/IEEE Design Automation Conference (DAC)* (San Francisco, CA, USA, 2020-07). IEEE, 1–6. doi:10.1109/DAC18072.2020.9218515

[31] Mohamed Saleh Abouelyazid, Sherif Hammouda, and Yehea Ismail. 2022. RC Parasitic-Aware Layout Analysis and Routing Optimization Methodology. *IEEE Access* 10 (2022), 92740–92759. doi:10.1109/ACCESS.2022.3203077

[32] Synopsys, Inc. n.d. StarRC. Product website. https://www.synopsys.com/implementation-and-signoff/signoff/starrc.html Accessed July 19, 2026.

[33] James H. Tanis, Chris Giannella, and Adrian V. Mariano. 2024. Introduction to Graph Neural Networks: A Starting Point for Machine Learning Engineers. arXiv:2412.19419 [cs.LG] https://arxiv.org/abs/2412.19419

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL] https://arxiv.org/abs/1706.03762

[35] Peng Xu, Jintao Li, Tsung-Yi Ho, Bei Yu, and Keren Zhu. 2024. Performance-Driven Analog Layout Automation: Current Status and Future Directions (Invited Paper). In *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)* (Incheon, Korea, Republic of, 2024-01-22). IEEE, 679–685. doi:10.1109/ASP-DAC58780.2024.10473859

[36] Dingcheng Yang, Haoyuan Li, Wenjian Yu, Yuanbo Guo, and Wenjie Liang. 2023. CNN-Cap: Effective convolutional neural network-based capacitance models for interconnect capacitance extraction. *ACM Transactions on Design Automation of Electronic Systems* 28, 4 (2023), 1–22.

[37] Dingcheng Yang, Wenjian Yu, Yuanbo Guo, and Wenjie Liang. 2021. CNN-Cap: Effective Convolutional Neural Network Based Capacitance Models for Full-Chip Parasitic Extraction. In *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. 1–9. doi:10.1109/ICCAD51958.2021.9643461

[38] Wenjian Yu, Shan Shen, Dingcheng Yang, Haoyuan Li, Jiechen Huang, and Chunyan Pei. 2025. Deep Learning Inspired Capacitance Extraction Techniques. In *Proceedings of the 30th Asia and South Pacific Design Automation Conference*. 106–112.

[39] W. Yu, M. Song, and M. Yang. 2021. Advancements and challenges on parasitic extraction for advanced process technologies. In *Proc. ASP-DAC*. 841–846.

[40] W. Yu and X. Wang. 2014. *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*. Springer.

[41] W. Yu, H. Zhuang, C. Zhang, G. Hu, and Z. Liu. 2013. RWCap: A floating random walk solver for 3-D capacitance extraction of very-large-scale integration interconnects. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* 32, 3 (2013), 353–366.

[42] Haoyi Zhang, Xiaohan Gao, Haoyang Luo, Jiahao Song, Xiyuan Tang, Junhua Liu, Yibo Lin, Runsheng Wang, and Ru Huang. 2023. SAGERoute: Synergistic Analog Routing Considering Geometric and Electrical Constraints with Manual Design Compatibility. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 1–6. doi:10.23919/DATE56975.2023.10137296

[43] Haoyi Zhang, Xiaohan Gao, Zilong Shen, Jiahao Song, Xiaoxu Cheng, Xiyuan Tang, Yibo Lin, Runsheng Wang, and Ru Huang. 2024. SAGERoute 2.0: Hierarchical Analog and Mixed Signal Routing Considering Versatile Routing Scenarios. In *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 1–6. doi:10.23919/DATE58400.2024.10546542

[44] Lihong Zhang, Nuttorn Jangkrajarng, Sambuddha Bhattacharya, and C.-J. Richard Shi. 2008. Parasitic-Aware Optimization and Retargeting of Analog Layouts: A Symbolic-Template Approach. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 27, 5 (2008), 791–802. doi:10.1109/TCAD.2008.917594

[45] Zheng Liu and Lihong Zhang. 2009. Performance-constrained parasitic-aware retargeting and optimization of analog layouts. In *2009 Canadian Conference on Electrical and Computer Engineering* (St. John's, NL, Canada, 2009-05). IEEE, 1194–1197. doi:10.1109/CCECE.2009.5090314

[46] Keren Zhu, Mingjie Liu, Yibo Lin, Biying Xu, Shaolan Li, Xiyuan Tang, Nan Sun, and David Z. Pan. 2019. GeniusRoute: A New Analog Routing Paradigm Using Generative Neural Network Guidance. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)* (Westminster, CO, USA, 2019-11). IEEE, 1–8. doi:10.1109/ICCAD45719.2019.8942164