XGBIR: An XGBoost-based IR Drop Predictor for Power Delivery Network

[1]Chi-Hsien Pao, [1]An-Yu, Su,[12]Yu-Min Lee
[1]Institute of Communications Engineering, College of Electrical and Computer Engineering, and
[2]Center for mmWave Smart Radar Systems and Technologies,
National Chiao Tung University, Taiwan
Email: {shawn07023.cm05g, aysu.cm07g, yumin}@nctu.edu.tw

***Abstract*—This work utilizes the XGBoost to build a machine-learning-based IR drop predictor, XGBIR, for the power grid. To capture the behavior of power grid, we extract its several features and employ its locality property to save the extraction time. XGBIR can be effectively applied to large designs and the average error of predicted IR drops is less than 6 mV.**

I. Introduction

Power grid contains horizontal and vertical wires and can be modeled as a resistive network. Given a power grid having M nodes, $\mathcal{S}=\{n_1, \cdots, n_M\}$, and coordinates of each node $p(x_p, y_p)$, power bumps are viewed as voltage sources, currents drawn from functional blocks are viewed as current loads, and wire segments are modeled as resistances. We have the equation

$$\mathbf{Gv} = \mathbf{Bu}, \tag{1}$$

where $\mathbf{G}$ is the conductance matrix, $\mathbf{v}$ is the nodal voltage vector, $\mathbf{B}$ is the incidence matrix, and $\mathbf{u}$ is the source vector.

Since it has tens of millions of nodes, directly solving (1) causes high memory and CPU usage. A hierarchical method and a multigrid-like method were used to reduce the problem size [1], [2], and the random walk and the second-order iterative methods were employed to iteratively solve it [3], [4]. [5] utilized effective resistances [6] to derive the closed-form approximation of IR drop, developed fast IR drop analysis algorithms, and utilized the locality of power grid to do parallel computation. However, it made several assumptions, and these lead to lower accuracy as nodes are near the boundary.

With machine learning techniques, we develop an IR drop predictor, XGBIR, to estimate IR drops of local regions or full power grid. XGBIR is used to assist power grid construction in the pre-floorplan/floorplan stage that decides initial structure of grid. Ensemble learning is used to construct our prediction model. Comparing to a single model, ensemble learning combines several models to build a final model and shows better performance in most applications. Our contributions are

- XGBIR is able to estimate IR drops of a partial or a full power grid. Hence, given specific local regions that might violate IR drop constraints, it can fast provide estimated results for saving the construction time of power grid.
- We extract features of power grid by utilizing its locality, using effective resistances, and considering boundary effects, Experimental results show that the extracted features can precisely capture the behavior of power grid.

We organize this work as follows. First, Section II reviews the effective resistance formula in a two layer mesh [6] and the approximated IR drop closed form [5]. Then, Section III details XGBIR. Finally, Sections IV and V show experimental results and conclusions, respectively.

II. Effective Resistance [6] & Closed Form of IR Drop [5]

Given an *infinite* regular power grid and its vertical/horizontal wire segment resistance $R_V=r/R_H=kr$ ($k>0$), [6] derived the effective resistance between two nodes $p(x_p, y_p)$ and $q(x_q, y_q)$ to be

$$\frac{R_{pq}}{r}=\frac{\sqrt{k}}{2\pi}\Big(\ln(m_{x,pq}^2+km_{y,pq}^2)+3.44388\Big)-a_k k-b_k k(k-c_k). \tag{2}$$

Here, $m_{x,pq}=|x_p - x_q|$ and $m_{y,pq}=|y_p - y_q|$. a_k, b_k, and c_k are constants. As $k\rightarrow 1$, a_k=0.033425, b_k=0.0629, and c_k=1.

Given a *finite* regular power grid with $\mathcal{S}_V$ ($|\mathcal{S}_V|=n_V$), a set of nodes having voltage sources, and $\mathcal{S}_I$ ($|\mathcal{S}_I|=n_I$), a set of nodes having current loads, [5] transformed voltage sources but one at node s to current sources and estimated the IR drop at p as

$$IR_{(p)}=\sum_{i\in\mathcal{S}_I}\frac{I_{x_i,y_i}}{2}(R_{sp}+R_{si}-R_{pi})+\sum_{j\in\mathcal{S}_V\setminus\{s\}}\frac{-I_{x_j,y_j}}{2}(R_{sp}+R_{sj}-R_{pj}). \tag{3}$$

Here, I_{x_i,y_i} is the current load at i, I_{x_j,y_j} is the current flowing into j converted from the voltage source at j, and $R_{\times\times}$ is an *approximated* effective resistance between two nodes by (2).

To utilize (2), the grid is *assumed* to be infinite, and (3) suffers from a non-ignorable error of $R_{\times\times}$ as nodes are near the boundary of power grid and its error can be up to 20% [6].

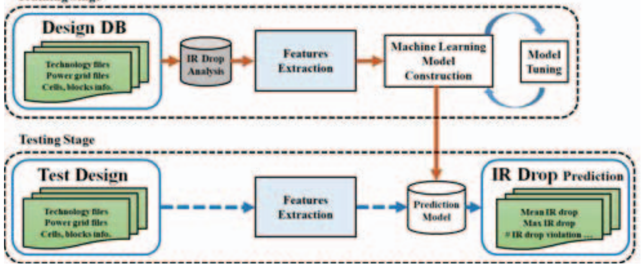

Fig. 1. The overall flow of XGBIR.

III. XGBIR: XGBoost-Based IR Drop Predictor

Fig. 1 shows the overall flow of XGBIR. In the training stage, we construct a set of circuits as the training data set. To obtain their IR drops, we implement a linear circuit solver by using a C++ linear algebra library [7]. Then, we utilize the training data, our developed extracted features of power grid, and XGBoost [8] to build XGBIR. After that, we can employ XGBIR to efficiently predict voltage drops of power grids.

A. Feature Extraction

To predict the IR drop at a node p, we have to study what circumstances will influence its IR drop. We propose several importance features. First, given a power grid, we adopt its known information, as follows, to be our basic features.

1) H_Grid: the number of horizontal power tracks.
2) V_Grid: the number of vertical power tracks.
3) H_Resis: the unit resistance of horizontal wire segment.
4) V_Resis: the unit resistance of vertical wire segment.
5) Tot_I: the total current consumption of the power grid.

The IR drop at a node p relies on its location and devices around it since the power grid is a mesh and IR drops are varied smoothly through nodes. Exploiting these characteristics, we observe some relations and extract the following features.

6) V2N: the relation between node p and voltage sources. When node p is close to voltage sources, as shown in Fig. 2(a), its IR drop decreases since currents provided by voltage sources just need to pass a small amount of resistances to reach it. Thus, voltage sources play the roles of pulling up nodal voltages. According to the locality, most of currents of current loads are provided by voltage sources that are close to current loads. Hence, we extract *V2N* features of node p that are proportional to currents provided by voltage sources and the effective resistances between voltage sources and node p. *V2N* features contain the *global V2N* and *local V2N*.

The *global V2N* of node p describes the effects of M nearest voltage sources to node p and is formulated as

$$V2N\_gb(p) = \sum_{j \in V_{locality}} R_{pj} I_{x_j,y_j}. \quad (4)$$

Here, I_{x_j,y_j} is the converted current of the voltage source at node j contributed by all current loads and $V_{locality}$ is the set of M nearest voltage sources to node p. We set M=10 for the tradeoff between runtime and accuracy.

I_{x_j,y_j} is got by the current division principle of current loads.

$$I_{x_j,y_j} = \sum_{i \in S_I} \frac{G_{ji}}{\sum_{l \in S_V} G_{li}} I_{x_i,y_i}, \quad (5)$$

where I_{x_i,y_i} is the current load value at node i.

To characterize the locality, the *local V2N* of node p describes the effect of its closest voltage source and is

$$V2N\_lc(p) = R_{pc} I_{x_c,y_c}. \quad (6)$$

Here, I_{x_c,y_c} is the converted current value of voltage source at node c that is the closest voltage source to node p.

The prediction model will balance weights of $V2N\_gb(\cdot)$ and $V2N\_lc(\cdot)$ while facing different cases in the learning process.

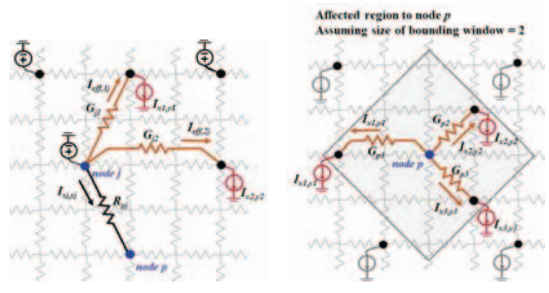
Affected region to node p
Assuming size of bounding window = 2
node j
node p
G_j1
G_j2
R_pj
I_s1,y1
I_s2,y2
node p
G_p1
G_p2
G_p3
(a) *V2N* (b) *I2N*

Fig. 2. Illustration of *V2N* and *I2N* features. (a) *V2N*. The voltage source at node j is the closest voltage source to node p and it provides the *local V2N* for node p. The *global V2N* for node p is the sum of the effects of M nearest voltage sources to node p. (b) *I2N*. The current loads inside the bounding window are counted for pulling down the voltage at node p.

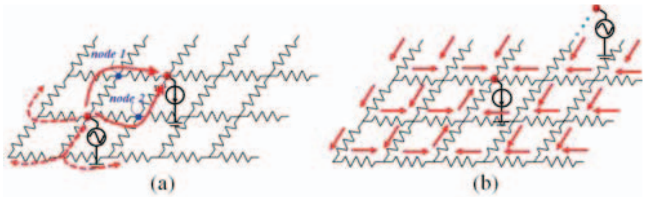

Fig. 3. Illustration of *V2I* feature. (a) The voltage source near the current load. (b) The voltage source far from the current load.

7) I2N: the relation between node p and current loads. In contrast, as shown in Fig. 2(b), when a node is close to current loads, its IR drop becomes more serious since the current load draws currents from nodes around it. Hence, current loads play the roles of pulling down nodal voltages and we might use currents consumed by current loads and effective conductances between current loads and node p to describe this effect. Here, we use $(R_{max} - R_{pi})$ to mimic the conductance effect between node p and a current load at node i where R_{max} is the maximum effective resistance in a given power grid. Also, because of the locality, we set a bounding window (*BW*) for node p to only consider current loads inside *BW*. The *I2N* of node p is

$$I2N(p) = \sum_{i \in I_{locality}} (R_{max} - R_{pi}) I_{x_i,y_i}. \quad (7)$$

Here, I_{x_i,y_i} is the current load at node i. $I_{locality}$ is the set of current loads inside the *BW* of node p and its size is 20×20.

8) V2I: the relation between voltage sources and current loads. Different distributions of voltage sources and current loads have different effects on the IR drop degree. As the voltage source is close to the current load shown in Fig. 3(a), the current flowing to nodes connected to the voltage source will mostly flow to node 1 and node 2 since the current will flow through the least impedance path. Hence, only a small portion of current will flow to rest nodes and has less influence on the IR drop in average. However, when the voltage source is far from the current load as shown in Fig. 3(b), the impedances between four nodes next to the voltage source to the current load will be pretty much the same because of the mesh structure. The current will be distributed more uniform and eventually flow to the current load. As a result, the overall IR drop will increase as well as the maximum IR drop.

We use the current load value and the equivalent effective resistance between the voltage sources and the current load to formulate this phenomenon, and $V2I$ is equal to

$$V2I = \sum_{i \in S_I} \frac{1}{\sum_{j \in S_V} G_{ji}} I_{x_i, y_i}. \quad (8)$$

9) Border: the distance information between node p and the power grid boundary. As node p is close to the boundary, the accuracy of its approximated effective resistances calculated by (2) will be degraded [6]. To quantify this effect, we set its $NBorder_H/NBorder_V$ to be the reciprocal of its distance to the horizontal/vertical boundary.

We also use $VBorder_H/VBorder_V$ and $IBorder_H/IBorder_V$ to quantify how close of voltage sources and current loads to the horizontal/vertical boundary is, respectively. We set $VBorder_H$ to be the average of the reciprocal of the distance between each voltage source to the horizontal boundary. The same way is used to assign $VBorder_V$, $IBorder_H$, and $IBorder_V$. We also find the M nearest voltage sources to node p as similar as $V2N\_gb(p)$ for $VBorder_H/VBorder_V$.

B. Machine Learning Model

There are two main algorithms, bagging and boosting, in ensemble learning. Bagging uses the subset data to train each model and is able to avoid outliers in training data set. Boosting builds a new model focusing on training the data whose predicted results got by the existing model are not accurate enough. We adopt the extreme boosting (XGBoost) [8] that combines the bagging and boosting as our model owing to its high performance and scalable. It also enables the *multi-core* and *out-of-core* operation to deal with huge amounts of data.

XGBoost is developed based on the gradient boosting decision tree (GDBT). The gradient boosting revises the error of existing model and its function is

$$y_i = \sum_{k=1} f_k(\mathbf{X}_i), \quad (9)$$

where y_i is the i-th predicted value, f_k is the k-th tree function, and $\mathbf{X}_i$ is the i-th input data.

XGBoost reduces the over fitting by adding a penalty term in the objective to limit the complexity of tree structure.

$$Obj = \sum_i L(t_i, y_i) + \sum_k \Omega(f_k). \quad (10)$$

$$\Omega(f) = \gamma T + 0.5\lambda ||\mathbf{w}||^2. \quad (11)$$

Here, $L(t_i, y_i)$ is a loss function, $\Omega(f_k)$ is a penalty term, γ and λ are regularization coefficients, T is the number of leaves on a tree, and $\mathbf{w}$ is the weight vector of leaves.

By including the second term of (10), γ controls the number of leaves on a tree and λ helps to prevent the weights of leaves from being a outlier for avoiding the over fitting.

TABLE I
Characteristics of Power Grids

Size Range	#nodes	R_H/R_V (Ω)	$\#V_{src}$	$\#I_{load}$	I_{total} (A)
S (Small)	84,100~396,900	0.1 ≀ 1	100 ≀ 200	70% ≀ 90% nodes	5~15
M(Medium)	396,900~940,900				15~25
L (Large)	940,900~1,716,100				25~35
E (Extra Large)	1,716,100~2,722,500				35~45

TABLE II
Prediction Results of XGBIR for Test Circuits

Size Range	Avg. IR (mV)	Max. IR (mV)	RMSE (mV)	MAE (mV)	Max. Error (mV)	CC (mV)	NRMSE (%)
S	32.5	196.3	5.5	4.0	46.4	0.976	16.9
M	47.4	198.4	6.6	4.8	52.4	0.976	13.9
L	53.8	200.0	7.5	5.6	66.7	0.969	13.9
E	60.6	199.7	7.6	5.5	63.2	0.973	12.5

IV. Experimental Results

We train the prediction model in Python, implement XGBIR in C++ language, and test it on an Intel Xeon E5-2620 v4 2.10 GHz CPU with 128 GB memory. For accuracy comparison, the linear circuit solver is used for obtaining exact IR drops.

A. Validation of XGBIR

To verify XGBIR, based on the benchmarks in [9], we randomly create many power grids and each size range contains 500 different power grids with different parameters as shown in Table I. The supply voltage is 1 V, "#nodes" is the number of nodes, "$\#V_{src}$" is the number of voltage sources, "$\#I_{load}$" is the number of current loads, and "I_{total}" is the total current consumption. 20% of nodes in each power grid are the sample nodes and the rest nodes are discarded since we experimentally discover that they are enough for learning the behavior of power grid. 80% of circuits are used for building the prediction model and 20% of them are used for testing XGBIR. We have over 500 million and 100 million data points for the training and testing, respectively. It is impossible to fit them into the main memory to do any further training. Hence, we implement the *out-of-core operation* in XGBoost and use 5-fold cross-validation with random search for tuning the hyper-parameters.

We use the root mean square error (*RMSE*), the normalized *RMSE* (*NRMSE*), the mean absolute error (*MAE*), and the correlation coefficient (*CC*) to measure our prediction accuracy.

$$NRMSE = \frac{RMSE}{\mu_{IR_{ref}}} \times 100\%, \quad (12)$$

$$CC = \frac{\sum_{p=1}^{N} (IR_{(p),ref} - \mu_{IR_{ref}}) \times (IR_{(p)} - \mu_{IR_{est}})}{\sqrt{\sum_{p=1}^{N} (IR_{(p)} - \mu_{IR_{ref}})^2 \times \sum_{p=1}^{N} (IR_{(p),ref} - \mu_{IR_{est}})^2}}. \quad (13)$$

Here, $\mu_{IR_{ref}}$ is the average voltage drops obtained by the linear circuit solver and $\mu_{IR_{est}}$ is the average predicted voltage drops.

As shown in Table II, *RMSE*'s of XGBIR are less than 7.6 mV, *MAE*'s are less than 5.6 mV, and *CC*'s are higher than 0.969 for all test power grids. Comparing with the closed-form method [5] and the linear circuit solver, Table III shows the results of the worst IR drop power grid in each size range. The closed-form method partitions a power grid to several overlapped sub-circuits and enables the parallel computation. Its *MAE* is higher than 15.1 mV and its maximum error can be up to 185.9 mV. Without considering the boundary effect, its accuracy decreases gradually when nodes move toward the boundary. Its runtime strongly depends on the number of current loads since the IR drop of a node has to consider every single current load effect. However, the predicted IR drops

TABLE III
Prediction Results Comparison

Ckt	Linear Circuit Solver			Closed Form [5]							XGBIR				
	Avg. IR (mV)	Max IR (mV)	Time (s)	#Partitions	Size of Overlapping	MAE (mV)	Max Error (mV)	NRMSE (%)	CC	Time (s)	MAE (mV)	Max Error (mV)	NRMSE (%)	CC	Time (s)
S-w	83.6	196.3	5.86	25	40	19.9	94.1	29.3	0.770	1.8	6.8	39.5	10.1	0.966	1.3
M-w	66.9	198.4	11.5	25	40	15.1	73.6	30.1	0.864	3.7	5.5	47.4	12.5	0.970	3.5
L-w	108.6	200.0	31.4	25	40	23.6	147.3	28.7	0.831	10.5	5.7	32.3	6.7	0.978	7.6
E-w	106.9	199.7	82.7	36	40	23.4	185.9	31.3	0.822	42.4	5.2	34.7	6.3	0.980	22.4

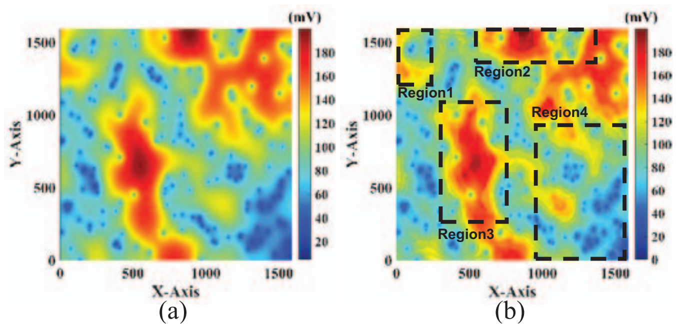

Fig. 4. IR drop maps of E-w. (a) IR drop map obtained by the linear circuit solver. (b) IR drop map obtained by XGBIR.

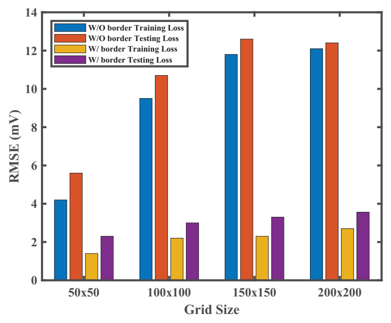

Fig. 5. The influence of with/without including border information into extracted features.

TABLE IV
Local Region Analysis of E-w

Local Region	#nodes	MAE (mV)	Max. Error (mV)	NRMSE (%)	Time (s)
Region 1	82,800	5.7	22.7	6.3	5.0
Region 2	267,615	5.8	33.8	5.6	5.9
Region 3	369,675	6.9	29.0	6.6	7.3
Region 4	587,600	5.1	34.7	8.2	8.5

from XGBIR are very consistent with those from the linear circuit solver. Its *MAE* is less than 7 mV and *CC* is higher than 0.966 for all the worst IR drop cases. Though we ignore less important information while extracting *I2N* by using *BW* for saving runtime, its influence on accuracy is minor. The results also show that the runtime of XGBIR outperforms the linear circuit solver. For the case of E-w, it takes 20 seconds in feature extraction and only 2.4 seconds in IR drops calculation. Fig. 4(a) and Fig. 4(b) are the IR drop distributions of E-w obtained by the linear circuit solver and XGBIR, respectively. Both plots agree very well.

To demonstrate the ability of XGBIR for efficiently predicting IR drops of local regions, we label four regions in E-w shown in Fig 4(b). Table IV shows that XGBIR takes less efforts for analyzing a local region since it can calculate the circuit partially. The runtime of a local region contains two parts. The default time is 3.25 seconds (the same for each region), and the extra time is proportional to the region size.

B. *Influence of Border Information*

To observe the influence of border features, we randomly generate training and testing data with different grid sizes as well as different distributions of power grids to build prediction models *with* and *without* including border features. Fig. 5 shows the comparison. The error increases as the grid size increases if the learning method does not consider the border information. However, once we add the border features into our model with all other features, we can observe that the level of *RMSE* can be well controlled. This demonstrates that the border features do help XGBIR to maintain its ability to catch the behavior of power grid regardless of its size.

V. Conclusion

An efficient IR drop predictor, XGBIR, has been built by utilizing machine learning techniques. With the extracted features, XGBIR can precisely capture IR drop variations under different scenarios.

Acknowledgement

This work was partially supported by the "Center for mmWave Smart Radar Systems and Technologies" under the Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE), partially supported under MOST 108-3017-F-009-001 and MOST 108-2221-E-009 -093 -MY2, in Taiwan, and partially supported by ITRI, Taiwan.

References

[1] M. Zhao, R. V. Panda, S. S. Sapatnekar, and D. Blaauw. Hierarchical analysis of power distribution networks. *TCAD*, 21(2):159–168, 2002.
[2] J. N. Kozhaya, S. R. Nassif, and F. N. Najm. A multigrid-like technique for power grid analysis. *TCAD*, 21(10):1148–1160, 2002.
[3] H. F. Qian, S. R. Nassif, and S. S. Sapatnekar. Random walks in a supply network. In *Proc. DAC*, pages 93–98, 2003.
[4] Y. Zhong and M. D. F. Wong. Efficient second-order iterative methods for IR drop analysis in power grid. In *Proc. ASP-DAC*, pages 768–773, 2007.
[5] S. Köse and E. G. Friedman. Fast algorithms for IR voltage drop analysis exploiting locality. In *Proc. DAC*, pages 996–1001, 2011.
[6] S. Köse and Eby G. Friedman. Effective resistance of a two layer mesh. *TCAS-II*, 58(11):739–743, 2011.
[7] G. Guennebaud and B. Jacob. Eigen v3. [Online]. Available: http://eigen.tuxfamily.org, 2010.
[8] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proc. SIGKDD*, pages 785–794, 2016.
[9] Sani R. Nassif. Power grid analysis benchmarks. In *Proc. ASP-DAC*, pages 376–381, 2008.