Predicting Variability in Nanoscale Lithography Processes

Dragoljub Gagi Drmanac
University of California,
Santa Barbara

Frank Liu
IBM
Austin Research Lab

Li-C. Wang
University of California,
Santa Barbara

ABSTRACT

As lithography process nodes shrink to sub-wavelength levels generating acceptable layout patterns becomes a challenging problem. Traditionally, complex convolution based lithography simulations are used to estimate areas of high variability. These methods are slow and infeasible for large scale full chip analysis. This work proposes a solution to this problem by using machine learning techniques to identify layout areas that are more prone to variability. A novel target layout representation is proposed, and the latest support vector machine (SVM) algorithms are used to detect variability within standard cells and between cells in a simulated full chip layout.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids; I.4.7 [**Feature Measurement**]: Feature representation

General Terms

Algorithms, Performance, Reliability, Verification

Keywords

Photo Lithography, Process Variation, Modeling Variability, Machine Learning, Kernel Methods

1. INTRODUCTION

As IC process nodes continue to shrink from 65 nm to 45nm and below, generating acceptable layout patterns becomes an increasingly difficult problem. Industry leading foundries continue to depend on 193nm lithography systems to print the latest 45nm designs. Sub-wavelength photolithography results in unavoidable manufacturability issues due to process variations and unforeseen cell-to-cell interactions. To combat these problems, several resolution enhancement techniques (RET) have been employed such as optical proximity correction (OPC) and phase shift masks (PSM); however, these methods often rely on complex simulation models and are usually applied late in the design cycle when only minor layout changes can be made. Lithography simulation is the current golden standard for understanding variability of new process nodes and cell designs, yet it is prohibitively time consuming for full chip analysis. A first order layout analysis approach is needed to quickly detect areas of high variability within standard cells, and between adjacent cells in full chip placements.

Dealing with variability early in the design process helps prevent manufacturability issues, yet very few tools exist that can choose between acceptable designs to minimize variability. Several model-based design for manufacturability (DFM) methodologies use variability analysis to manage complex process-design interactions, however none of them directly predict variability. For example, simulating standard cells in various configurations to measure context based variability, or performing litho-aware electrical current calculations to extract transistor parameters for accurate timing analysis [9]. The aforementioned DFM methods could all benefit from a direct variability prediction tool, to avoid the complications of lithography simulation and process parameter selection.

Understanding and predicting sources of variability is a ubiquitous problem that has applications in various forms of DFM. Most work in this area has attempted to combine variability metrics or models with existing DFM frameworks. For example, pattern matching techniques have been used in conjunction with standard design rule checking (DRC) software to identify layout configurations that are difficult to manufacture in the presence of lithographic process variation [4]. While this helps DRC software capture some problematic 2D geometries, it relies on manual pattern selection based on slow lithography simulation and human judgment.

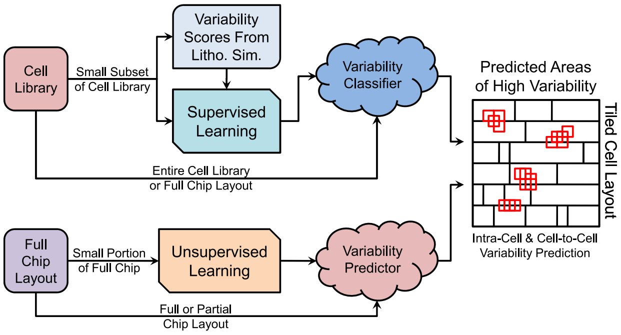

Cell Library
Small Subset of Cell Library
Variability Scores From Litho. Sim.
Supervised Learning
Variability Classifier
Entire Cell Library or Full Chip Layout
Predicted Areas of High Variability
Tiled Cell Layout
Intra-Cell & Cell-to-Cell Variability Prediction
Full Chip Layout
Small Portion of Full Chip
Unsupervised Learning
Variability Predictor
Full or Partial Chip Layout

Figure 1: Variability Prediction Overview

This work is supported in part by National Science Foundation, Grant No. 0541192 and Semiconductor Research Corporation, project task 1585.001.

DAC'09, July 26-31, 2009, San Francisco, California, USA
Copyright 2009 ACM 978-1-60558-497-3/09/07....10.00

Improvements to standard OPC techniques have also been made by incorporating analytical process variation models that account for fluctuations in dosage and focus [10]. While this improves OPC's awareness to variability issues, it does so at a cost of two to three times traditional OPC runtime. Most existing DFM tools work to improve printability of placed and routed layouts, however many key problems are generated early in the design cycle when details about the manufacturing process are unavailable [6].

This paper proposes a novel machine learning framework for quickly predicting areas of high variability within standard cells, and between cells in full chip layouts. Figure 1 shows an overview of two proposed machine learning approaches for predicting variability. The first method builds a variability classifier using training samples labeled with lithography simulation scores from a subset of cells. The second approach learns to predict variability by analyzing a small layout portion without simulation. Both methods can predict variability in standard cell libraries or across full chip layouts. The rest of the paper is organized as follows. Section 2 will go over measuring variability using lithography simulation. Section 3 will describe our novel layout representation that makes variability prediction possible. Section 4 will outline the two variability prediction methods in detail, and section 5 will discuss the experiments and results. Section 6 will conclude the paper followed by references in section 7.

2. MEASURING VARIABILITY

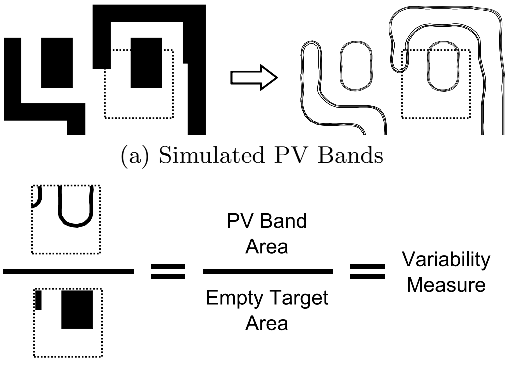
In this work we focus on predicting variability in printed 45nm metal layers with respect to changes in dosage and focus. To quantify variability, an industry leading lithography simulator is used in conjunction with a defined variability metric to guide and validate our predictions.

2.1 Lithography Simulation

Lithography simulation is currently the most common approach for predicting intra-cell and cell-to-cell variability. We use a lithography simulator to create process variation (PV) bands showing the possible areas within which a given metal layer will print, as dosage and focus conditions vary. Figure 2(a) shows a metal target layer and its corresponding PV bands. Notice how the width of the bands changes depending on different neighboring geometries; the thicker the band, the more uncertainty there is in the final edge placement.

2.2 Variability Metric

Lithography simulators commonly scan PV bands and assign variability scores to layout areas according to a simple metric. Usually, the metric is a measure of PV band area normalized by the window under which the variability calculation is performed. In this work we introduce an analogous variability measure, the PV band area normalized by the empty target area. Figure 2(b) shows how this calculation is performed. First a square window is selected to perform the calculation under, then the PV band area within the window is calculated and normalized by the area within the window not covered by metal layer polygons. We normalize by the empty target area to ensure that variability in denser layout regions is treated as more critical. For a fair comparison of variability we purposely chose a metric similar to existing industry standards.



(a) Simulated PV Bands

(b) Variability Metric

Figure 2: Calculating the Variability Measure

3. NOVEL LAYOUT REPRESENTATION

The graphic data system (GDSII) file format is the industry standard for storing and transferring large layout designs, however its representation is limited since entire layouts are saved as a set of planar geometric shapes. To perform variability prediction we propose an intermediate layout representation that captures relative shape and position information, called the histogram distance transform (HDT). Concepts from image processing and feature encoding are discussed to introduce this novel representation.

3.1 Image Processing

Since variability changes based on configurations in a two dimensional space, image processing techniques can help decompose target layouts, and encode them such that efficient variability prediction is possible. In this section we discuss two bitmap image formats, image histograms, the distance transform, and raster scanning.

3.1.1 Bitmap Image Representations

All target layouts, PV bands, and intermediate transformations are represented using simple bitmap images. The simplest image format we use is called the portable bitmap (PBM) which is just an array of 1's and 0's representing black and white pixels. This image format was selected to represent our target layouts because of its simplicity and software compatibility. To store results of image transforms we used a grayscale image format called portable gray map (PGM) which is similar to PBM except each pixel is an integer from 0-255 representing the 256 grayscale levels. Since we do not store intermediate images during processing our methodology does not require a large memory or compressed image formats.

3.1.2 Image Histogram

An image histogram is a compact representation that captures the distribution of intensity levels present within a grayscale image. In PGM images there are 256 grayscale levels so the image histogram has 256 bins on the x-axis and displays the number of pixels per bin on the y-axis. An example grayscale image and its corresponding histogram are shown in Figure 3. Usually, one grayscale value dominates the image so to get a more normalized result we take the log of the number of pixels for each intensity value. This is an abstract translation and rotation invariant representation that captures a great deal of information content [3].

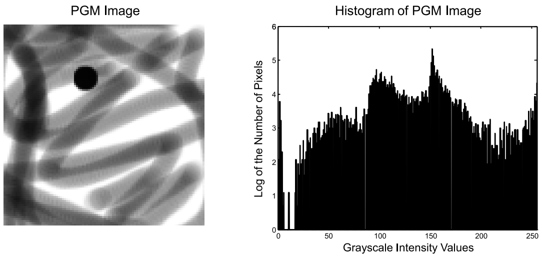
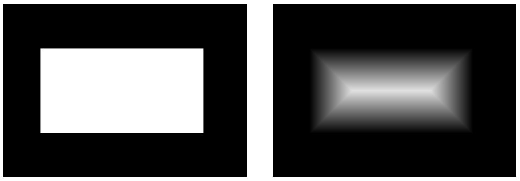


Figure 3: Image Histogram

3.1.3 Distance Transform

The distance transform is an image processing technique that converts a black and white bitmap image into a grayscale image by replacing each white pixel with the distance to the nearest black/white pixel boundary. Since computed distances can be large, each pixel is renormalized to fit in the 256 grayscale range. An example binary image and its distance transform are shown in Figure 4. The distance transform has converted the white rectangle into a grayscale image where each pixel value represents its distance to the nearest boundary. In this simple example the boundary is the perimeter of the white rectangle.

Performing the distance transform can be seen as an encoding of shape information for a given binary image. After the transform is performed a more descriptive image results that captures overall shape, length, and width [5]. The distance transform also approximates the skeleton of a binary shape, which shows up as the bright pixel areas similar to an x-ray. We use an efficient Euclidian distance transform algorithm proposed in [5].



(a) PBM Image (b) Distance Transform

Figure 4: Distance Transformed Rectangle

3.1.4 Raster Scanning

Since layouts are often large compared to the local features contributing to variability, it is important to raster scan layouts to extract local features. Our raster scanning procedure is shown in Figure 5. We start with a 100x100 pixel window and move it around the target layout capturing samples with a step size of 50 pixels. This gives us a 50% overlap between raster windows and ensures good coverage of each cell. The scanned images are scaled so each pixel corresponds to $3nm^2$ on the actual target layout. This resolution is fine enough to capture detailed spatial information without slowing the system down with unnecessary data processing.

3.2 Encoding Layout Features

Figure 6 shows the procedure for transforming a GDSII layout into our novel HDT representation. We extract the target outline from a GDSII file and save it as a binary PBM image, next we compute the distance transform of the target outline and obtain a grayscale PGM image, finally we take the histogram of a small raster scanned portion of the resulting image as our target layout representation for that small area. The distance transformed layout is fully raster scanned; with each step of the raster window a histogram is computed and stored as a 256 dimensional vector. After the transformation is complete the target layout is represented as a set of histogram vectors.

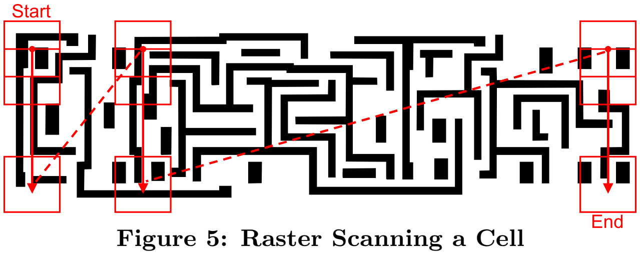


Figure 5: Raster Scanning a Cell

The distance transform of a target layout captures the distribution of spacing between polygons. Bright areas correspond to spread out polygons and dark areas correspond to tightly packed polygons. Notice that polygon shape is also captured by the transform computed within the target outlines. Fine skeleton structures appear encoding the shape and size of the original layout. Although this is an abstract representation of relative shape and spacing, small layout areas can accurately be represented in this way. The distance transform captures local shape and spacing information, while the histogram gives us an efficient means of storing and working with that information.

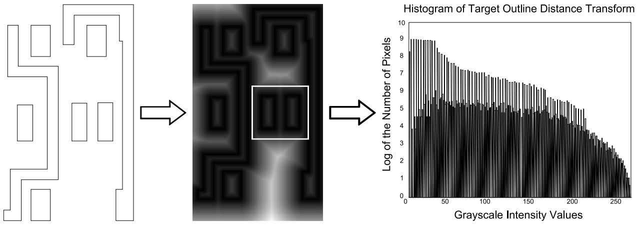

Histogram of Target Outline Distance Transform
Log of the Number of Pixels
Grayscale Intensity Values
0
1
2
3
4
5
6
7
8
9
10
0
50
100
150
200
250

Figure 6: Novel Target Layout Representation

4. PREDICTING VARIABILITY

Variability prediction is performed using SVM learning algorithms. In machine learning theory there are two fundamental learning approaches, supervised and unsupervised. Supervised learning estimates a function from a set of labeled training examples, to solve regression or classification problems, while unsupervised learning estimates the support of the training data without labels, to solve clustering or outlier detection problems. In this work we use two-class and one-class nu support vector classifier (ν-svc) algorithms proposed by [7] and [8], and a modified version of the open source LibSVM software package implemented by [2].

4.1 Layout Similarity Measure

To use our HDT layout representation in a machine learning framework we define a function to measure similarity between two raster windows. A kernel function $K(\vec{x}, \vec{y})$ computes the similarity between two sample vectors. Since our target layout representation is histogram based we use the histogram intersection kernel shown below, where $\vec{x}$ and $\vec{y}$ are 256 dimensional histogram vectors.

$$K(\vec{x}, \vec{y}) = \sum_{i=1}^{256} min(x_i, y_i)$$

This kernel is efficient because it calculates the intersection of each histogram bin without time consuming multiplication or exponentiation. Intuitively, we can see that the larger the intersection between two histograms the more similar they are. The histogram intersection kernel has been shown to perform well in image recognition tasks by [1].

4.2 Two-Class Supervised Learning

Two-class SVM analysis generates a predictive classifier that detects areas of high variability based on labeled training samples picked from a standard cell library. A training sample is a 100×100 pixel layout window encoded using our HDT layout representation. Each window is assigned a variability score calculated from PV bands using the variability metric previously defined. Since ν-SVC performs pass/fail classification, a variability score threshold is established to label high variability areas as failing and low variability areas as passing. Each sample is constructed as a training-label histogram-vector pair $(l_i, \vec{s_i})$, where $\vec{s_i} = (h_1, h_2, ..., h_{256})$ contains 256 histogram components $h_1 - h_{256}$, encoded using our HDT representation, and a pass/fail classification label l_i, determined by variability score thresholding.

To train a variability prediction model the ν-SVC soft margin classifier is used to separate high and low variability samples with maximum margin in a 256 dimensional feature space. Since the classification space is large many separating hyperplanes exist, so optimization is used to find one that maximally separates the training data. A maximally separating hyperplane ensures that our classifier will be general enough to correctly label unseen samples as having low or high variability. The problem is formulated as the following quadratic programming optimization.

$$Maximize \quad -\frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j l_i l_j K(\vec{s_i}, \vec{s_j})$$

$$Subject\ to \quad 0 \le \alpha_i \le \frac{1}{m},\ \sum_{i=1}^{m} \alpha_i l_i = 0,\ \sum_{i=1}^{m} \alpha_i \ge \nu.$$

Where $\alpha_{i,j}$ are Lagrangian multipliers, $\vec{s_{i,j}}$ are training samples, $l_{i,j} \in \{\pm 1\}$ are pass/fail training labels, ν is the fraction of accepted training errors, and $K(\vec{s_i}, \vec{s_j})$ is the histogram intersection kernel. In general, training samples may not be linearly separable so ν is used to control the fraction of misclassified samples allowed when establishing a boundary, namely how soft the margin will be. When optimization is complete, some Lagrangian multipliers are zero, indicating that their corresponding training samples play no role in establishing the classification boundary. Samples with nonzero Lagrangian multipliers influence the boundary and are called support vectors. The resulting two-class SVM classifier consists of a decision function $f(\vec{x})$ used to classify new samples, and set of support vectors defining the classification boundary. The decision function is shown below.

$$f(\vec{x}) = sgn\left(\sum_{i=1}^{m} \alpha_i l_i K(\vec{x}, \vec{s_i}) + b \right)$$

$f(\vec{x})$ is a weighted sum of similarity between all support vectors and the current sample being classified. The histogram intersection kernel $K(\vec{x}, \vec{s_i})$ compares new samples to the set of support vectors, so that a pass/fail decision can be made.

4.3 One-Class Unsupervised Learning

One-class SVM can be viewed as an outlier analysis algorithm that places the ultimate outlier at the origin. Samples far from the origin are labeled as normal and samples near the origin as abnormal. In the context of our problem and the HDT layout representation, the origin represents layouts that have uniform shape and spacing. When raster scanning metal layers, uniform shape and spacing most often occurs in areas with tightly packed parallel wires. In these areas, the critical dimension of processing is being pushed to the limit, and surrounding geometries substantially influence parallel wire widths, so we expect these areas to exhibit variability.

Outlier analysis is performed using the one-class ν-SVC algorithm. Each raster scanned window is constructed as a sample vector $\vec{s_i} = (h_1, h_2, ..., h_{256})$. In this case our sample has no training label attached to it since the algorithm assumes all samples are of the same class, and the origin is the only sample of the outlier class. The one-class algorithm maximally separate samples from the origin using the following quadratic programming optimization.

$$Minimize \quad \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j K(\vec{s_i}, \vec{s_j})$$

$$Subject\ to \quad 0 \le \alpha_i \le \frac{1}{\nu m},\ \sum_{i=1}^{m} \alpha_i = 1.$$

Where $\alpha_{i,j}$ are Lagrangian multipliers, $\vec{s_{i,j}}$ are training samples, ν is the fraction of samples we expect to have high variability, and $K(\vec{s_i}, \vec{s_j})$ is the histogram intersection kernel. This is similar to the previous optimization problem, except it does not depend on any training labels. After optimization is complete, samples with nonzero Lagrangian multipliers are support vectors and define the classification boundary. Once the boundary is established the outlier measure for each sample can be computed using the function $g(\vec{x})$ shown below.

$$g(\vec{x}) = \sum_{i=1}^{m} \alpha_i K(\vec{x}, \vec{s_i}) - \rho$$

In general, there will be many outliers of various degrees, so it is important to perform an outlier ranking to examine the worst outliers. $g(\vec{x})$ returns positive and negative numbers, where positive numbers correspond to normal samples and negative numbers correspond to outliers. Using $g(\vec{x})$ it is possible to rank-select the top 500 outlier windows and draw them on the target layout giving a clear picture of where variability occurs.

5. EXPERIMENTS AND RESULTS

Variability analysis was performed on 45nm metal layers extracted from 22 standard cells, and a large random tiling of those cells. Analysis performed on individual cells, from which more complicated layouts can be constructed, is called cell-based, while analysis performed on a tiling of cells is called chip-based. Both cell-based and chip-based analysis was performed using two-class and one-class algorithms.

First, a 22 cell library was analyzed using both methods to ensure prediction accuracy, then a large layout based on a random cell tiling was analyzed to measure variability prediction accuracy on a realistic scale.

5.1 Cell Library Analysis

The first cell based experiment examined how accurately a classifier trained with 11 of 22 simulated cells could correctly predict variability in the unseen half. To verify prediction accuracy all 22 cells were simulated and pass/fail variability labels were calculated. Overall prediction accuracy was determined by comparing simulation and classification results. To train the classifier 11 large cells were scanned and generated 3960 training samples. To validate the classifier 11 smaller cells were scanned and generated 1584 testing samples. Larger cells generating more samples were used for training so that a generally representative variability classifier could be built. The second cell-based approach directly applied one-class outlier analysis on all 22 cells to compare how accurately it could classify samples relative to simulation results. In this case, training labels were not needed for analysis but only used to compute the classification accuracy.

Table 1 shows the classification accuracy for cell-based analysis using both one-class and two-class methods. Two-class SVM analysis achieved a 92.4% classification accuracy on unseen cells, and one-class outlier analysis correctly classified 85.1% of all cells without simulating to generate variability scores. The one-class method classified low and high variability samples with the same 85% accuracy, while the two-class method more accurately classified low variability samples than high variability samples. This is expected, since the two-class method was trained with more low variability samples, it learned a better concept of low variability. In one-class analysis no training labels were used, so the learned model was not biased toward predicting low or high variability.

Learning Method	Variability Prediction Accuracy		
	Total	Low Var.	High Var.
2-Class	92.4%	95.0%	86.5%
1-Class	85.1%	85.2%	84.8%

Table 1: Cell-Based Prediction Accuracy

5.2 Full Chip Analysis

Full chip analysis was performed in two ways; either by binary classification, or by one-class SVM outlier analysis. The advantage of one-class analysis is that it is an unsupervised method capable of classifying samples without depending on lithography simulation for training labels. The disadvantage is that the only way to introduce prior knowledge about variability is through your data representation and kernel function. According to features encoded by our HDT layout representation, the one-class outlier analysis method will identify abnormal layout areas that are likely to exhibit high variability, while the two-class method will identify layout areas that are similar to previously seen high variability samples.

The tiled layout shown in Figure 7 was analyzed using a two-class variability predictor trained with the full 22 cell library, as well as a one-class outlier model trained with samples from the scanned tiled cell layout. Raster scanning the layout generated 50,000 samples that needed to be analyzed and validated. Since the constructed tiling was very large it was time consuming to fully simulate, so three predicted clusters of high variability were selected for validation by lithography simulation. These cell clusters are thickly outlined in Figure 7 and labeled Tiled Simulations 1-3.

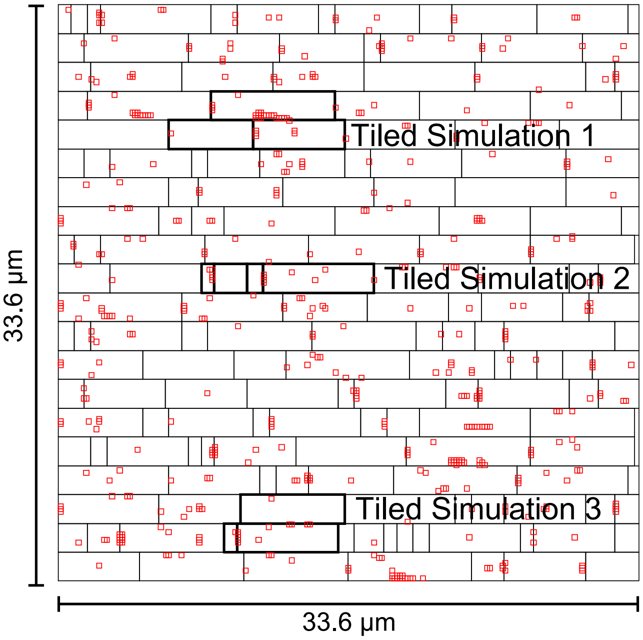
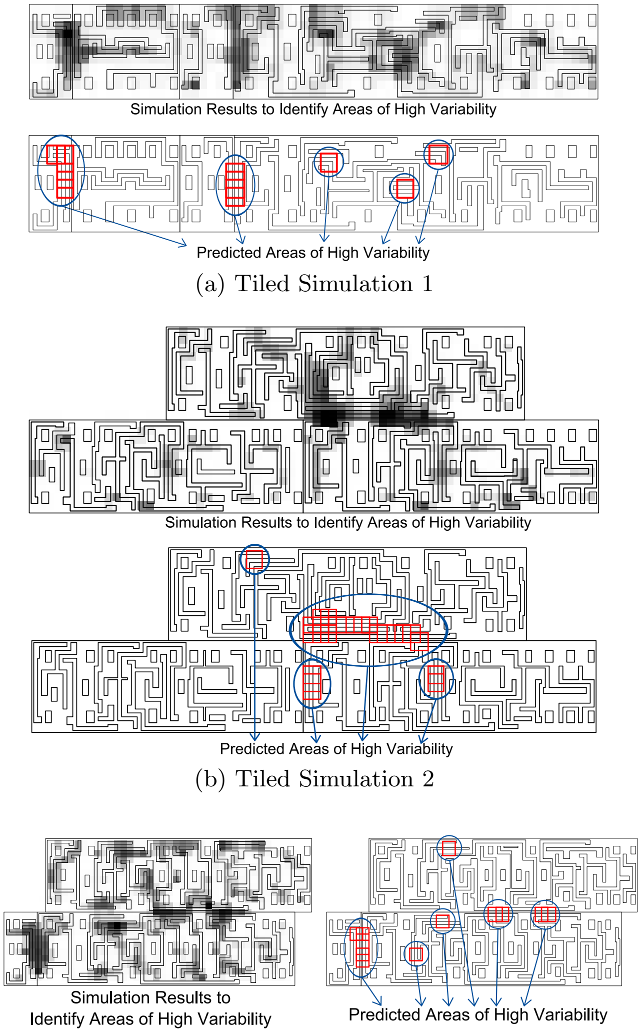


Figure 7: Predicted Variability on a Chip Layout

Two-class SVM analysis trained a high/low variability predictor based on lithography simulation of a cell library, and subsequently used that predictor to perform full layout analysis. Figure 7 show the top 500 areas of high variability detected by the binary classifier marked with small red squares. High variability areas are detected within and between standard cells showing that the model can generalize well and detect variability caused by unforeseen cell-to-cell interactions. One-class outlier analysis was performed on the same tiled layout and the top 500 areas of high variability were found to be very similar to those predicted in Figure 7 indicating that our two variability prediction approaches are consistent.

To show full-chip variability prediction is accurate, we chose three highlighted groups of adjacent cells in Figure 7, to verify by lithography simulation. After simulation, a grayscale map corresponding to the variability scores was printed across each set of adjacent cells. Darker areas correspond to higher variability, and lighter areas indicate lower variability. Figure 8 shows high variability areas predicted by lithographic simulation as well as areas of high variability predicted by our two-class full chip method. Across all three simulations it is clear that the darkest areas corresponding to high variability are well predicted by our SVM models. Moreover, we see that accurate variability prediction is possible between adjacent cells, without explicit enumeration of cell-to-cell interaction contexts.


Simulation Results to Identify Areas of High Variability
Predicted Areas of High Variability
Simulation Results to Identify Areas of High Variability
Predicted Areas of High Variability
Simulation Results to Identify Areas of High Variability
Predicted Areas of High Variability

(a) Tiled Simulation 1

(b) Tiled Simulation 2

(c) Tiled Simulation 3

Figure 8: Simulated vs. Predicted Variability

5.3 Performance

Table 2 summarizes the runtime of SVM variability prediction compared to conventional lithography simulation. Both cell-based models ran quickly, taking on average 30 seconds to train, and 6 seconds to classify the full cell library. This speed can be attributed to the small number of support vector comparisons required to determine the class of a sample. Chip-based analysis was slower due to the larger area being scanned and analyzed. Two-Class chip-based analysis took 249 seconds to complete, since training only required scanning the small cell library. One-class chip-based analysis took 1308 seconds to analyze all 50,000 layout samples. This was a much slower analysis approach since it trained with more samples and classification depended on more support vectors. Classification can be accelerated by splitting test data into small subsets and using multiple processors to classify each subset in parallel. In comparison, lithography simulation took 1.5 hours to analyze the cell library, and approximately 33 hours to analyze the tiled cell array, which is 70-90x slower than SVM classification. All analysis was performed on a 3 GHz Core 2 Quad running Linux.

Method	Raster	Train	Predict	Total
2-Class Cell	18 s	48 s	5 s	71 s
1-Class Cell	18 s	11 s	6 s	35 s
Litho. Cell	-	-	1.5 h	1.5 h
2-Class Chip	165 s	48 s	36 s	249 s
1-Class Chip	165 s	865 s	278 s	1308 s
Litho. Chip	-	-	33 h*	33 h*

*Based on 10 minute per-cell average runtime.

Table 2: Runtime Comparison

6. CONCLUSION

This paper introduced a new machine learning methodology for detecting areas of high variability within and between standard cells in full chip layouts. A novel HDT layout representation was used in conjunction with the latest SVM learning techniques to correctly predict areas of high variability according to a defined variability metric. Several experiments were performed and validated by an industry leading lithography simulator, showing that intra-cell and cell-to-cell variability could be predicted with high accuracy. Runtime of variability prediction was shown to be much faster than that of conventional lithography simulation, allowing high-speed first-order variability prediction.

7. REFERENCES

[1] S. Boughorbel, J. Tarel, and N. Boujemaa. Generalized histogram intersection kernel for image recognition. In *International Conference on Image Processing*, 2005.

[2] C. Chang and C. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[3] O. Chapelle, P. Haffner, and V. N. Vapnik. Support vector machines for histogram-based image classification. *Transactions on Neural Networks*, 1999.

[4] V. Dai, J. Yang, N. Rodriguez, and L. Capodieci. DRC Plus: Augmenting Standard DRC with Pattern Matching on 2D Geometries. In *SPIE*, 2007.

[5] P. F. Felzenswalb and D. P. Huttenlocher. Distance transforms of sampled functions. *Cornell Computing and Information Science Technical Report*, 2004.

[6] J. C. Rey, N. Nagaraj, A. Kahng, R. Aitken, L. Capodieci, F. Klass, C. Hou, and V. Singh. DFM in Practice: Hit or Hype? In *Design Automation Conference*, 2008.

[7] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 2001.

[8] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *Neural Computation*, 2000.

[9] N. Verghese, R. Rouse, and P. Hurat. Predictive Models and CAD Methodology for Pattern Dependent Variability. In *Design Automation Conference*, 2008.

[10] P. Yu, S. X. Shi, and D. Z. Pan. Process Variation Aware OPC with Variational Lithography Modeling. In *Design Automation Conference*, 2006.