ICCAD-2016 CAD Contest in Pattern Classification for Integrated Circuit Design Space Analysis and Benchmark Suite

(Invited Paper)

Rasit O. Topaloglu
IBM, Hopewell Junction, NY
rasit@us.ibm.com

ABSTRACT

Layout pattern classification has been utilized in recent years in integrated circuit design towards various goals such as design space analysis, design rule generation, and systematic yield optimization. There is a need for open source or academic solutions as very limited vendors are available to provide this functionality. Speed and accuracy are key aspects to target in the solutions. Given a circuit layout and various markers, contestants are asked to provide a reduced set of representative layout clips around these markers. Each such representative clip identifies a class and has an associated set of one or more unique layout markers.

1. BACKGROUND

Integrated circuit design requires polygons that will be printed on wafers to be drawn first in layout. A layout thus contains all polygons for all features that will eventually be present on a wafer.[1] Whenever a defect is found on wafer, a design space analysis may be conducted to find relevant layout clips of interest. Often times, these clips are similar to each other and replicated throughout the layout. It then becomes useful to categorize these clips and cluster ones that are similar to each other. This step is called layout pattern classification, or pattern classification in short. It serves to make the number of patterns to analyze manageable. This paper proposes the pattern classification problem and provides a link to a benchmark suite consisting of layouts, necessary inputs, and the outputs.

While pattern classification has been utilized in literature, there has not been a standard problem definition nor a common set of benchmarks. Some usages relevant to pattern classification are as follows: [2] provides an overall flow for pattern classification. [3] utilizes wavelet analysis to identify high and low frequency components, where the latter is used as an approximation.

We have used layout pattern classification and pattern matching towards determining and optimizing electrical performance caused by stress or local layout effects [4]. Authors of [5] and [6] have utilized machine learning methods towards classification and hotspot detection. [7] focuses on developing a metric that can be used for hotspot classification. Wuu et al. [8] utilize support vector machines towards layout pattern classification. Wu et al. [9] implement improvements to pattern classification in order to classify patterns without interference from fills for an EUV-specific technology.

A number of work in the literature has focused on the application aspects of design flows utilizing layout patterns. Xu et al. [10] and Chang at al. [11] utilize patterns in design for manufacturability (DFM) optimization. Gennari et al. [12] have linked TCAD to EDA using a pattern-based system.

While layout pattern classification is a specialized problem to VLSI, earlier work in the literature focus on more general aspects of data or image clustering. For example, Wright et al. [13] implement facial recognition using sparse representations. Lu et al. [14] implement neural networks to classify text. Aharon et al. [15] implement a k-dimensional singular value decomposition technique for sparse representation of data. Kanungo et al. [16] implement a k-means based clustering algorithm. Comaniciu et al. [17] conduct a feature space analysis using a mean shift methodology.

2. INPUT TO REQUESTED TOOL

- Layout in GDS format
- Markers (polygons) to indicate hotspots on this layout.

The markers are placed on a different layer than the design layer. For example design features could be present on layer 10 in the GDS, whereas the markers would all be present in layer 11. Marker sizes and shapes do not have to be fixed within the same layout.

Markers also indicate where a clip would be centered. Clip is a set of polygons that is extracted from the original design; an example is shown in Figure 1. The center point of a clip can fall anywhere within or on the edges of a marker polygon. For most practical cases, marker sizes will be picked small. Assuming the center point for a marker is also the clip center

[1] Certain features, such as fills, may be added later on in data preparation stages before a mask is written. Readers may refer to [1] for an overview of fill insertion from a previous ICCAD Contest.

ICCAD '16, November 07 - 10, 2016, Austin, TX, USA

ISBN 978-1-4503-4466-1/16/11...$15.00

DOI: http://dx.doi.org/10.1145/2966986.2980073

point could be a reasonable assumption. A square that is of width and height equal to the minimum width allowed in that layer of layout could be a practical marker choice. Markers do not have to intersect any design feature on the layout, i.e., they can be placed on space between features.

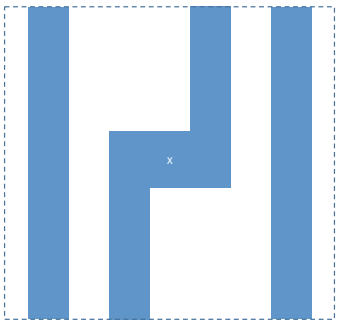

Figure 1: An example layout clip. Three interconnects are shown, with the center one utilizing a routing track jumper. X indicates the center point of the clip, the boundary of which is shown with the dashed border.

3. PARAMETERS OF REQUESTED TOOL

- Clip size of interest
- Area match constraint (a)
- Edge displacement constraint (e)

In addition to the inputs of the problem, a set of parameters are also used. These would correspond to the parameters a user of your pattern classification EDA tool could select. For the same GDS and marker inputs, multiple different parameters may be asked in different runs of the tool. With different input parameters, the output will be different. In regular usage, a user of your tool would be setting these parameters. The default parameters of a=1 and e=0nm corresponds to finding exact match for patterns.

Clip size is the size of a rectangular or square window around a clip, e.g., the dashed border in Figure 1. It is given as an (w, h) pair, the former indicating the width of the window and the latter indicating the height of the window in nanometers. If h is selected equal to w, a square window is implied.

Your pattern classification EDA tool will support two main classification setups. These are

1. area-constrained clustering,
2. edge-constrained clustering

A cluster in this context is a group of clips that resemble each other. The resemblance is determined according to the area-constrained and edge-constrained criteria. Furthermore, mirrored images can also be clustered together.

In area constrained clustering (**ACC**), an area match constraint parameter is provided. This parameter indicates the maximum area match ratio allowed between any clip of a cluster and the representative clip of a cluster.

Let us define a clips R and call it a representative clip of a cluster. Clip R does not have to be present in the design but all other clips that are to be a member of this cluster can be obtained by modifying the representative clip according to a constraint.

The ACC requires that $[Area(Xor(R, S))]/w.h \leq (1 - a)$ for a clip S to be considered in the same cluster as R. Here, the $Area()$ function takes in a set of polygons defined in the geometric space R^2 and outputs the total area of the polygons. $Xor(R, S)$ is a geometric *exclusive OR* operation that is applied across two clips, it takes two sets of polygons defined in R^2 as input and returns a set of polygons that shows the geometric difference between the two input sets. Figure 2 shows an example.

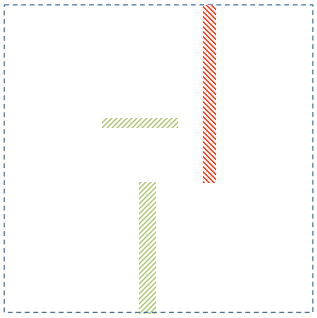

Figure 2: Xor function demonstrated. Assuming inputs to the Xor function are the set of polygons from Figure 1 and the set from Figure 3, the output is shown in this figure.

Parameter a is a real number and can be set anywhere in between and including 0 and 1, however for most practical cases, it will be set to close to 1. Setting it to 1 would indicate an exact area match.

In the edge-constrained clustering (**ECC**), e is a parameter given in nanometers and indicates by how much a given edge can shift inward or outward. Multiple edges can shift by varying amounts as long as each shift is $\leq e$. Edges can only shift with a Manhattan grid restriction, i.e., with orthogonal projections. Polygon edges connected to a shifted edge gets projected so that no gaps will exist. ECC operation is illustrated in Figure 3. Any clip of a cluster should satisfy ECC constraints with respect to the representative clip of a cluster.

Your tool is asked to run in either ACC or ECC mode, but not both at the same time.

Parameter e is a non-negative real number. For practical cases, it will be set close to 0. Setting it to 0 would indicate an exact edge match.

4. OUTPUT OF REQUESTED TOOL

- Clip border overlay file in GDS

Your pattern classification EDA tool needs to extract one clip covering each marker, and find the minimum number of clusters. The clusters will be named by a number starting from 1, i.e., cluster1, 2, 3, etc. Clusters are ranked by the number of clips they represent and the largest one becomes

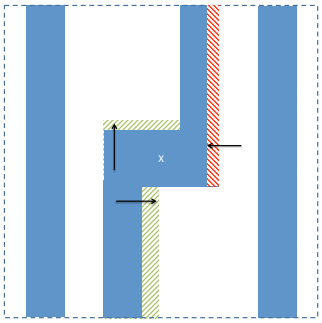

Figure 3: ECC demonstrated. Two edges shown to be extruded outward, increasing the area of original polygons. Both horizontal and vertical edge displacements shown. One edge is displaced inward, reducing the area of original polygon.

cluster1. Each cluster contains one or more clips. The clip border overlay file contains all the clip borders as rectangles or squares as specified by the user. All member clip borders of a given cluster will be printed on the same layer. Cluster1 clip borders will be printed on layer 1, cluster2 on 2, and so on.

5. SCORING OF SOLUTION QUALITY

To validate your tool output, the following are recommended. Some of these checks can be done by visual inspection of your output GDS files by opening it in a layout viewer.

1. Check that each input marker is covered in a cluster. In the clip border overlay file, a marker should be covered by one clip border.
2. Check that a marker is not listed in more than one cluster.
3. Cluster clip counts, when summed, should equal total number of markers in the layout.

Coverage metric constitutes 51% of score and is defined as follows:

Let us define three sets: Let S_{11} be pairs that are in the same cluster in reference vs. submitted solution, S_{10} be pairs in same cluster in reference solution but not in submitted solution, S_{01} be pairs in same cluster in submitted solution but not in reference solution.

Let us define $n_{xy} = |S_{xy}|$, i.e., size of each set. Then let us define a metric as $n_{11}/(n_{11} + n_{10} + n_{01})$. This metric (i.e. Jaccard index [18]) gives a higher score if submitted score has more similarity to a reference solution. This metric times 0.51 is added to overall score.

Maximum cluster size (C_{max}, size of largest cluster) score is set to 12%. Each difference of 1 below a reference solution deducts 0.01 from total score until a full reduction of 0.12 is reached.

Cluster number (C_{num}, total number of clusters) score is set to 12%. Each difference of 1 above a reference solution deducts 0.01 from total score until a full reduction of 0.12 is reached.

Runtime (t) constitutes 25% of score. Each one second difference above a reference solution deducts 0.01 from final score, until a full deduction of 0.25 is reached. We have the following scoring formula, where sub refers to submitted and ref refers to reference solution:

$$\begin{aligned}(0.51n_{11})/(n_{11}+n_{10}+n_{01})+\\ max[0, min(0.12, 0.12 - 0.01 * (C_{max-ref} - C_{max-sub}))]+\\ max[0, min(0.12, 0.12 - 0.01 * (C_{num-sub} - C_{num-ref}))]+\\ max[0, min(0.25, 0.25 - 0.01 * (t_{sub} - t_{ref}))]\end{aligned} \quad (1)$$

The larger score is the better. All testcases have equal weight.

6. TESTCASES

As of this writeup, 10 testcases have been provided including input parameter variations, each of which requires a separate run and unique outputs. The following table summarizes runtime and cluster statistics. Testcases 1 and 2 use 0.2x0.2μm^2 clips, whereas testcases 3 use 0.25x0.25μm^2.

name	t_{ref} (s)	$C_{num-ref}$	$C_{max-ref}$
testcase1	0.903	8	5
testcase1ap95	1.808	4	6
testcase1e4	1.324	5	5
testcase2	1.167	26	104
testcase2ap95	0.854	13	106
testcase2ap90	1.168	10	114
testcase2e4	0.874	18	104
testcase3	1.48	85	776
testcase3ap95	1.123	53	784
testcase3ap90	1.5	39	1040

7. CONTEST RESULTS

This paper is written before the contest is over. For updates to the contest and results, as well as testcases to download and an extensive Q&A, please visit: http://cad-contest-2016.el.cycu.edu.tw/problem_C/default.html

8. REFERENCES

[1] R.O. Topaloglu, "ICCAD-2014 CAD Contest In Design For Manufacturability Flow For Advanced Semiconductor Nodes And Benchmark Suite," Proc. ICCAD, 2014, pp. 367-368.

[2] Y.-C. Lai, F.E. Gennari, M.W. Moskewicz, J. Lei, and W. Lai, "Method and System for Performing Pattern Classification Of Patterns In Integrated Circuit Designs," US8,079,005.

[3] M. Gabrani and P. Hurley, "IC Layout Pattern Matching And Classification System And Method," US8,363,922.

[4] R.O. Topaloglu, "Methods For Fabricating An Electrically Correct Integrated Circuit," US8,336,011.

[5] D. Ding, X. Wu, J. Ghosh, and D.Z. Pan, "Machine Learning Based Lithographic Hotspot Detection with Critical-Feature Extraction and Classification," Proc. IEEE Int. Conf. IC Design and Technology, 2009, pp. 219-222.

[6] Y.-T. Yu, G.-H. Lin, I.H.-R. Jiang, and C. Chiang, "Machine-Learning-Based Hotspot Detection Using Topological Classification and Critical Feature Extraction," Proc. DAC, 2013, pp. 1-6.

[7] J. Guo, F. Yang, S. Sinha, C. Chiang, and X. Zeng, "Improved Tangent Space Based Distance Metric for Accurate Lithographic Hotspot Classification," Proc. DAC, 2012, pp. 1169-1174.

[8] J.-Y. Wuu, F.G. Pikus, A. Torres, M. Marek-Sadowska, "Rapid Layout Pattern Classification," Proc. ASP-DAC, 2011, pp. 781-786.

[9] P.-H. Wu, C.-W. Chen, C.-R. Wu, and T.-Y. Ho, "Triangle-based Process Hotspot Classification with Dummification in EUVL," Proc. VLSI-DAT, 2014, pp. 1-4.

[10] J. Xu, K.N. Krishnamoorthy, E. Teoh, V. Dai, and L. Capodieci, "Design Layout Analysis and DFM Optimization using Topological Patterns," Proc. SPIE 94270Q-2, 2015.

[11] C.C. Chang, I.C. Shih, J.F. Lin, Y.S. Yen, C.M. Lai, et al. "Layout Patterning Check for DFM," Proc. SPIE 69251R-1, 2008.

[12] F.E. Gennari and A.R. Neureuther, "A Pattern Matching System for Linking TCAD and EDA," Proc. ISQED, 2004, pp. 165-170.

[13] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Y. Ma, "Robust Face Recognition via Sparse Representation," IEEE Tran. on Pattern Analysis And Machine Intelligence, Vol. 31, No. 2, 2009, pp. 210-227.

[14] G. Lu and F.T.S. Yu, "Pattern Classification Using A Joint Transform Correlator Based Nearest Neighbor Classifier," SPIE Opt. Eng, 35(8), 1996, pp. 2162-2170.

[15] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," IEEE Tran. Signal Processing, Vol. 54, No. 11, 2006, pp. 4311-4322.

[16] T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, and A.Y. Wu, "An Efficient k-Means Clustering Algorithm: Analysis and Implementation," IEEE Tran. Pattern Analysis And Machine Intelligence, Vol. 24, No. 7, 2002, pp. 881-892.

[17] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis," IEEE Tran. Pattern Analysis And Machine Intelligence, Vol. 24, No. 5, 2002, pp. 603-619.

[18] S. Wagner and D. Wagner, "Comparing Clusterings - An Overview," 2007. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.164.6189