Machine Learning based Generic Violation Waiver System with Application on Electromigration Sign-off

Norman Chang, Ajay Baranwal, Hao Zhuang, Ming-Chih Shih, Rahul Rajan, Yaowei Jia, Hui-Lun Liao, Ying-Shiun Li, Ting Ku, Rex Lin

ANSYS, Inc.

Abstract - Manually analyzing the results generated by EDA tools to waive or fix any violations is a tedious, error-prone and time-consuming process. By automating these time-consuming rigorous manual procedures by aggregating key insights across different designs using continuing and prior simulation data, a design team can speed up the tape-out process, optimize resources and significantly minimize the risk of overlooking must fix violations that are prone to cause field failures. In this paper, a machine learning based generic waiver system is proposed which continuously learns to improve with new design data using K-means clustering and nearest neighbor algorithms for risk scoring. The system has been used on new designs to demonstrate on-chip Electromigration (EM) waiver (EMWaiver) mechanism that yielded highly confident results.

I Introduction

Sign-off EDA tools spit out design violations to help inform the designer about any potential issues in the design. Design teams then carefully investigate every violation and take an expert decision to either ignore (waive-off) the violation or resolve it (must-fix) in the design [1]. This process requires iteration, good expertise of design domain knowledge and consumes precious resources and time. Needless to say, this is manual and requires human judgement which can be error prone. Automating this process with an assistive and adaptive learning method helps in eliminating manual risks and increasing efficiency. This can be achieved by using a machine learning based generic waiver system. This system uses historical violation data and past expert judgement to decide if a violation in the new design either needs to be waived or fixed and it continuously learns to improve this decision process with new design and simulation data.

As an example, sign-off task of on-chip Electromigration (EM) as shown in Fig. 1 is always a challenge as it is difficult to keep track of the EM waiving criteria used in past designs and applying it to newer designs. The process of EM waiving involves expert designers who carefully debug and methodically eliminate the risk of failure for each violation. This learning comes with experience and needs to be carried forward to future designs by CAD teams who are responsible for auditing such a process. One of the CAD teams’ responsibility is to enforce meaningful EM waiver system to emphasize must-fix violation to the designers. Advanced FinFET process technologies makes this even more complicated. Having a machine learning based EMWaiver System helps significantly in achieving this goal.

![Fig. 1. On-chip Electromigration sign-off procedure. The process of waiving a design EM violation involves several considerations including the comparison to previously waived decision on how similar in geometry and design conditions](image)

This generic waiver system has been successfully implemented with EM violation waiver system (EMWaiver). EM violations are increasing in FinFET processes and becoming harder to evaluate and fix due to narrower width and complex geometries. Waiving or fixing violations in the EMWaiver is based on historical EM violations perspective which requires design team’s effort. The system emphasizes on must-fix EM violations leveraging previous similar violations. As an artifact, the system has been used on new FinFET designs to perform EMWaiver and demonstrated highly confident results. Also, the generic Waiver System behind EMWaiver can be applied on other EDA sign-off functions such as DvD (Dynamic Voltage Drop) Waiver where it is required to identify specific features to identify similar violations and train with the features preferential...
weights. In this paper, we focus on EMWaiver and illustrate the flow of how we apply machine learning and distributed computing in the system.

The rest of the paper is organized as follows. In Section II Waiver System Methodology is introduced, which uses EMWaiver as an example to demonstrate the usage. In Section III SeaScape distributed computing platform [2] is briefly shown, which accelerates machine learning based EMWaiver model training. Results of EMWaiver is shown in Section IV. Section V concludes this paper.

II. Waiver System Methodology

Our current generic waiver system uses clustering algorithms (e.g. K-means, K-nearest neighbor algorithms [3-6]) to identify similarity between violations. In this section, EMWaiver is demonstrated, as an example application for the violation waiver system. For a large GPU design, quite a few IP blocks need to run through IR, DvD, and EM analysis [7-10]. In particular, all the EM violations need to be examined by senior designers to either waive or fix them. The label of “waive/non-waive” for each previous violation in the history can be learned to recommend or predict the next EM violation in a new design. This EM Waiver system is used for the GPU designs in FinFET processes. It has been proven that using this waiver system is far less error-prone than manual methods. It also reduces the number of high-risk “must-fix” entries sifted through as waived entries.

A. Architecture of Machine Learning based EM Waiver System

As shown in Fig. 1, an EMWaiver uses historical design cases to train the ML-based inference model via clustering methods, which is used to predict violation for new design case; After predicting the new violation in the new design, feedback to EMWaiver is given to have incremental training (Fig. 2), where it uses a K-nearest-neighbor (KNN) algorithm [3, 4] to evaluate the similarity of one violation with other violations.

As discussed below, selected features would need to be determined for each violation waiver system. For EM violation waiver application, it uses several proprietary EDA tools’ features [1] to identify similar violations to be able to score risks associated with each violation and learn human decision from labeled training dataset. The following example features are chosen for EMWaiver and can be input from users

- Timestamp of EM violation data
- EM Limit
- Location of EM violation segment, and etc

Fig. 2. Machine learning based EMWaiver uses historical EM waiving experience on new designs which emphasizes must-fix EM violations leveraging previous similar violations. This system is far less error-prone than manual methods

Feature Normalization

For every feature identified, normalized values per case are derived in the form of Gaussian Distribution; This was necessary to generalize features for other cases, superimpose across all designs to have global z-normalized features (Fig. 3).

Weight matrix and distance metric

Not all features are created equal from user’s experience. Therefore, weight matrix is added during feature engineering process, which leads to a distance metric as follows. Distance $D(\vec{x}, \vec{y})$ is calculated between two violations $\vec{x}, \vec{y}$ as shown below.

$$D(\vec{x}, \vec{y}) = (\vec{x} - \vec{y})^T M (\vec{x} - \vec{y})$$

where $M = W^T W$. $M$ and $W$ are diagonal matrices. The diagonal element of $W$ is $w_i$, which is the weight for feature $i$.

B. Waiver Clustering via K-means algorithm

Intuitively, K-means algorithm finds a partition such that
the squared error between the empirical mean of a cluster and
the points in the cluster is minimized [5]. K-means method
in our current system is described in a high level as follows.

**Algorithm 1: K-means**

1. Select $K$ initial centroids for $K$ clusters; repeat steps 2 and
   3 until cluster membership stabilizes.
2. Generate a new partition by assigning each pattern to its
closest centroids.
3. Compute centroid of each new cluster.

The clusters are built inside our database (Fig. 4). Regarding
selection of the initial centroids, as we all know, the process
remains as an art. The techniques from the work by J. He, *et al.*
[11] can be applied. EMWaiver is not too sensitive to initial
centroid selection since we choose closest $M$ clusters for
KNN-based risk scoring given a new EM violation.

![Database of all violations](image)

**Fig. 4.** Clustering with K-means on the database of all violations. C1, C2, C3 are the cluster IDs. Each cluster contains violation entries annotated as `<C#V#>` while each entry contains multi-dimensional features.

**C. Inference using Trained Models**

Given an EM violation of a new design, the item will be
scored by the system, which is based on the distances to the
identified closest data entries inside nearby clusters. The risk
score can be computed via various methods. For example,
we can derive the score of violation $j$ with the softmax
function,

$$score(j) = \frac{1}{\sum_i e^{-\beta d_{ij}}} \sum_i e^{-\beta d_{ij}} score(i)$$

where item $i$ in the close-by violations in the trained model
inside nearby clusters. $d_{ij}$ is the distance between violation
$i$ and violation $j$. The parameter $\beta$ is a scaling constant.

A user-defined threshold $T$ is used to classify the item $j$ to
be waived or non-waived.

$\{\text{non - waive, } R(j) \geq T \}$
$\{\text{waive, } R(j) < T \}$

For stricter constraint from users, some post-processing
can be applied as well.

**D. Incremental Training**

The K-means training module on violation database and
inference model has been provisioned in a Client-Server
based Architecture over on-premises based model or can be
deployed to cloud computing as well (Fig. 5).

![Client-server architecture of EMWaiver](image)

**Fig. 5.** The client-server architecture of EMWaiver. A console has been created for design admin who can perform inference model generation.

After getting the feedback from the users on the inferencing
result, EMWaiver may encounter some violations which were
mis-predicted or totally new to the system. EMWaiver will
incrementally training the machine learning model by adding
those data items on the existing trained entries (clusters).
Furthermore, reasons are also annotated by designers on
waived entries including “unique use cases”, “minor violations”,
“proven low risk”, and “known tool bug”. Designers can also annotate reason for “non-waived” entries.

**III. MapReduce and Dataflow Computation Infrastructure on SeaScape**

**A. K-means Implementation using MapReduce on SeaScape**

The main clustering method, K-means, is used in the
violation waiver system. Since the number of violations in the
database can be quite large in millions, the performance of K-
means would need to be carefully designed and monitored.
The custom K-means is implemented using MapReduce and
dataflow [12-14] method on a customized SeaScape
architecture [2]. SeaScape is a distributed architecture using
mixed local disk for storage and in-memory database with
dataflow programming as shown in Fig. 5 and 6. This purpose
built architecture is used to build distributed scalable EDA
applications for IR/EM and Machine Learning applications.
Parallel clustering with centroid determination and distance
calculation is done in Mapper and result are accumulated in
Reducer in Fig. 7.

Fig. 5. SeaScape architecture [2] for distributed dataflow processing where distributed K-means is implemented on. (DS: distributed system)

Fig. 6. Dataflow programming focusing on the partition of data in hierarchical data flow stages and can be easily distributed and run in parallel.

Fig. 7. MapReduce flow with Mapper and Reducer functions are supported in SeaScape and K-means can be realized and parallelized in the architecture.

MapReduce based K-means clustering algorithm is investigated in the work [15]. In our current waiver system, the distributed K-means is described in the following.

- The data sets are divided evenly based on the number of MapReduce workers.
- Each worker holds the information of the centroids (either from Step 1 or Step 3 of Algorithm 1).

- The Map process maps the subsets of the dataset to different workers and run the Step 2 of Algorithm 1 with centroids.
- The Reduce operation collects the clusters from all the workers, computes centroids and check the convergence.

B. Weight Tuning for the Features in K-means Clustering

The clustering distance features also use unique weights to learn precedence of these features. These weights need to be fine-tuned. For the weight of feature, to the best of our knowledges, there is no efficient method to this end. We can simply create an objection function and use some general methods for optimizing the objection function. Common derivative-free methods can be applied here include downhill Simplex search, random search, simulated annealing, genetic algorithms, etc. In our distributed system, we easily run the machine learning algorithms in our computer servers and clusters with over hundred processors to search a good weight combination.

IV. EMWaiver Results and Analysis

Industrial dataset from Nvidia is used in this section to demonstrate the performance of the work. We utilized the SeaScape distributed machine learning infrastructure to fast train the data set as well as to perform data inference. Fig. 8-11 are the snapshots of EMWaiver with IP IR/EM tool GUI [1].

Fig. 8. Training EM violations from historical database
Besides, **confusion matrix** is used to check accuracy as below in Table II.

**TABLE II**

Confusion matrix is used to make sure minimum number of “False Negative” generated from the waiver system.

<table>
<thead>
<tr>
<th></th>
<th>Predict Non-Waive</th>
<th>Predict Waive</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actual Non-Waive</strong></td>
<td>True Positive (TP)</td>
<td>False Negative (FN)</td>
</tr>
<tr>
<td><strong>Actual Waive</strong></td>
<td>False Positive (FP)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>

Precision: \( P = \frac{TP}{TP + FP} \); Recall: \( R = \frac{TP}{TP + FN} \).

The category that needs to be minimized is “False Negative (FN)” which means the waived decision from the inference model on a new design should actually need to be fixed (non-waive). On the other way, “False Positive (FP)” may increase fixing time and chip area since the EM violation needs not be fixed. We run k-folds cross-validation (k=10) to evaluate our model, which has average precision 87% (predicted results compared to the actual label). The precision rate \( P \) is 79% and the recall rate \( R \) is 100%. The recall rate \( R \) is very important to us, where we need to keep FN to 0, even if we sacrifice the precision rate. However, the weight parameter tuning can be performed more exhaustively to search for higher precision and save more fixing time in our case.

**TABLE III**

Confusion matrix for Nvidia EMWaiver case

<table>
<thead>
<tr>
<th></th>
<th>Predict Non-Waive</th>
<th>Predict Waive</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actual Non-Waive</strong></td>
<td>49%</td>
<td>0%</td>
</tr>
<tr>
<td><strong>Actual Waive</strong></td>
<td>13%</td>
<td>38%</td>
</tr>
</tbody>
</table>

The runtime performance of model training of Nvidia case is shown in Table I. The communication, work balance, and sequential parts of the code (the white gap between the colored blocks-distributed computing jobs, in Fig. 12.) become more dominant when increasing the number of workers.

**TABLE I**

Runtime Performance of Training in EMWaiver for Nvidia case (#entry 351K of dataset)

<table>
<thead>
<tr>
<th>#Workers</th>
<th>1</th>
<th>4</th>
<th>16</th>
<th>64</th>
<th>256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runtime (s)</td>
<td>5438</td>
<td>2891</td>
<td>972</td>
<td>452</td>
<td>286</td>
</tr>
<tr>
<td>Speedups</td>
<td>-</td>
<td>1.9X</td>
<td>5.6X</td>
<td>12.0X</td>
<td>19.0X</td>
</tr>
</tbody>
</table>

V. Summary and Future Work

It is currently painful for designers to waive EM violations manually for every tape-out and can be error-prone. Using
historical data to learn this behavior significantly automates or assists this process by applying ML techniques presented in this paper for an EM waiver system, the process can be significantly automated. This EM Waiver System allows a design team to minimize resources for EM waiving decision, speed up tape-out, eliminate errors and minimize risk. Also, this system incrementally learns/re-trains with new design and simulation data for better prediction of future violations.

Though this system is currently customized for EM waiving application, it can be generalized and extended to other applications/violation waivers as well such as voltage drop. The underlying machine learning techniques help achieve this goal and also provides flexibility to learn in different process nodes.

For incremental learning and to capture feedback from user on predicted scores, a reinforcement based learning can be used by rewarding positive or negative feedbacks by users, especially for weight tuning and risk score threshold setting.

Acknowledgements

We would like to thank Nvidia designers providing insightful experience on the reasons for waiving or fixing EM violations. Thanks to M. Vusirikala, S. Kim, D. Zhu, L. Yin and A. Odabasi for Waiver and SeaScape discussion. Also thanks to Tsai-Jung Chung for GUI implementation help. Also thanks to helpful discussion with Prof. Roger Jang at National Taiwan University.

References