IncPIRD: Fast Learning-Based Prediction of Incremental IR Drop

Chia-Tung Ho[2] and Andrew B. Kahng[1,2]
[1]CSE and [2]ECE Departments, UC San Diego, La Jolla, CA, USA
{c2ho, abk}@ucsd.edu

***Abstract*—The on-chip power delivery network (PDN) is an essential element of physical implementation that strongly determines functionality, quality and reliability of a given IC product. To meet IR drop requirements, a denser power grid is desirable. On the other hand, to meet timing and layout density requirements, a sparser power grid leaves more resources for routing. Often, numerous time-consuming iterations among PDN design, IR analysis, and floorplanning or placement are needed during the physical implementation of modern high-performance designs. Thus, fast and accurate incremental IR prediction has emerged as a critical need, as it can potentially reduce the turnaround time between design and analysis and help improve design convergence. In this work, we apply superposition and partitioning techniques to extract relevant electrical features of a given SOC floorplan and PDN. We then use a machine learning model to predict the updated static IR drop for each power node (having tap current source attached) in the design throughout a series of changes (PDN modification, block movement, block power change, power pad movement) to the SOC floorplan, without needing to rerun a golden IR drop tool. We develop our model with more than 150 generated SOC floorplans with different PDN structures in $28nm$ foundry technology. Compared to an industry-leading, golden IR drop signoff tool (ANSYS *RedHawk*), we achieve 20-$1000\times$ speedup with less than $1mV$ average absolute error and approximately $5mV$ maximum absolute error.**

I. Introduction

The on-chip PDN is a key determinant of modern SOC product quality, as it affects both performance (IR drop, timing) as well as area and cost (routability, layout density, metal stack). Today, SOC physical implementation requires many iterations among PDN design, IR analysis, and floorplan/placement updates. Hence, there is a need for fast and accurate incremental IR prediction to reduce turnaround times in the design-analysis loop, and to help improve design convergence. This need has been well-noted in the research literature, e.g., [1] proposes an incremental IR drop analysis flow for use in a physical synthesis iteration, and [2] [3] study the conflict between PDN robustness and routability.

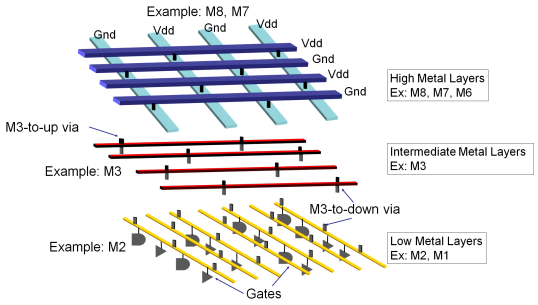


Fig. 1: Illustration of a typical on-chip PDN structure.

Figure 1 illustrates a typical on-chip PDN structure. Power pads connect to the high (upper-most) metal layers and deliver power to cells through intermediate metal layers and lower metal layers. Often, the higher layers are used with maximum density to reduce IR drop; lower metal layers' pitch is determined by the standard-cell height. Thus, track usage of intermediate metal layers becomes a major lever to satisfy both IR drop and routability constraints. Note that our work assumes a locality of power integrity analysis that is a consequence of the flip-chip packaging that is used for modern high-performance designs – i.e., affording high density of (C4 bump) power/ground connections in the core region of the SOC.

Modified nodal analysis (MNA) [25] gives the well-known PDN circuit network equation

$$\mathbf{G}\mathbf{x} = \mathbf{b} \tag{1}$$

where $\mathbf{G}$ is an $M \times M$ system matrix, $\mathbf{x}$ is an $M \times 1$ nodal voltage and current variable vector, $\mathbf{b}$ is an $M \times 1$ vector of current and voltage source values, and M is as defined in Table I below. The PDN circuit network equation will change after incremental modification of PDN and/or floorplan layout. With a change or move of macros/cell blocks in the floorplan, which will change locations and values of tap current sources, the equation becomes

$$\mathbf{G}\overline{\mathbf{x}} = \mathbf{b} + \mathbf{\Delta b} \tag{2}$$

where $\overline{\mathbf{x}}$ denotes the voltage of each node after modification. When the PDN structure or power pad locations are modified, the equation becomes

$$\overline{\mathbf{G}}\overline{\mathbf{x}} = \mathbf{b} \tag{3}$$

where $\overline{\mathbf{G}}$ denotes the new system matrix after PDN structure or power pad modifications.[1]

Incremental Prediction of IR Drop (IncPIRD) Problem Statement. Traditional layout-based techniques for fixing IR drop violations include (i) macro/cell block movement, (ii) macro/cell block change, (iii) modification of the PDN to have a denser grid, and (iv) adding/moving power pads. On the other hand, designers can reduce (overdesigned) PDN resources by replacing with sparser PDN grids, removing power pads, etc. We therefore study the following problem of **Incremental Prediction of IR Drop (IncPIRD)**.[2]

Given: LEF, DEF, current distribution, *ANSYS RedHawk (RH) v15.1* [28] technology file, RH IR drop file, and RH power pad file of floorplan design after modifications.

Train: Extract features from input and train a learning-based model to predict static IR drop.

Output: Static IR drop of each power node that has tap current source attached.

Contributions of This Paper. We present *IncPIRD*, a learning-based approach to address the incremental IR drop prediction problem. We use KVL, KCL, branch equations [25], and superposition to efficiently extract relevant electrical features. Along with layout features (PDN structure and chip/block attributes), these enable prediction of static IR drop through incremental changes to the SOC floorplan, without needing to rerun the golden IR drop tool. Our main contributions are as follows.

- Our IncPIRD methodology can handle macro/cell block modification, PDN structure modification, and power pads modification, and can be used across different block/chip sizes and designs without retraining the model.
- We use KVL, KCL, branch equations and superposition to efficiently extract relevant electrical modeling features. Since we capture the IR drop in view of circuit analysis, our model can be used on different designs with different current distributions, power pad locations and PDN structures without retraining as long as the floorplan design's extracted circuit remains within a range that is determined by the training set.

[1]For example, if M3 stripes are removed from the PDN, there are fewer crossovers/vias between M3 and neighboring-layer stripes. Therefore, there are fewer KCL, KVL, and branch equations. Both N and M decrease, and the values in $\mathbf{G}$ change with changed conductance between power nodes. Similarly, if the designer removes power pads, both N^{pad} and M decrease.

[2]In what follows, we use the ANSYS *RedHawk* tool [28] as a representative "golden", signoff-quality power integrity analysis tool. Our proposed methodology is applicable in conjunction with any power integrity analysis tool for which users seek to reduce incremental analysis runtimes.

- The IncPIRD methodology offers scalability since the model input feature set is independent of design size, power distribution and cell library for a fixed technology node and subset of metal layers used for the PDN.
- We propose a criterion by which users can automatically decide when to launch new golden tool runs to update a trained model. This criterion is based on the delta between new input features and the feature training range of the current model (see Section IV-D and Figure 3 below).

The remainder of this paper is organized as follows. Section II reviews related works. Section III presents our flow, KCL and KVL-based feature extraction methodology, and the IncPIRD prediction model. Section IV describes experimental confirmations. We conclude in Section V.

II. Related Previous Works

We summarize two key threads of relevant works within the large literature on IR drop analysis and mitigation.

Simulation-Based IR Analysis Works. Numerous IR drop analysis methods span hierarchical methods [4], Krylov-subspace methods [5] [6], multi-grid techniques [7] [8] [9] [10], random walk algorithms [11], and vectorless verification methods [12]. While these can perform well on power grid analysis, they may reanalyze a modified design as a whole even when the power grid has been modified only locally. *Incremental* methods that take advantage of known information to simulate or verify an updated power grid have been proposed in [13] [14] [15] [16]. The authors of [13] [14] [15] use sparse recovery and orthogonal matching pursuit to reduce the dimensionality of the incremental IR drop problem. However, runtime complexity is highly dependent on the number of changed regions; if the design is changed significantly, runtime complexity is similar to that of performing from-scratch power grid simulation on the entire design.

Compared to previous IR drop simulation methods, our IncPIRD approach can directly and quickly predict IR drop at each instance node without solving Equation (1) for the entire system. Furthermore, IncPIRD can efficiently predict updated IR drop values even when significant PDN structure modifications are made, such as changing the pitch of PDN structure (note that a small change in PDN stripe pitches can have a large impact on the dimension of $\mathbf{G}$).

Machine Learning-Based (Incremental) IR Estimation Works. Machine learning-based methods have been used in recent works on design-stage IR drop classification or prediction. A fast power integrity classifier is proposed in [17] to detect IR/EM hotspots. However, it does not provide the hotspot location or the worst IR value, both of which may be needed in the context of PDN design. ECO-based dynamic IR drop prediction methods are proposed in [18] and [19] to reduce analysis turnaround time for ECO steps involving cell location changes. The model in [18] requires retraining for different designs, and its error increases with the number of ECOs. [19] uses a regional model to predict dynamic IR drop, with more than $10\times$ speedup (at the cost of analysis error) compared to a leading industry tool.

In contrast to the previous machine learning-based methods, our (IncPIRD) work handles layout modifications to macro/cell block locations, PDN structure, and power pads. It can also be used across different block/chip sizes and designs without model retraining. Thus, for incremental IR analysis, IncPIRD makes progress toward combining the flexibility of simulation-based methods (e.g., to handle larger ECOs than cell placement changes) with the speedups and scalability of machine learning-based methods.

III. Incremental IR Drop Prediction

We now describe specifics of our IncPIRD prediction methodology: (i) overall modeling flow, (ii) methodology for feature extraction, (iii) model input features, (iv) complexity analysis of feature extraction, and (v) use of XGBoost for modeling.

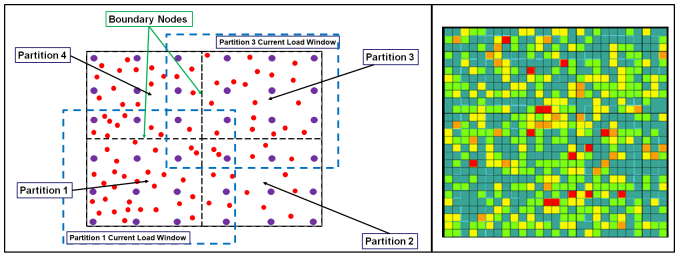


Fig. 2: Left: Illustration of *partitions* in the PDN. Black dashed lines delimit the four partitions. Blue dashed lines delimit the current load windows CW_1 and CW_3 of Partitions 1 and 3. Note that a given partition's current load window has overlaps with other partitions' current load windows. Purple dots indicate power pads, and red dots represent tap current sources. Right: Power distribution map of a generated design from *RedHawk*.

ML-Based Modeling Flow. Figure 3 shows our proposed training and prediction flows. In the *training* phase, we generate different PDN structures, power pad locations, current distributions, and chip/block sizes with a foundry $28nm$ technology in *Cadence Innovus Implementation System v17.1* [29].[3] Layout including power/ground networks is output in DEF format. We then run IR drop simulation in the golden RH tool; IR drop of each node and current distribution are used to train our model.

In the *prediction* phase, we first input DEF, LEF, RH technology file, RH GSR (global system requirements) file, and RH power pad and current distribution files [28] to build an initial database. In our envisioned usage scenario, a designer will make modifications to the PDN or macro/cell block (i.e., to fix IR violations). IncPIRD will then extract the input features and automatically check whether its model is up-to-date with respect to the feature constraints in Equations (10) and (11) (see Section IV.D below). If the input features satisfy the constraints, the IR drop value of every power node (having tap current source attached) is predicted with the model. Otherwise, we export modified DEF and run RH and update the model with the design data. If IR violations remain, the designer will perform another round of incremental fixing and iterate the prediction flow again. In consideration of continuous learning methodology [20] [30], we also propose running RH to update the model whenever a counter of incremental modifications exceeds max_iter. Terminologies and notations used in this paper are given in Table I.

Methodology of Extracting Electrical Input Features. To exploit the spatial locality characteristics of (flip-chip) power grids [21], we divide the PDN into several smaller partitions. In Figure 2 (left), black dashed lines show four partitions and blue dashed lines show two current load windows CW_1 and CW_3 of Partitions 1 and 3. The current load window of each partition overlaps with other partitions' current load windows. Larger and overlapped current load windows are used to accurately extract the pulldown component, PC_{down}, on boundary nodes.[4] For a given partition, we calculate the PC_{down} of internal nodes of the partition by considering the tap current sources in its current load window.

[3]We use industry-standard commercial EDA tools as the experimental testbed for our work. No "benchmarking" of any commercial EDA tool is intended in, or should be inferred from, the data that we report.

[4]Throughout this paper, we fix the size of partitions at $500\mu m \times 500\mu m$, the size of current load windows at $750\mu m \times 750\mu m$, and ϵ_{th} at 0.001.

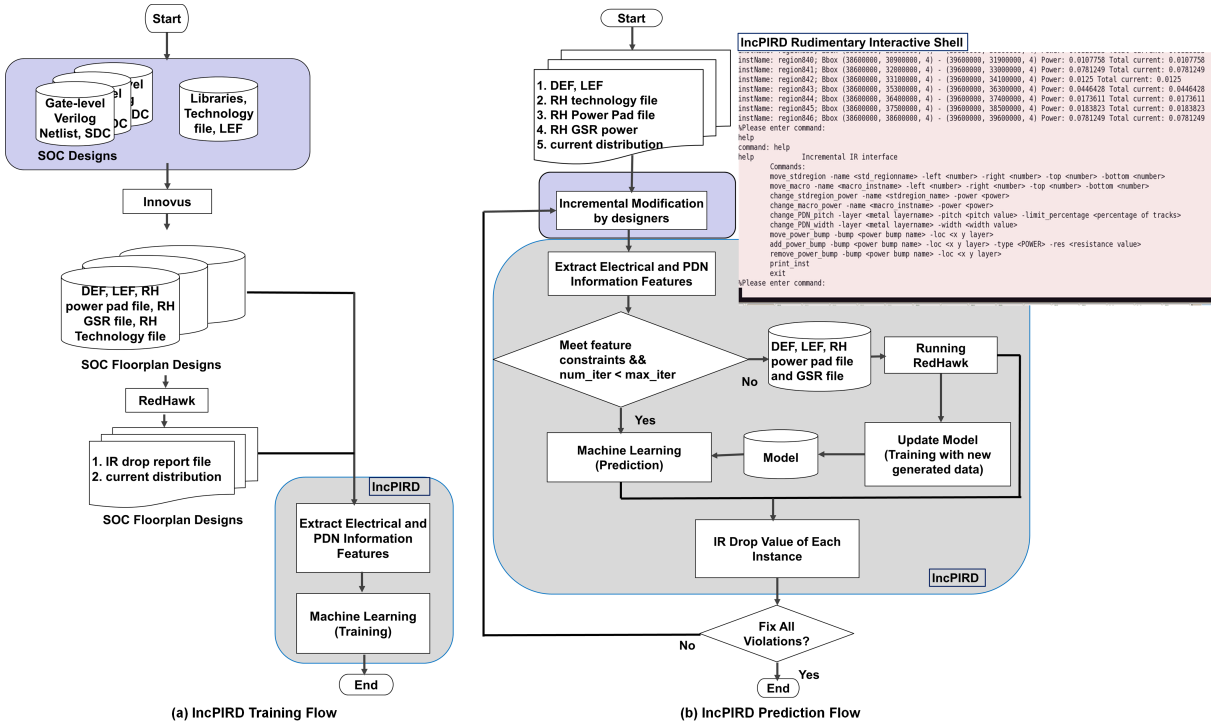


Fig. 3: Overall flow: (a) Training flow, and (b) Prediction flow.

TABLE I: Terminology and notation table.

Name	Description
N	Number of power nodes after PDN circuit extraction.
L	Number of metal layers used in PDN.
N^{pad}	Number of power pads in the design.
N^{tap}	Number of tap current sources in the design.
N_l^{track}	Number of tracks on metal layer l.
$N_{l,max}^{track}$	Maximum number of tracks among L metal layers.
M	Dimension of system matrix G, where $M = N + N^{pad}$.
N_p^{tap}	Number of tap current sources in partition p.
$N_{p,max}^{tap}$	Maximum number of tap current sources among partitions.
$N_{p,w}^{tap}$	Number of tap current sources in current load window of partition p.
$N_{p,w,max}^{tap}$	Maximum number of tap current sources in current load window, among all partitions.
N_{part}	Number of partitions.
$N_{impacted\_part}$	Number of impacted partitions during feature extraction in Algorithm 1.
CW_p	Current load window of partition p.
I_k	k^{th} tap current source.
R_{sl_k}	Effective resistance of shortest path resistance between power pads and k^{th} tap current source in Equation (4).
R_{sn}	Effective resistance of shortest path resistance between power pads and node n.
R_{nl_k}	Shortest path resistance value between k^{th} tap current source and node n.
$IR_{n,k,symbolic}$	Symbolic IR drop value at node n caused by k^{th} tap current source in Equation (6).
$IR_{n,symbolic}$	Symbolic IR drop value at node n in Equation (7).
PC_{up}	Pullup component.
PC_{down}	Pulldown component.
$PC_{up,n}$	Pullup component of node n.
$PC_{down,n}$	Pulldown component of node n.
$PC_{down,n,ori}$	Pulldown component of node n before calculation in Algorithm 2.
$List_{tap}$	A set of tap current sources that need to be considered in the calculation of PC_{down}.
R_f	Feature training range defined in Section IV-D.
$Q_{impacted\_part}$	A container which stores impacted partitions and corresponding $List_{tap}$'s.

We further limit PC_{down} extraction error with a threshold parameter ϵ_{th}. If PC_{down} on the boundary caused by $List_{tap}$ of a given partition exceeds the threshold ϵ_{th}, we say that each neighbor partition is *impacted*, i.e., the neighbor partition must consider the effect of $List_{tap}$ of the given partition. We calculate and extract PC_{up} considering all the power pads in design.

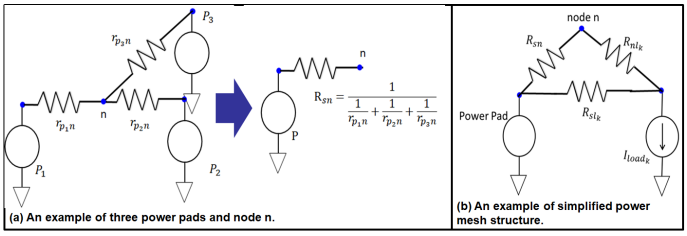


Fig. 4: (a) An example of three power pads and node n. (b) An example of a simplified power mesh structure.

Whenever the designer makes any modification, IncPIRD will update an internal database and extract the electrical features based on the type of modification made. (i) If **only macro/cell blocks are modified**, then PC_{down} is updated to consider the difference of tap current sources, $\Delta \mathbf{b}$, in the impacted partitions in Equation (2). (ii) If **PDN structure or power pads are modified**, i.e., R_{sl_k}, R_{sn}, and R_{nl_k} in Equation (7) are changed, then the electrical features must be recalculated in every partition. Algorithm 1 shows the IncPIRD prediction flow.

Model Input Features. We now describe the extraction of electrical features for each PDN node. Besides these electrical features, chip/block sizes and PDN stripe pitches per layer are also used as input features.

Pullup Component. The pullup component of node n is the effective resistance between itself and power pads, R_{sn}. Figure 4(a) shows an example with three power pads. R_{sn} can be obtained by calculating parallel resistance values from the power pad nodes to n. Here, r_{p_1n}, r_{p_2n}, and r_{p_3n} are shortest path resistances. With N^{pad} voltage sources,

$$PC_{up,n} = R_{sn} = \frac{1}{\sum_{p=0}^{N^{pad}} \frac{1}{r_{pn}}} \tag{4}$$

Pulldown Component. Based on the superposition methodology, only one tap current source is turned on at a time. Therefore, the sum of currents from all the power pads is equal to the turned-on k^{th} tap

Algorithm 1 IncPIRD prediction algorithm.

Procedure IncPIRD
Input: DEF, LEF, *RedHawk* power pad / technology file, and current distribution
Output: Static IR drop of each power node attached by tap current sources

Partition the PDN and create partition database.
for each partition p in all partitions **do**
  $List_{tap}$ = tap current sources in CW_p;
  CalcPartitionElectricalFeature($Q_{impacted\_part}$,True, $List_{tap}$);
end for
CalcImpactPartitionFeature($Q_{impacted\_part}$, True);
$num\_iter = 0$
Violations = getIRViolations;
while Violations > 0 **do**
  Enter IncPIRD rudimentary interactive shell;
  if Macro/Cell block modification has been made **then**
    if Add/Move macro/cell block **then**
      Update tap current sources location;
    end if
    Find set of modified partitions, P_{mod};
    for each partition p in P_{mod} **do**
      $List_{tap}$ = delta tap current sources in CW_p;
      CalcPartitionElectricalFeature($Q_{impacted\_part}$, False, $List_{tap}$);
    end for
    CalcImpactPartitionFeature($Q_{impacted\_part}$, True);
  end if
  if PDN structure or power pad modification has been made **then**
    if PDN structure change **then**
      Update PDN structure database;
      Distribute tap current sources to new PDN power nodes;
    end if
    if power pad change **then**
      Update power pad database;
    end if
    for each partition p in all partitions **do**
      $List_{tap}$ = tap current sources in CW_p;
      CalcPartitionElectricalFeature($Q_{impacted\_part}$,True, $List_{tap}$);
    end for
  end if
  if (Equations (10) and (11) satisfied) && (num_iter < max_iter) **then**
    Perform model prediction;
  else
    Export DEF, LEF, *RedHawk* GSR file, *RedHawk* power pad file;
    Run RH static IR analysis;
    Update model with new data;
  end if
  num_iter++;
  Violations = getIRViolations;
end while

Algorithm 2 Calculation of electrical features in each partition.

Procedure CalcPartitionElectricalFeature
Input: $Q_{impacted\_part}$, isCalcPullU, $List_{tap}$
Output: $Q_{impacted\_part}$

for each power node, n, with tap current source in $List_{tap}$ attached **do**
  if isCalcPullU == True **then**
    for each power pad, p, in $PowerPadList$ **do**
      Calculate shortest path resistance r_{pn} (Section III);
      $G = G + \frac{1}{r_{pn}}$
    end for
    $PC_{up,n} = \frac{1}{G}$
  end if
  Restore $PC_{down,n_b,ori}$ for each boundary node before calculation;
  for each tap current source, I_k, in $List_{tap}$ **do**
    Calculate R_{nl_k};
    $PC_{down,n} = PC_{down,n} + I_k(\frac{R_{sl_k}}{R_{sn}+R_{sl_k}+R_{nl_k}})R_{sn}$;
  end for
end for
for each boundary node, n_b **do**
  if $PC_{down,n_b} - PC_{down,n_b,ori} > \epsilon_{th}$ **then**
    p_n = neighbor partition of p_s;
    Add p_n and $List_{tap}$ information to $Q_{impacted\_part}$;
  end if
end for

Algorithm 3 Impacted partition feature calculation.

Procedure CalcImpactPartitionFeature
Input: $Q_{impacted\_part}$, isCalcPullU

while $Q_{impacted\_part}$ is not empty **do**
  Pop out partition p and $List_{tap}$ in $Q_{impacted\_part}$;
  $List_{tap}$ = $List_{tap}$ excludes tap current sources in CW_p;
  p -> CalcPartitionElectricalFeature($Q_{impacted\_part}$, isCalcPullU, $List_{tap}$);
end while

current source, I_k.

$$\sum_{i=0}^{N^{pad}} I_i = I_k \tag{5}$$

The power grid network with only one turned-on tap current source could be simplified as shown in Figure 4(b). From KVL and KCL, the symbolic IR drop of node n in this simplified network could be derived from Equation (6). Here, the R_{sl_k} is calculated with $n = l_k$ in Equation (4).

$$IR_{n,k,symbolic} = I_k(\frac{R_{sl_k}}{R_{sn} + R_{sl_k} + R_{nl_k}})R_{sn} \tag{6}$$

By superposition, the symbolic IR drop caused by each tap current source of node n can be summed as

$$PC_{down,n} = IR_{n,symbolic} = \sum_{k=1}^{N^{tap}} I_k(\frac{R_{sl_k}}{R_{sn} + R_{sl_k} + R_{nl_k}})R_{sn} \tag{7}$$

Design and Technology Information. The IR drop is closely related to the technology parameters and power grid design. Sheet resistances of metal layers are used when we calculate R_{sl_k}, R_{sn}, and R_{nl_k}, and PDN stripe pitches per metal layer greatly affect the IR drop. Last, chip/block dimensions affect power density for a given total power consumption. Hence, these parameters are used as input features of our model.

Electrical Features of Neighbor Nodes. Due to the locality characteristics [21] of power grids, a nodal voltage is greatly affected by its neighbor nodes. Therefore, the PC_{up} and PC_{down} of neighbor nodes are used as input features. We choose the neighbor nodes based on the physical metal direction. For example, if the direction of metal layer of the targeted power node is horizontal (resp. vertical), the nodes immediately to its left and right (resp. immediately above and below) are considered as its neighbor1 and neighbor2, respectively.

In summary, the input features for our model are as listed in Table II. Our model is independent of power distribution because we use superposition to extract PC_{down}. Different cell libraries have different tap current values of each cell. Our model is independent of cell libraries because we consider the tap current value to extract features. Model training with different chip/block sizes enables our model to handle different layout dimensions. In this way, our methodology enables model independence from PDN structure, cell libraries, and chip/block sizes.

TABLE II: Input features for our IR drop prediction model.

Input Feature	Description
$width$	Width of the Chip/Block.
$length$	Length of the Chip/Block.
$Pitch_{top,h}$	Pitch of top horizontal metal layer.
$Pitch_{top,v}$	Pitch of top vertical metal layer.
$Pitch_{inter,h}$	Pitch of intermediate horizontal metal layer.
$Pitch_{inter,v}$	Pitch of intermediate vertical metal layer.
$Pitch_{lower}$	Pitch of lower metal layer.
$PC_{down,n}$	Pulldown component at node n.
$PC_{up,n}$	Pullup component at node n.
$PC_{down,neighbor1}$	Pulldown component of node n's Neighbor1.
$PC_{up,neighbor1}$	Pullup component of node n's Neighbor1.
$PC_{down,neighbor2}$	Pulldown component of node n's Neighbor2.
$PC_{up,neighbor2}$	Pullup component of node n's Neighbor2.

Feature Extraction Complexity Analysis. We now discuss time complexity of electrical feature extraction considering a design with N^{pad} power pads, N^{tap} tap current sources, and N power nodes. We note that $N^{pad} \ll N$ and $N^{tap} \ll N$, because tap current sources are attached to only the lower metal layers of PDN (recall Figure 1). After partitioning the PDN, we obtain N_p^{tap}, $N_{p,max}^{tap}$, $N_{p,w}^{tap}$, and $N_{p,w,max}^{tap}$ as defined in Table I. To predict the IR drop value of each instance, we extract the electrical features of the power node with tap current source attached. Before calculating the electrical features, we need to calculate the shortest path resistances. Here, we use map structure to store tracks of each metal layer; hence, the complexity to find the via points between two metals is $O(N_l^{track} \log(N_l^{track}))$. Finding the shortest path between two nodes has complexity $O(\sum_{l=1}^{L} N_l^{track} \log(N_l^{track}))$, where L is the number of metal layers between these two nodes. Therefore, finding the shortest path between any two nodes in the PDN has complexity bound $O(LN_{l,max}^{track} \log(N_{l,max}^{track}))$, where $N_{l,max}^{track} \ll N^{tap} \ll N$.

In more detail: (i) For the PC_{up}, the complexity of calculating PC_{up} is $O(N^{tap} N^{pad} (LN_{l,max}^{track}(\log(N_{l,max}^{track})))$.[5] (ii) For the PC_{down}, after partitioning the PDN, the complexity of calculating PC_{down} is $O(N_{part} N_{p,max}^{tap} N_{p,w,max}^{tap} (LN_{l,max}^{track} \log(N_{l,max}^{track})))$. From superposition, the symbolic IR drop of every power node with tap current source attached needs to be calculated $N_{p,w}^{tap}$ times as shown in Equation (7), and we need to calculate R_{nl_k} each time. Here, we use the upper bound $N_{p,max}^{tap}$ and $N_{p,w,max}^{tap}$ in the complexity calculation. In general, we can approximate $N_{impacted\_part} + N_{part}$ to N_{part} with sufficiently large partition size [21]. As a result, the complexity is multiplied by N_{part}. (iii) For design and technology information feature extraction, the complexity is $O(1)$. In summary, the total complexity of feature extraction is $O((LN_{l,max}^{track} \log(N_{l,max}^{track})(N^{pad} N^{tap} + N_{part} N_{p,max}^{tap} N_{p,w,max}^{tap}))$, where $N_{part} \ll N_{l,max}^{track}$. With fixed partition size, the complexity grows by $(LN_{l,max}^{track} \log(N_{l,max}^{track}))$ times when $N_{l,max}^{track}$ increases.

Modeling With XGBoost. We use XGBoost [24] as our machine learning model. XGBoost implements machine learning algorithms under a gradient boosting framework. The parallel gradient boosted regression tree (GBRT) achieves state-of-the-art results on many machine learning problems. A given data set is of form $D = (x_i, y_i)$, where $(|D| = N^{tap}, x_i \in R^m, y_i \in R)$. Here, N^{tap} is the number of power nodes having attached tap current sources in the training set; m is the number of input parameters in Table II; and y_i is the golden IR drop value. In Equation (8), the predicted IR drop, $\widehat{y_i}$, is an ensemble of K additive functions

$$\widehat{y_i} = \sum_{k=1}^{K} f_k(x_i), f_k \in F \tag{8}$$

where *F* is the space of regression trees. Our goal is to minimize

$$L(\phi) = \sum_i l(y_i, \widehat{y_i}) + \sum_k \Omega(f_k) \tag{9}$$

where each $l(y_i, \widehat{y_i})$ is a differentiable convex function that measures the difference between the predicted IR drop $\widehat{y_i}$, and the golden IR drop y_i. (In our work, we seek to minimize mean absolute error.) Ω is a function that penalizes the complexity of the model. We use grid search and 10-fold cross-validation [23] in-built to XGBoost to determine the values of hyperparameters, such as min_child_weight, max_depth, $subsample$, $colsample\_bytree$, and eta.

[5]For every power node with tap current source attached, we must calculate the shortest path resistance from the power node to each power pad and calculate the R_{sn}.

TABLE III: Training data information.

Technology	$28nm$ foundry technology
Package	Flip Chip
Chip/Block Size	$6000\mu m - 30000\mu m$
Power	$10W - 20W$
Intermediate Metal Layer Track Usage	$10\% - 20\%$
Supply Voltage	$1V$
Power Pad Pitch	$250\mu m - 1000\mu m$ [22] [31]
Max Number of Instances	756850

TABLE IV: Design information.

Name	# of Nodes	Total Power (W)	# of tap current sources
Design1	5.4 million	10.8756	95250
Design2	9.6 million	14.3402	263950
Design3	17 million	19.2701	481150
Design4	27 million	24.8753	742300
Design5	10.6 million	14.3485	182900

IV. Experimental Results

A. Experimental Setup

We implement our work in C++ with LEF/DEF reader/writer parsers [27] and OpenMP [26]. All experiments are performed on a 2.6GHz Intel Xeon Dual-CPU server with 256GB RAM.

Table III shows the information of generated SOC floorplan designs. We generate 150 SOC floorplan designs with different power distributions, PDN structures, and power pad pitches to train our model. In Figure 2 (right) depicts one of the SOC floorplan designs generated with nonuniform power distribution. The training flow is as shown in Figure 3. First, we split the generated data to 80% training data and 20% testing data. Second, we use 10-fold cross-validation [23] for setting the parameters of our model. Third, we split the 80% training data such that 64% of original data is used for model training and 16% of original data is used for model validation. The total runtime to extract input features from generated SOC floorplan designs and to train the model is around 2.5 hours. However, running *RedHawk* to obtain the golden IR for training for a given SOC floorplan design takes more than 12 hours.

B. Experiment 1: Model Accuracy

Figures 5(a) and (b) show predicted IR drop values versus golden IR drop values for training and testing sets. The solid blue line in the middle indicates a perfect correlation between golden IR drop and predicted IR drop. The upper and lower green and black solid lines are $2mV$ and $5mV$ away from the solid blue line, respectively. Figure 5(c) shows that errors of predicted IR drop in training and testing sets are less than $5mV$. The mean average error is -4.26 $\times$ $10^{-3}mV$, with standard deviation of $0.52mV$ (hence, 99.7% of predicted IR drop values are within the 3-sigma range of +/-$1.5mV$).

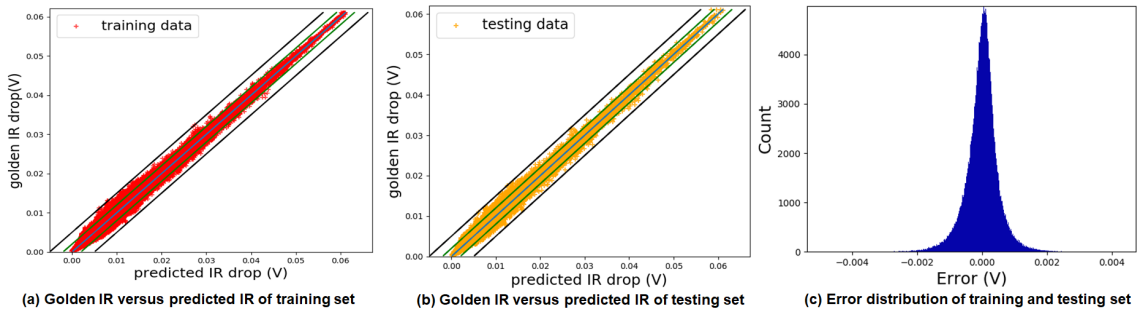
C. Experiment 2: Incremental IR Prediction

For each of Design1 to Design4 in Table IV, we perform **one incremental modification** and validate the predicted IR drop using the golden IR drop tool. We separately perform modifications of macro and cell blocks, power pads, and intermediate metal layer track usage. Table V reports average absolute error e_{avg}, maximum absolute error e_{max}, absolute worst IR (WIR) error e_{WIR}, relative absolute WIR error $e_{WIR,rel}$, feature extraction time t_{fx}, and prediction time t_{pred}. The delta power of changed region (for modifications of macro and cell blocks), delta track usage (for modification of metal layer usage), and power pad modification are shown in the table's Modifications columns.

We see from Table V that our method achieves 22 to 919$\times$ speedup compared with the golden IR tool, even as e_{avg} is less than $1mV$ and e_{max} is $5.07mV$. IncPIRD's fast runtimes are further advantaged in that the tool does not need to read the input files again after designers

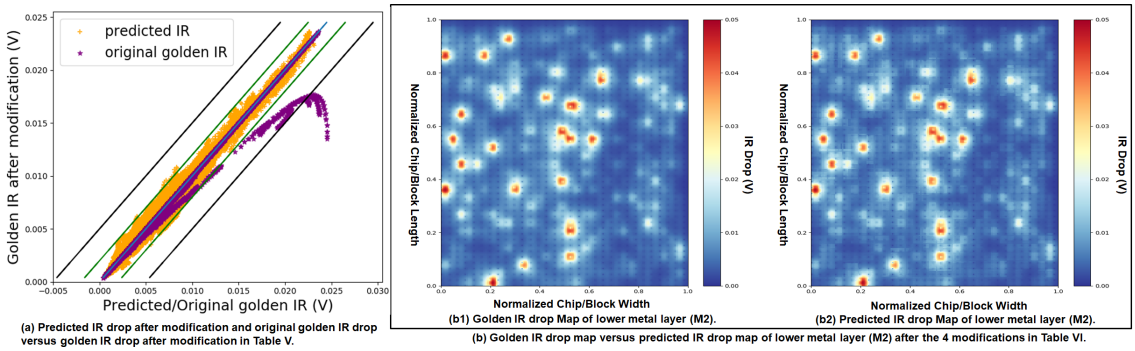
TABLE V: Incremental modification table.

PDN info	Modifications				*RedHawk*		Our Proposed Method (IncPIRD)						Speedup
Name	$\Delta Power_{region}$ (%)	$\Delta Power_{region}$ (mV)	ΔTrack Usage (%)	Power Pad Modification	Runtime (sec)	WIR (mV)	t_{fx} (sec)	t_{pred} (sec)	e_{avg} (mV)	e_{max} (mV)	e_{WIR} (mV)	$e_{WIR,rel}$ (%)	*RedHawk* (×)
Design1	272	20.0	-	-	308	25	0.11	0.28	0.55	3.94	0.57	2.3	787
	-		-	V	308	25	5.76	0.27	0.32	2.51	0.45	1.9	51
	-		5	-	370	22	4.98	0.38	0.31	2.47	0.47	2.1	69
Design2	200	50.0	-		562	24	0.75	0.55	0.29	3.40	0.44	1.8	432
	-		-	V	562	24	24.11	0.56	0.28	3.25	0.44	1.8	22
	-		-10	-	473	21	10.13	0.34	0.45	3.70	1.40	6.7	45
Design3	400	23.4	-	-	2021	27	0.48	1.72	0.34	4.16	0.32	1.2	919
	-		-	V	2021	26	48.84	2.26	0.33	5.07	0.12	0.5	40
	-		-10	-	1489	31	32.65	1.42	0.39	4.07	1.13	3.7	44
Design4	500	76.9	-	-	3273	21	0.70	3.34	0.27	3.04	0.41	2.0	810
	-		-	V	3273	19	72.94	3.01	0.27	3.84	0.53	2.7	43
	-		-10	-	2653	22	52.10	1.87	0.29	3.90	1.27	5.9	49



(a) Golden IR versus predicted IR of training set (b) Golden IR versus predicted IR of testing set (c) Error distribution of training and testing set

Fig. 5: Predicted IR drop versus golden IR drop of (a) training set and (b) testing set, along with (c) error distribution.



(a) Predicted IR drop after modification and original golden IR drop versus golden IR drop after modification in Table V.

(b1) Golden IR drop Map of lower metal layer (M2). (b2) Predicted IR drop Map of lower metal layer (M2).

(b) Golden IR drop map versus predicted IR drop map of lower metal layer (M2) after the 4 modifications in Table VI.

Fig. 6: (a) Predicted IR drop after modifications and original golden IR drop versus golden IR drop after modifications of Design1 in Table V. (b) Golden IR drop map versus predicted IR drop map of lower metal layer (M2) after the 4 modifications in Table VI.

TABLE VI: Results from an example chain incremental modification.

Moves	*RedHawk*		Proposed Method						Speedup
	Runtime (sec)	WIR (mV)	t_{fx} (sec)	t_{pred} (sec)	e_{avg} (mV)	e_{max} (mV)	e_{WIR} (mV)	$e_{WIR,rel}$ (%)	*RedHawk* (×)
1st Move	640	53	0.20	0.43	0.23	1.93	0.67	3.76	1015
2nd Move	634	53	0.19	0.38	0.23	1.93	0.67	3.76	1112
3rd Move	638	52	8.68	0.35	0.23	2.12	0.04	0.24	71
4th Move	2033	48.8	42.57	0.68	0.25	2.37	0.80	5.60	47

perform layout modifications. The runtimes we report for *RedHawk* include parsing and static IR solving time.[6]

Figure 6(a) shows for Design1: (i) the original golden IR drop values, (ii) the IncPIRD-predicted IR drop values after power pad modification, and (iii) the golden IR drop values after power pad modification. We observe that even though the difference between original golden IR and golden IR after modification can exceed $10mV$, our model successfully predicts the IR drop after addition of a power pad near one of the original IR hotspots.

[6]Note that *RedHawk* runtimes do not change with modifications of macro and cell blocks, or of power pads; this is because the number of power nodes in the design does not change significantly. IncPIRD has better speedup with modification of macros/cell blocks because we just need to update features in the affected region.

Table VI summarizes prediction results for **chain incremental modifications** of Design5. We perform four modifications in succession: (i) change the power of one cell block from $10mW$ to $100mW$, (ii) move a macro, (iii) add a power pad, and (iv) use denser intermediate metal layer. We validate the prediction results after each of these incremental modifications. After changing the intermediate metal layer, there are 17 million power nodes. Figure 6(b) shows the golden IR drop map and predicted IR drop map of lower metal layer (M2) after the chain of four incremental modifications. (We export the golden IR drop on M2 from RH and reconstruct the golden IR map with DEF using python. The predicted IR drop map is also reconstructed with DEF using python.)

D. Experiment 3: Robustness of Model

We propose a methodology to determine when to update our model based on the delta of input features between the changed design and the designs in the training set. This is prudent to have in our methodology since the accuracy of our model is related to the diversity of training data. In general, using the current trained model, $\mathbf{W}^*$, to predict new data, $\mathbf{d_{new}}$, could be written as $\mathbf{W}^{*T}(\mathbf{d_{train}} + \delta) = \widehat{y}_{new}$, where $\mathbf{d_{new}} = \mathbf{d_{train}} + \delta$.

The prediction error, e_{pred}, equals $\widehat{y}_{new} - y_{new}$. We have empirically study how to calibrate a bound on delta of the input

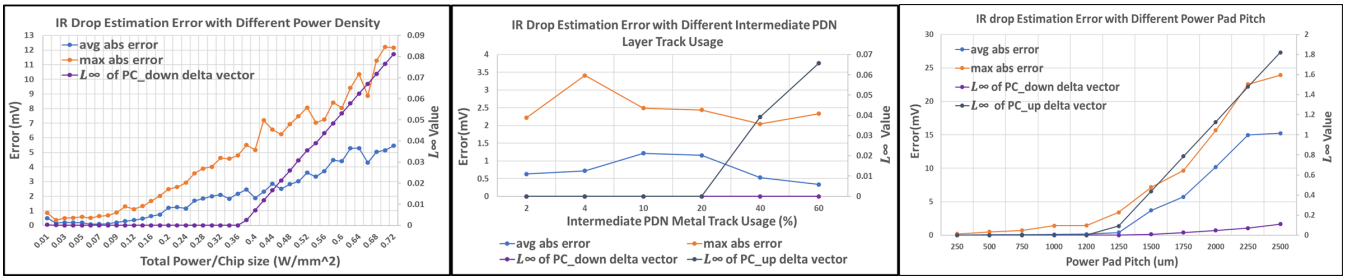


Fig. 7: Prediction error and L_∞ norm of feature delta vector (left) with varying power density (Robustness Expt. 1); (middle) with varying intermediate metal layer track usage (Robustness Expt. 2); and (right) with varying power pad pitch (Robustness Expt. 3).

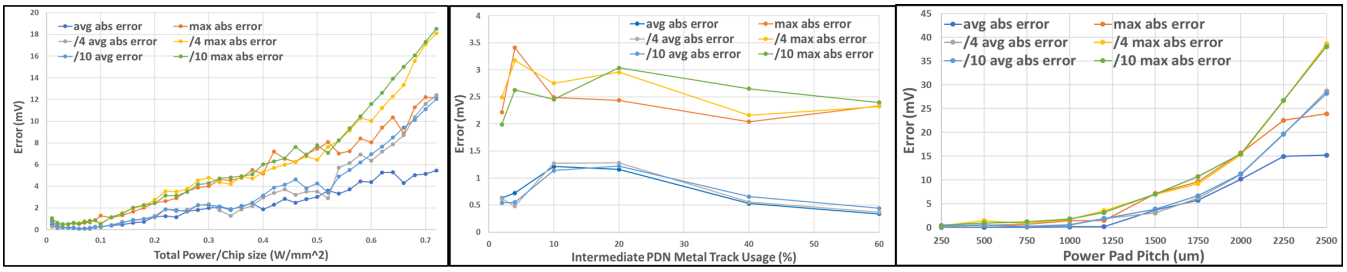


Fig. 8: Prediction error with different amounts of training data: (left) with varying power density (Robustness Expt. 1); (middle) with varying intermediate metal layer track usage (Robustness Expt. 2); and (right) with varying power pad pitch (Robustness Expt. 3).

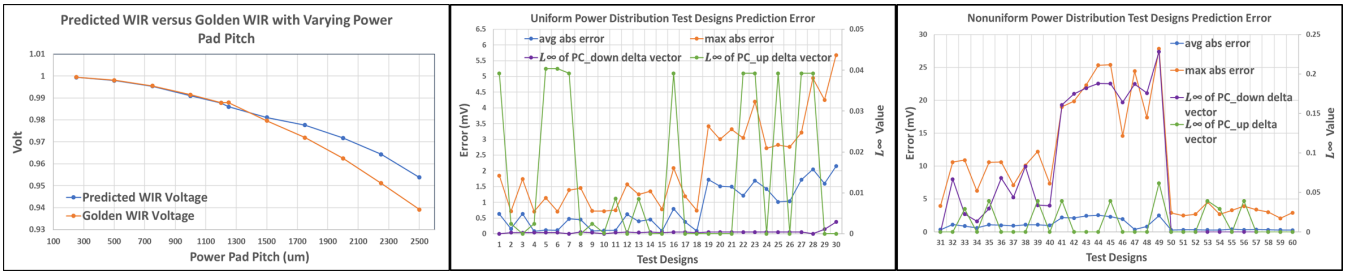


Fig. 9: Left: Golden WIR versus predicted WIR with varying power pad pitch. Middle and right: prediction error of uniform and nonuniform power distribution test designs, respectively.

features, δ_{bound}, which leads to unacceptable e_{pred}. (For example, here we consider $5mV$ as our threshold for acceptable e_{pred}.) In the experiments that we now describe, we vary power density, intermediate metal layer track usage, and power pad pitch until the maximum e_{pred} exceeds the threshold. From this, we can obtain a δ_{bound} of electrical features.

1) **Robustness Experiment 1**: We fix the intermediate metal layer track usage, power pad location, and chip/block size. We vary only the power, and test the trained model to find the bound δ_1. The power density range is 0.1 to 0.2 in the training set.[7]
2) **Robustness Experiment 2**: We fix the power, power pad location, and chip/block size. We vary only the intermediate metal layer track usage, and test the trained model to find the bound δ_2. The intermediate metal layer track usage range is 10% to 20% in the training set.
3) **Robustness Experiment 3**: We fix the power, intermediate metal layer track usage and chip/block size. We vary only the power pad pitch, and test the trained model to find the bound δ_3. The power pad pitch range is $250\mu m$ to $1000\mu m$ in the training set.

[7]We change chip/block power with fix chip/block size to vary the power density. Also, because we only vary power, PC_{down} is the only changed electrical feature.

To obtain the delta of features between the changed design and designs in training set, we first pick the designs that have small difference on the chip/block size and PDN structure. Then, we find the range of values for each feature R_f used in the picked designs in training set. If the input feature of the changed design is within R_f, the delta is zero. On the other hand, the delta equals the distance of the input feature to closest end point of R_f. Given a changed design, we can obtain feature delta vectors that consist of the delta value of each power node (having tap current source attached) in the changed design. We use the L_∞ norm of feature delta vectors to determine δ_{bound} when the maximum e_{pred} is considered. Figure 7 shows prediction error and L_∞ of feature delta vectors with varying power density (Robustness Experiment 1), intermediate metal layer track usage (Robustness Experiment 2), and power pad pitch (Robustness Experiment 3).

Table VII shows the δ_{bound} when the maximum e_{pred} first exceeds $5mV$ in each experiment. The L_∞ of PC_{down} delta vector slightly increases between 0.2 to 0.38 power density range then increases rapidly at >0.38 power density, where the power density is approximately $2\times$ larger than the largest power density in the training set. In Robustness Experiment 2, the maximum e_{pred} and average e_{pred} remain less than $5mV$, even though the delta of $Pitch_{inter,v}$ and $Pitch_{inter,h}$ become large. We therefore do not find any δ_2 from

TABLE VII: Table of robustness δ values.

Robust Expt. 1		Robust Expt. 2		Robust Expt. 3	
$\delta_{1,PC_{down}}$	$\delta_{1,PC_{up}}$	$\delta_{2,PC_{down}}$	$\delta_{2,PC_{up}}$	$\delta_{3,PC_{down}}$	$\delta_{3,PC_{up}}$
0.0025	0	-	-	0.00846	0.4375

Robustness Experiment 2.

Figure 9(a) shows the predicted worst IR drop value and golden worst IR drop value with different power pad pitches (Robustness Experiment 3). Here, the maximum e_{pred} exceeds $20mV$; at the same time, the predicted worst IR drop of our model still follows the trend seen with golden IR drop.

Finally, we identify the minimum values of $\delta_{PC_{down}}$ and $\delta_{PC_{up}}$, respectively, seen in Table VII. These induce feature constraints in our prediction flow, as given in Equations (10) and (11). It is reasonable in practice to have such constraints on PC_{up} and PC_{down}, inferred from Robustness Experiments, since these electrical features are closely related to IR drop.

$$Constraint1 : \delta_{PC_{down}} < 0.0025 \tag{10}$$

$$Constraint2 : \delta_{PC_{up}} < 0.4375 \tag{11}$$

In Figure 3, if the electrical features of the changed design do not satisfy the feature constraints, the model is updated with the changed design's data. Users and designers can obtain the feature constraints of their trained model to decide when to update the model with this methodology. Last, in Figure 8, we eliminate our training data based on the power, PDN structure and power pad pitch systematically and perform the Robustness Experiments. We observe that the error grows faster when we remove more training data. The error from varying metal layer track usage (Robustness Experiment 2) does not change too much because power density and power pad pitch are fixed and PC_{down} changes little. We conclude that the accuracy of our model can be strongly affected by the diversity of the training data.

E. Experiment 4: Model Accuracy on Different Designs

To confirm the ability to predict incremental IR drop in new designs without retraining our model, we generate 30 different non-uniform power distribution and 30 uniform power distribution test designs and test our model, which is trained by non-uniform power distribution designs from Table III. Table VIII shows the information of generated SOC floorplan designs for this testing. Figures 9(b) and (c) show the prediction error of our model across these test designs.

The e_{avg} and e_{max} of uniform power distribution designs are smaller than those seen for nonuniform power distribution test designs. Also, we observe that most of the test designs with prediction error larger than $5mV$ do not satisfy Equation (10). As a result, our model is quite robust even when faced with totally different power distributions and PDN structure designs as long as the new extracted features satisfy Equations (10) and (11).

TABLE VIII: Attributes of test designs.

Technology	$28nm$ foundry technology
Package	Flip Chip
Chip/Block Size	$6000\mu m - 8000\mu m$
Power	$5W - 30W$
Intermediate Metal Layer Track Usage	$5\% - 25\%$
Supply Voltage	$1V$
Power Pad Pitch	$250\mu m - 500\mu m$

V. Conclusions

We have proposed a learning-based modeling approach, *IncPIRD*, which predicts incremental IR drop at all nodes of an on-chip PDN based on extracted electrical and structural features. The approach can handle modifications of macro/cell block locations, PDN structure, and power pads. In studies with a $28nm$ FDSOI foundry enablement, IncPIRD achieves 22-$1000X$ speedup compared to *RedHawk* with average error less than $1mV$ and maximum error around $5mV$. In Section IV-D above, we also propose a methodology and criterion by which users can automatically know when the current model must be updated with further runs of a golden IR analysis tool, in order to continue satisfying a user-specified error limit. Directions of ongoing work include (i) improvement of accuracy and speed of our method, and (ii) extension of this work to dynamic voltage drop prediction.

Acknowledgments

Research at UCSD is supported by Qualcomm, Samsung, NXP Semiconductors, Mentor Graphics, DARPA (HR0011-18-2-0032), NSF (CCF-1564302) and the C-DEN center.

References

[1] S. Bhattacharya, "Early IR-drop Analysis Flow for Hot-Spot Pre-Emption", *Master's Thesis*, North Carolina State University EE Dept., 2015.
[2] Y. Cao, J. Li, A. B. Kahng, A. Roy, V. Srinivas and B. Xu, "Learning-Based Prediction of Package Power Delivery Network Quality", *Proc. ASP-DAC*, 2019, pp. 160-166.
[3] W.-H. Chang, M. C.-T. Chao, L.-D. Chen, Y.-C. Chiu, C.-H. Lin, S.-P. Mu and C.-H. Tsai, "Generating Routing-Driven Power Distribution Networks with Machine-Learning Technique", *IEEE Trans. on CAD* 36(8) (2017), pp. 1237-1250.
[4] D. Blaauw, R. V. Panda, S. S. Sapatnekar and M. Zhao, "Hierarchical Analysis of Power Distribution Networks", *IEEE Trans. on CAD* 21(2) (2002), pp. 159-168.
[5] C. C.-P. Chen and T.-H. Chen, "Efficient Large-Scale Power Grid Analysis Based on Preconditioned Krylov-Subspace Iterative Methods", *Proc. DAC*, 2001, pp. 559-562.
[6] S.-C. Chang, C.-H. Chou, C.-R. Lee, Y. Shi, N.-Y. Tsai and H. Yu, "On the Preconditioner of Conjugate Gradient – A Power Grid Simulation Perspective", *Proc. ICCAD*, 2011, pp. 494-497.
[7] E. Acar, S. R. Nassif and H. Su, "Power Grid Reduction Based on Algebraic Multigrid Principles", *Proc. DAC*, 2003, pp. 109-112.
[8] H.-Y. Chou, P.-Y. Huang and Y.-M. Lee, "An Aggregation-Based Algebraic Multigrid Method for Power Grid Analysis", *Proc. ISQED*, 2007, pp. 159-164.
[9] K. Chen, J. Hu, M. Zhao and C. Zhuo, "Power Grid Analysis and Optimization Using Algebraic Multigrid", *IEEE Trans. on CAD* 27(4) (2008), pp. 738-751.
[10] Y.-C. Cai, Z.-W. Li, J.-L. Yang and Q. Zhou, "PowerRush: A Linear Simulator for Power Grid", *Proc. ICCAD*, 2011, pp. 482-487.
[11] S. R. Nassif, H.-F. Qian and S. S. Sapatnekar, "Power Grid Analysis Using Random Walks", *IEEE Trans. on CAD* 24(8) (2005), pp. 1204-1224.
[12] J. Wang and X. Xiong, "Vectorless Verification of RLC Power Grids with Transient Current Constraints", *Proc. ICCAD*, 2011, pp. 7-10.
[13] X. Li, P. Sun and M. Y. Ting, "Efficient Incremental Analysis of On-Chip Power Grid via Sparse Approximation", *Proc. DAC*, 2011, pp. 676-681.
[14] L.-C. Chang, Y.-T. Chang, Y.-H. Lee and Y.-M. Lee, "A Robust Incremental Power Grid Analyzer by Macromodeling Approach and Orthogonal Matching Pursuit", *Proc. ASQED*, 2012, pp. 64-70.
[15] C.-T. Ho and Y.-M. Lee, "InTraSim: Incremental Transient Simulation of Power Grids", *IEEE Trans. on CAD* 36(12) (2017), pp. 2052-2065.
[16] Abhishek and F. N. Najm, "Incremental Power Grid Verification", *Proc. DAC*, 2012, pp. 151-156.
[17] L. Kagan, H.-C. Zhang and C. Zheng, "Machine Learning Based Fast Power Integrity Classifier", *arXiv preprint arXiv:1711.03406*, 2017.
[18] E. J.-W. Fang, Y.-C. Fang, C.-M. Li, Y.-C. Li, S.-C. Lin, S.-Y. Lin, Y.-C. Liu and T.-S. Yang, "IR Drop Prediction of ECO-Revised Circuits Using Machine Learning", *Proc. VLSI Test Symposium*, 2018, pp. 1-6.
[19] E. J.-W. Fang, Y.-C. Fang, C.-M. Li, H.-Y. Lin and M.-Y. Sui, "Machine-Learning-Based Dynamic IR Drop Prediction for ECO", *Proc. ICCAD*, 2018, pp. 1-7.
[20] A. Bifet and G. D. F. Morales, "SAMOA: Scalable Advanced Massive Online Analysis.", *J. Machine Learning Research* 16(1) (2015), pp. 149-153.
[21] E. Chiprout, "Fast Flip-Chip Power Grid Analysis via Locality and Grid Shells", *Proc. ICCAD*, 2004, pp. 485-488.
[22] B. H. Meyer, K. Skadron, M. R. Stan, K. Wang and R.-J. Zhang, "Tolerating the Consequences of Multiple EM-induced C4 Pad Failures", *IEEE Trans. on VLSI* 24(6) (2016), pp. 2335-2344.
[23] S. Arlot and A. Celisse, "A Survey of Cross-Validation Procedures for Model Selection", *Statistics Surveys* 4 (2010), pp. 40-79.
[24] T.-Q. Chen and C. Guestrin, "Xgboost: A Scalable Tree Boosting System", *Proc. ACM Intl. Conf. on Knowledge Discovery and Data Mining*, 2016, pp. 786-794.
[25] F. N. Najm, *Circuit Simulation*, John Wiley & Sons, 2010.
[26] OpenMP Architecture Review Board, "OpenMP Application Program Interface, Version 4.0".
[27] LEF/DEF reference 5.7. http://www.si2.org/openeda.si2.org/projects/lefdefnew
[28] RedHawk User Guide, http://www.ansys.com
[29] Cadence Innovus User Guide, http://www.cadence.com
[30] IBM, "Continuous Learning and Model Evaluation", *https://dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ ml-continuous-learning.html*, (2019).
[31] A. Huffman, "50 Micron Pitch Flip Chip Bumping Technology: Process and Applications", *http://www.ewh.ieee.org/soc/cpmt/presentations/cpmt0609a.pdf*, (2006).