Identification of Efficient Clustering Techniques for Test Power Activity on the Layout

Harshad Dhotre* Stephan Eggersglüß*† Rolf Drechsler*†
*Institute of Computer Science, University of Bremen, 28359 Bremen, Germany
†Cyber-Physical Systems, DFKI GmbH, 28359 Bremen, Germany
dhotre@uni-bremen.de, segg@informatik.uni-bremen.de, drechsler@uni-bremen.de

***Abstract*—With the increase in transistor density in state-of-the-art circuits the power behavior of integrated circuits changes drastically, which may result in device failures. This may become worse while testing, because of the high transient activity in smaller area. This may lead to high power consumption and failures in certain areas as compared to other parts of the die. For this reason, high power density areas on the integrated circuits need to be identified on the layout to avoid effects such as IR-drop, EM and noise as early as possible. Previously, this was usually considered by manually dividing the layout in equal blocks. However, this method may not provide the desired accuracy due to e.g. boundary effects and manual errors. In this paper, we propose the use of pattern recognition/machine learning techniques to dynamically partition the layout in clusters to identify high power density areas under test application. We show how machine learning techniques can be used to model the clustering problem and analyze the feasibility as well as the performance of several algorithms on benchmark circuits. These techniques avoid the errors on static boundaries and account for pattern dependent behavior. Furthermore, the proposed clustering is validated by comparing the results to a contour of an industrial tool.**

I. Introduction

The technology is scaling to its peak with multiple modules on single SOCs resulting in high compaction of transistors in smaller area. Before production or sample manufacturing, simulations and analysis are performed for sign-off to reduce chances of real time failures of the chip. The design flow of RTL to GDS generates a huge database for the design, which is used for various sign-off analysis purposes before tape-out in the industry. The layout design database along with various libraries is used for different analysis purposes, e.g. power and IR-drop. Considering the placement and routing of the cells is important because of the local behavior on the silicon, which is under high stress of power and heat due to the high density.

The significance of test power analysis is growing. Due to test time and test costs constraints, algorithms for *Automatic Test Pattern Generation* (ATPG) are supposed to generate a very compact test set. This correlates typically with an increased number of transitions during the shift and capture cycles and the risk of failures increases due to high power consumption in the test mode. It is not only important to analyze the test power of the final layout with the final test set, but also for estimation purposes in various earlier design stages. By this, countermeasures can be taken into account early in the design phase. However, highly accurate power and IR-drop simulations are limited for very few test patterns because of time and other resource constraints. As a result, approximation metrics such as *Weighted Switching Activity* (WSA) are often used.

Besides the analysis of the global power consumption for the complete layout, the localization of hot-spots and power-critical areas is very important. A test is able to have a low global power consumption which is evenly distributed around most of the area, but has a concentrated high activity in certain areas on the layout. This could lead to thermal or IR-drop problems. Typically, the layout is viewed as static divisions where the whole layout is divided in areas of fixed sizes. This view is applicable in almost all areas like cell placement [14], [32], scan cells routing and power analysis of test patterns [11], [34]. this is perfectly suitable for some kind of analyses and tasks, e.g. during place and route. However, it might not be suitable for methods such as test power and IR-drop analysis, where the results are mostly in contour form with manual thresholds [27].

Previous methods for test power analysis and test pattern regeneration such as [1], [11], [16], [17] also use static partitioning techniques to partition the layout in equal areas. Such kind of partitioning may be disadvantageous and, hence, there is the need to dynamically identify such power critical areas with more accuracy taking the test behavior into account.

Recently, a new power metric terminology i.e. *Transient Power Activity* (TPA) was proposed in [9], which is more accurate as compared to the commonly used WSA metric. This previous work focuses on the accuracy of the proposed metric and indicates the introductory use of machine learning techniques to identify the power locality. However, it has not been shown how to identify suitable clustering techniques and how to model the problem in detail.

In contrast, this paper concentrates on modeling and the application of different machine-learning based clustering algorithms and their comparison. The application of the clustering technique is metric-independent and these kind of algorithms are applied successfully in the field of image processing. The layout of modern high density SOCs can be homologous with an image, where the pixel in the image are analogous with the library design units or instances. Certain image processing applications involve clustering of similar pixels to identify objects [3], [15].

This paper describes how different machine learning-based algorithms can be used to model the test power analysis problem, i.e. the identification/estimation of high power consuming areas on the layout. We show that dynamic clustering algorithms can be successfully applied in a fast manner without producing the risk of boundary errors. Furthermore, we compare the results and the performance of these algorithms and validate the result with a commercial tool.

The rest of paper is organized as follows. Section II discusses the related work, while Section III presents the used clustering techniques. The experimental results are given in Section V and conclusions are drawn in Section VI.

II. Related Work

The related approaches to identify the high power consuming areas used in previous work are based on the static partitioning of the layout in equal areas. There are mostly rated

according to their weighted switching activities [8], [18], [19]. Another similar technique proposed in [22] identifies areas where IR-drop likely occurs, but it is based on the probability of switching activity at gate level and does not take the test patterns into account.

The uniform partitioning is disadvantageous because it may not detect the high power activity areas crossing borders between the partitions, e.g. when a high power-consuming region is spread over more than one static partition. Hence, there is a need to introduce a dynamic partitioning approach, which clusters the high power activity areas depending on neighboring instances aggregation.

Another approach [33] identifies the peak current to determine the power-safe patterns. This approach uses WSA and a WSA threshold to relate it to current limits and also partitions the layout in equal sizes. A further method for pattern grading based on WSA for critical paths is used in [21]. Here. layout information is used to identify aggressors and power-supply noise.

Clustering techniques used in image processing are highly specific depending on their application and suitability. For instance, the technique to find a tumor via MRI image analysis (proposed in [7]) is highly specific to its defined task. Another specific application for segmentation of a cerebral cortex image using a unique Artificial Neural Network (ANN) training algorithm is used in [26]. The algorithm proposed in [31] is to find dimensions of droplets in an image. Some more clustering algorithms in image processing are proposed in [24], [29], [30]. Clustering techniques are used in different applications like big data, security, DSP, WSN etc. [10], [20]. A lot of other algorithms are proposed for similar tasks but not referenced due to page limitation.

Since most of the algorithms mentioned before are application-specific, it is particularly important to identify a suitable algorithm for the specific needs of the test power layout clustering problem. A number of clustering techniques are proposed in the field of data science, but these algorithms may not give the same results for different applications. Also there are very few metrics to identify the perfect algorithm for the desired application, hence we pre-selected a few algorithms depending upon their characteristics and suitability and compared their results among themselves along with the power analysis results of commercial tools.

III. Design Layout

Since the layout is to be clustered in different areas, first lets see the general layout formation because it will help to pre-select the clustering algorithms. After the finalization of the design specification, the high level architecture of the design is divided into different blocks like analog, digital or mixed. These are further sub-divided into modules and their sub-modules maintain their various hierarchical levels. But the approximate top-level floorplan usually is decided initially with these reference top-level modules in order to ease various design activities. Further, the detailed placement and routing of each module and sub-module can be done manually, automated, semi-automated or in a customized way. Figure 1 shows a small layout of a mixed signal design of a Digital to Analog Converter (DAC). It can be seen that the analog components such as OP-Amp and analog multiplexer as well digital components control logic are placed in a specific way in order to minimize the silicon area and other effects such as EM interference, operating frequency and voltage. After the power

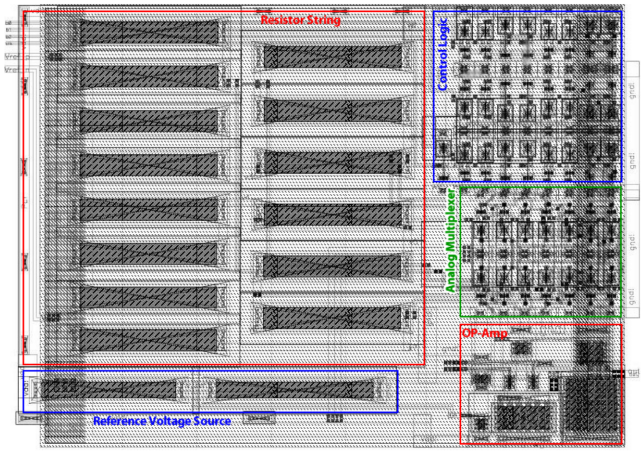

Fig. 1: Layout of DAC

y6	R_{x1y6}	R_{x2y6}	R_{x3y6}	R_{x4y6}	R_{x5y6}	R_{x6y6}	R_{x7y6}
y5	R_{x1y5}	R_{x2y5}	R_{x3y5}	R_{x4y5}	R_{x5y5}	R_{x6y5}	R_{x7y5}
y4	R_{x1y4}	R_{x2y4}	R_{x3y4}	R_{x4y4}	R_{x5y4}	R_{x6y4}	R_{x7y4}
y3	R_{x1y3}	R_{x2y3}	R_{x3y3}	R_{x4y3}	R_{x5y3}	R_{x6y3}	R_{x7y3}
y2	R_{x1y2}	R_{x2y2}	R_{x3y2}	R_{x4y2}	R_{x5y2}	R_{x6y2}	R_{x7y2}
y1	R_{x1y1}	R_{x2y1}	R_{x3y1}	R_{x4y1}	R_{x5y1}	R_{x6y1}	R_{x7y1}
	x1	x2	x3	x4	x5	x6	x7

Fig. 2: Circuit partitioning example [11]

and IR-drop analysis of the design, the power distribution map can be obtained in form of contours.

For more complex circuits or higher density SOCs, the detailed placement and routing is done in a semi-automated or automated way using commercial tools. These layouts have specific patterns for the component placing in order to keep the area, routing and total cost low. Highly accurate power analysis commercial tools perform power and IR-drop analysis which is typically based on VCD processing. This kind of analysis is highly time and resource consuming, hence this is not possible for all test patterns. Usually, results are given in form of files or lists which are less useful for diagnosis and automation.

Sometimes the power consumption is also taken into account considering the toggling probability or average functional power consumption for each power domain. The test power dissipation is typically very high as compared to the functional mode [2], [23]. The ATPG tools typically uses the WSA as a measuring factor. As described above, previous work deals with the static partitioning and power-safe patterns are generated such that the power consumption is diminished or averaged in such a static block which is identified as power-critical.

Figure 2 shows the manual partition of the layout in equal parts where x and y represent coordinates and Rxy represents the WSA of the corresponding block. A common method to identify power-critical regions on the layout is to compare the WSA value of these blocks under the application of test patterns. However, these blocks are formed without taking the test patterns into account. Similar kinds of manual partitioning

approaches are used in [1], [8], [18], [22].

Hence, dynamic partitioning is necessary to build layout-based clusters according to the test pattern's behavior. This can be done using machine learning-based clustering techniques.

IV. Clustering Techniques

The instances in the layout are to be regionally clustered depending on the density of the power activity under test application. The metric used is not relevant for the application. For instance, it can be WSA or TPA [9] or any other instance-related metric depending on the design phase and desired level of accuracy. We refer to the considered power metric therefore as generalized *Local Power Metric* (LPM).

The proposed modeling as a clustering problem is based on the following information collected for each instance of the design:

- instance name or hierarchical name for identification
- library reference design unit and technology database for LPM extraction, e.g. to extract power factors
- location of the instance, i.e. x and y coordinates, obtained from the DEF file
- area information of the cell obtained from the LEF file

Figure 3 shows an example of a partitioning approach on the layout. The instances in the design are arranged in horizontal rows from 1, 2,. . .,10 and in vertical columns as A, B,. . .,J. Each number represents the LPM value of an instance on the layout. Using a static partitioning approach, the layout is divided in 25 equal blocks consisting of four instances each (indicated by a different blue/grey color). According to the metric used and the applied simulation, each instance is assigned an LPM value.

For example, the first block in the upper left corner has 4 instances with 4,0,1,1 as LPM values and an average LPM value of 1.5. The average LPM is calculated in this example based on the assumption that each instance has the same size. This is different in reality. However, taking different instance sizes into account is straight-forward.

Such a static partitioning scheme may cause errors if neighboring instances at the boundary but in different blocks have a high LPM value and the other instances in the same block have a low or average LPM value. For example, the red cluster (which is not part of the initial blocks) shown in Figure 3 contains 4 neighboring instances of higher LPM from different static blocks. If these instances form a block, the average LPM will be 7.5 which is high compared to the average LPM value of all other static blocks. Therefore, the identification of hotspots may fail because the blocks are formed in an unfortunate way. Hence, there is the need to dynamically form partitions or clusters of instances to detect such high power activity areas.

The layout consists of different cells of varied areas and sizes which are arranged in horizontal rows and columns. In order to keep the routing and area cost minimal the cells are very closely placed. As described above, machine learning-based clustering techniques can be used to form these dynamic partitions automatically. Considering the placement of cells on the layout, some clustering algorithms may not be suitable because of the structure of the layout. For example, the Hierarchical algorithm results into a single cluster because varied sizes of blocks agglomerate under one block. Other clustering techniques like edge-based segmentation and snake-based approaches will cluster each instance on the layout separately in an individual cluster because of the small boundary between instances, which can be recognized as edge. Other

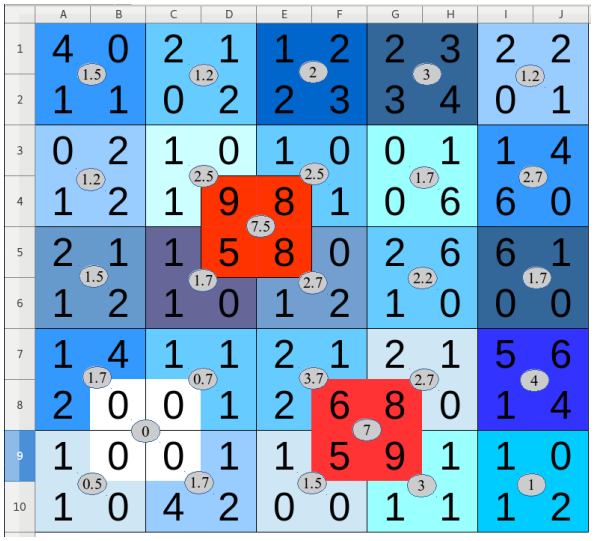

Fig. 3: Static (blue/grey) and dynamic (red) partitions

modern clustering techniques like fuzzy clustering, neural networks and evolutionary approaches are used for higher dimensions and supervised learning. These cause high costs in terms of computational complexity and run time.

The dynamic partitioning technique for the considered problem should be fast, efficient from distinct cluster points and should be based on unsupervised clustering since each design is different and varies with technology and physical properties. Based on this filters, we analyzed different algorithms and from our requirements pre-selected the following algorithms for a detailed analysis.

A. *k-means Clustering*

The *k-means clustering* algorithm is one of the popular algorithms used in data clustering and is also often applied in image processing [4], [6], [28]. The targeted problem requires to average out the local LPM activity and to form partitions with varied/distinct averaged areas dynamically. This makes it suitable for our approach. For modeling the clustering problem, the features for dynamic partitioning on the layout consist of the following:

- instance location, i.e. x and y coordinates,
- area a
- LPM value l_x for each instance x

The clusters, which are to be formed by the algorithm, will have attributes such as the cluster center, i.e. x and y coordinates of a cluster and the LPM density. Also a label is assigned to the cluster, which can be referred as the name of the cluster and attributes as centroid. The LPM density is defined as the accumulated LPM value of all instances in the formed cluster divided by the area a.

The k-means clustering algorithm divides m instances $x_i = (x_1, \ldots, x_m)$ in the design into k partitions/clusters (k is a parameter given by the user) such that the function for the centroids or means $m_1, m_2, \ldots, m_k$ is kept minimum concerning the variance, i.e. within-cluster sum of square for dimensions d_k. The dimension is defined as the distance between the clusters, which depends on the x and y coordinates of the cell, the LPM value of the instances and the area.

$$\sum_{k=1}^{K} \sum_{i \in d_k} ||x_i - m_k||^2 \tag{1}$$

The k-means clustering algorithm partitions the layout into different clusters by aggregating the instances having similar LPM density values and are close to each other. The Euclidean distance parameter is considered here for keeping the function minimum. The outcome of the algorithm is that instances, which have a similar LPM density and are close to each other are clustered in one partition. An arbitrary clustering is not possible because the maximum number of clusters is limited. The algorithm therefore optimizes the clusters according to the given objective.

The algorithm assigns the mean value of the LPM density to each cluster during the computation and gives the values as a result along with the mean x and y coordinates of all instances in the cluster. These values are considered as centroid parameters and characterize each cluster. This technique overcomes the drawback caused by static partitioning of the layout in equal blocks. Another advantage is that the borders between clusters are formed such that instances of similar LPM values belong to the same partition, which is not the case when manual partitioning is applied.

B. Mean Shift Clustering

A similar centroid based algorithm, i.e. the *Mean Shift* [5], [13] algorithm, is further analyzed. This algorithm works on the principle of calculating and updating the mean point of the given cluster through its instances. The input to this algorithm is similar to those in the k-means algorithm, i.e. x, y coordinates as well as LPM density. The difference to the k-means algorithm is that the number of clusters is not given as a parameter, but the quantile. This parameter is used to specify the LPM bandwidth of a cluster. The main idea of this algorithm is to identify local maxima and collect surrounding points of similar value, i.e. the neighboring area converges to one point of high density by hill-climbing. The algorithm updates the centroid of a cluster consisting of coordinates and LPM density with every iteration. For every instance corresponding to the centroid x_i for iteration n, the updated centroid after adding a new instance to the cluster is given by:

$$x_i^{n+1} = x_i^n + m(x_i^n) \tag{2}$$

Here, m is the mean shift vector, which is calculated for each iteration using a kernel function K. The kernel function uses the bandwidth and represents the kernel density estimation, which is typically Gaussian and uses the bandwidth as deviation parameter. The bandwidth can be estimated by providing the median pairwise distance between two points known as quantile for the function. Applying the mean shift vector represents therefore a shift of the centroid to a higher density.

The number of clusters varies for different values of the quantile parameter. Considering the layout information and geometric structure of the library design units, the updated centroid of a cluster is different for each iteration. The clusters are less distinct but a major drawback is the run time. The algorithm performs multiple nearest neighbor searches during the execution of the iterations. The algorithm converges at a certain point and stops iterating when the changes in the centroid are small. In our case, this effect results often in a single cluster, which is not desirable.

C. DBSCAN clustering

We performed further more experimentation with the *Density Based Spatial Clustering of Applications with Noise* (DBSCAN) algorithm [12]. Basically, this algorithm separates the areas of low densities and high densities by grouping together those points, which are closely related to each other. The algorithm depends on additional inputs like the minimum set of points and the maximum distance between the neighborhood points known as EPS along with the regular data-set. The provided data-set in our case are the x and y coordinates and the LPM density, which form the dimensions for the clustering.

Mainly, the algorithm distinguishes the between two different points, core points and outliers. The algorithm works in a deterministic way concerning the clustering of the core points. A core point is a point which is connected to another core point in the given range. All others points are outliers or considered as noise. These outliers are assigned to a cluster in a non-deterministic (heuristically) way. In this way, the clusters are formed of various densities.

The DBSCAN behavior results in multiple trials of the same points in different clusters thereby increasing the run time. Even though it has the advantage of noise immunity, the distance has to be pre-selected, which is difficult to predict. Also, the number of clusters varies depending on these input parameters. Sometimes, the small difference in densities might result in mixing high value LPM areas and low value LPM areas. The above algorithms can be summarized in Table I The experimentation results of all the above algorithms are discussed in the next section.

TABLE I: Algorithms summary

Algorithms		
k-means	Mean Shift	DBSCAN
Aggregation of neighbours and calculating means, number of clusters as parameter, less runtime (heuristic) and efficient clustering for layout partitioning,	Iterative mean shift approach for clustering, bandwidth estimation with specific quantile, number of clusters varies with settings and kernel function, long runtime and inefficient clustering for layout partitioning	Density of instances based approach, number of clusters varies with distance, partly deterministic, location specific with noise immunity, longer runtime and less suitable for layout partitioning

V. Experimental Results

For the experimentation purpose, OpenCores benchmarks are used synthesized and routed with an 180nm open source library. The test patterns were generated using a commercial ATPG tool. The test pattern simulation database, the technology files and the design files were processed using an in-house tool to generate the data-set for clustering. Here, we analyzed the shift and capture cycles for each test to generate the LPM value based on the toggling activity and technology-related power factors (TPA metric [9]).

This data-set consists of the x and y coordinates of all instances in the design, the corresponding areas and the LPM values of all instances. The clustering is performed using Python scripts [25]. For each benchmark, a single power-risky test pattern was chosen for the detailed locality analysis.

Table II shows the results of the clustering algorithms along with details of the benchmarks. The *Benchmarks* column contains the name of the circuit. The area in millimeters is mentioned in the second column. The total number of scan cells and instances in the design are given in the corresponding columns. The results of the clustering algorithms along with their corresponding parameters and results, i.e. the number of determined clusters (*#Clus.*) and the run time of the algorithm are shown in their respective columns. The parameter for the k-means algorithm is the number of clusters, while the *EPS* and *Quantile* are the parameters for DBSCAN and Mean Shift.

TABLE II: Results for open cores

Benchmarks	Area	Scan_cells	Instances	Clustering Algorithms							
				Mean Shift			DBSCAN			k-means	
				Quantile	#Clus.	Time	EPS	#Clus.	Time	#Clus.	Time
Ethernet	3.48 X 3.48	11617	229036	0.2	1	16h	0.1	14	34min	20	1.9sec
				0.5	1	20h	0.2	14	40min	40	9.3sec
				0.7	NA	NA	0.3	3	70min	60	17.4sec
				0.9	NA	NA	0.5	1	90min	80	22.0sec
wb_conmax	2.21 X 2.21	4538	85065	0.2	1	10h	0.1	9	22min	20	1.0sec
				0.5	1	15h	0.2	2	30min	40	1.9sec
				0.7	NA	NA	0.3	1	58min	60	2.8sec
				0.9	NA	NA	0.5	1	75min	80	3.4sec
pci_bridge32	2.18 X 2.18	5381	88552	0.2	1	11h	0.1	7	25min	20	1.2sec
				0.5	2	16h	0.2	3	32min	40	1.8sec
				0.7	2	24h	0.3	1	62min	60	2.7sec
				0.9	NA	NA	0.5	1	70min	80	3.8sec
ac97_ctrl	1.64 X 1.64	3034	49995	0.2	1	6h	0.1	4	16min	20	0.6sec
				0.5	2	8h	0.2	2	20min	40	0.9sec
				0.7	2	12h	0.3	1	28min	60	1.2sec
				0.9	NA	NA	0.5	1	36min	80	1.5sec

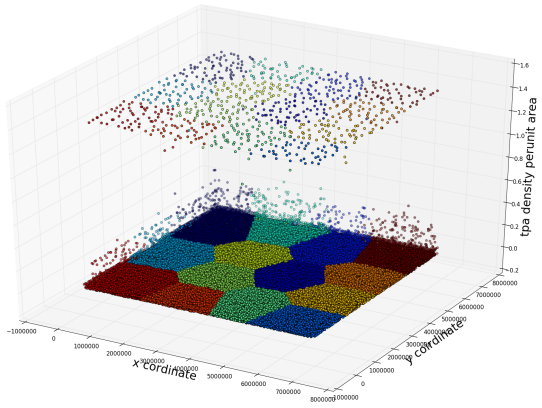

Fig. 4: Ethernet: DBSCAN result for EPS=0.1

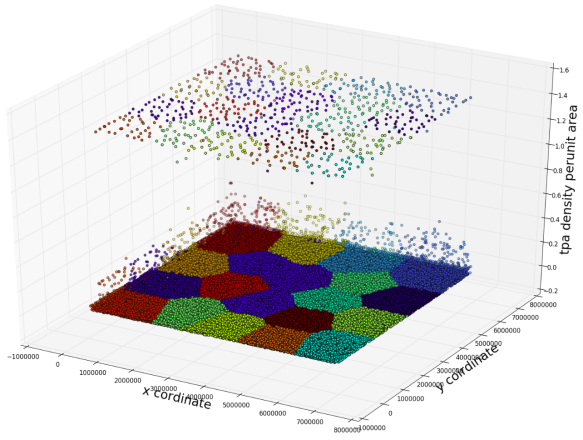

Fig. 5: Ethernet: k-means result for 20 clusters

Four runs with different parameters were performed for each algorithm.

It can be observed that the Mean Shift algorithm needs much run time to form the clusters even when the quantile parameter is low. Some of the runs were aborted after one day of runtime (NA). Also, the number of generated clusters is very low. For some circuits, only one cluster is determined. This is due to the small variability of the LPM values. Therefore, the Mean Shift algorithm is not suited for the considered problem.

The DBSCAN algorithm generates more clusters in general. An example illustration of the DBSCAN-based clustering is shown in Figure 4. Besides the image, detailed data which instance belongs to which cluster is also available. However, the run time of the DBSCAN algorithm is still quite high taking several minutes up to more than an hour for one single pattern depending on the parameters.

In contrast, the k-means algorithm is very fast. Given the number of desired clusters as parameter, the algorithm needs only a few seconds to cluster the layout. An illustration of the clustering is given in Figure 5. The more clusters are targeted, the longer is the run time. However, it is still significantly lower than the other algorithms. Therefore, this algorithm is well suited for a quick test pattern analysis and estimation, in particular, when many patterns have to be analyzed. The main drawback of this algorithm is that the number of clusters is typically not known beforehand. Therefore, the algorithm has to be applied for more parameters, i.e. number of clusters. Future work will be the development of a strategy to find the optimal number of clusters in an automated way.

We verified our proposed clustering approach by a comparison to a commercial tool, which produces an IR-drop contour map on the layout. Unfortunately, the results may vary because different metrics were used and the commercial tool produces the contour based on a VCD file and its own internal calculations. While the commercial tool uses accurate and time-consuming calculations, our approach was based on the approximate TPA metric.

For this comparison, a single capture cycle of a test was analyzed due to the high run time of the commercial tool. The analysis of the commercial tool took around one hour for this one cycle and the contour shown in Figure 6 was produced. The clustering results obtained for the same capture cycle were correlated with the higher power consuming areas of the contour. As used in practice, a design-specific threshold (filter) is defined to identify power-critical areas of the design. It was found that 67% of the instances in higher value LPM clusters produced by k-means clustering were part of hot areas of the contour. This confirms that the proposed clustering approach is well suited for a quick test power analysis and estimation on the layout.

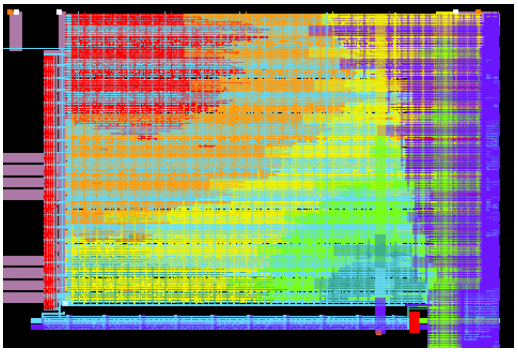

Fig. 6: IR-drop contour by commercial tool

VI. Conclusion and Future Work

Test power analysis is an important task in the design flow of today's circuits. Here, early analysis runs are important to estimate critical areas in order to prepare countermeasures. Besides the global power consumption on the complete chip layout, the identification of local hot spots is also very important. In this work, we have shown how to use machine learning techniques, i.e. clustering algorithms, in order to identify power-critical areas under test application. We presented the use of several clustering algorithms and compared the results to each other on benchmark circuits. As the most advantageous algorithm, the k-means algorithms has been identified. This algorithm is able to cluster the layout according to a given approximate power metric very fast. Future work is to incorporate more details of the layout such as the power grid and additional clustering dimensions for further accuracy improvement.

VII. Acknowledgment

The work has been supported by the Institutional Strategy of the University of Bremen, funded by the German Excellence Initiative and by the German Research Foundation (DFG) under contract number EG 290/5-1. This work was also supported by the subproject P01 Predictive function of the Collaborative Research Center SFB1232, funded by the German Research Foundation.

References

[1] F. Bao, M. Tehranipoor, and H. Chen, "Worst-case critical-path delay analysis considering power-supply noise," in *IEEE Asian Test Symp.*, 2013, pp. 37–42.

[2] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Kluwer Academic, 2000.

[3] C. Cariou and K. Chehdi, "A new k-nearest neighbor density-based clustering method and its application to hyperspectral images," in *IEEE Int'l Geoscience and Remote Sensing Symp.*, 2016, pp. 6161–6164.

[4] C. W. Chen, J. Luo, and K. J. Parker, "Image segmentation via adaptive K-mean clustering and knowledge-based morphological operations with biomedical applications," *IEEE Trans. on Image Processing*, vol. 7, no. 12, pp. 1673–1683, 1998.

[5] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. on Pattern Analysis and Machine Intell.*, vol. 17, no. 8, pp. 790–799, 1995.

[6] D. A. Clausi, "K-means iterative fisher (KIF) unsupervised clustering algorithm applied to image texture segmentation," *Pattern Recognition*, vol. 35, no. 9, pp. 1959–1972, 2002.

[7] M. Dadiani, N. Kahana, B. Kaufman, E. Konen, M. Sklair-Levy, and A. Mayer, "Assessment of interstitial fluid pressure in solid tumors via image processing of dce-mri," in *Int'l Conf. on Imaging Systems and Techniques*, 2016, pp. 189–194.

[8] V. Devanathan, C. Ravikumar, and V. Kamakoti, "On power-profiling and pattern generation for power-safe scan tests," in *Design, Automation and Test in Europe*, 2007, pp. 1–6.

[9] H. Dhotre, S. Eggersglüß, M. Dehbashi, U. Pfannkuchen, and R. Drechsler, "Machine learning based test pattern analysis for localizing critical power activity areas." in *IEEE Int'l Symp. on Defect and Fault Tolerance in VLSI Systems*, 2017.

[10] S. Din, A. Ahmad, A. Paul, M. M. ullah Rathore, and J. Gwanggil, "A cluster-based data fusion technique to analyze big data in wireless multi-sensor system," *IEEE Access*, 2017.

[11] S. Eggersglüß, K. Miyase, and X. Wen, "SAT-based post-processing for regional capture power reduction in at-speed scan test generation," in *IEEE European Test Symp.*, 2016, pp. 1–6.

[12] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[13] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Trans. on Information Theory*, vol. 21, no. 1, pp. 32–40, 1975.

[14] P. Ghosal, T. Samanta, H. Rahaman, and P. Dasgupta, "Thermal-aware placement of standard cells and gate arrays: Studies and observations," in *INTEGRATION, the VLSI Journal*, 2008, pp. 369–374.

[15] M. A. Hasnat, O. Alata, and A. Trémeau, "Unsupervised clustering of depth images using watson mixture model." in *ICPR*, 2014, pp. 214–219.

[16] S. Kiamehr, F. Firouzi, and M. B. Tahoori, "A layout-aware X-filling approach for dynamic power supply noise reduction in at-speed scan testing," in *IEEE European Test Symp.*, 2013, pp. 52–57.

[17] J. Lee and M. Tehranipoor, "Layout-aware transition-delay fault pattern generation with evenly distributed switching activity," *Journal of Low Power Electronics*, vol. 4, no. 3, pp. 1–12, 2008.

[18] ——, "Layout-aware transition-delay fault pattern generation with evenly distributed switching activity," *Journal of Low Power Electronics*, vol. 4, no. 3, pp. 360–371, 2008.

[19] Y.-H. Li, W.-C. Lien, C. Lin, and K.-J. Lee, "Capture-power-safe test pattern determination for at-speed scan-based testing," *IEEE Trans. on CAD of Integrated Circ. and Systems*, vol. 33, no. 1, pp. 127–138, 2014.

[20] A. Little, X. Mountrouidou, and D. Moseley, "Spectral clustering technique for classifying network attacks," in *IEEE Int'l Conf. on Big Data Security on Cloud, IEEE Int'l Conf. on High Performance and Smart Comput., and IEEE Int'l Conf. on Intell. Data and Security*, 2016, pp. 406–411.

[21] J. Ma, M. Tehranipoor, and P. Girard, "A layout-aware pattern grading procedure for critical paths considering power supply noise and crosstalk," *Journal of Electronic Testing: Theory and Applications*, vol. 28, no. 2, pp. 201–214, 2012.

[22] K. Miyase, M. Sauer, B. Becker, X. Wen, and S. Kajihara, "Identification of high power consuming areas with gate type and logic level information," in *IEEE European Test Symp.*, 2015, pp. 1–6.

[23] N. Nicolici and B. Al-Hashimi, *Power-constrained testing of VLSI circuits*. Kluwer Academic Publishers, Boston, MA, 2003.

[24] A. Oliver, X. Muñoz, J. Batlle, L. Pacheco, and J. Freixenet, "Improving clustering algorithms for image segmentation using contour and region information," in *IEEE Int'l Conf. on Automation, Quality and Testing, Robotics*, vol. 2, 2006, pp. 315–320.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[26] M. S. Pukish, S. Wang, and B. M. Wilamowski, "Segmentation of cerebral cortex mri images with artificial neural network (ann) training," in *Int'l Conf. on Human System Interaction*, 2013, pp. 320–327.

[27] T. Sekine and H. Asai, "Meshless modeling for electrical-thermal cosimulation of IR-drop analysis," in *IEEE Int'l Symp. on Electromagnetic Compatibility*, 2016, pp. 182–186.

[28] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis, and machine vision*. Cengage Learning, 2014.

[29] S. N. Sulaiman and N. A. M. Isa, "Adaptive fuzzy-k-means clustering algorithm for image segmentation," *IEEE Trans. on Consumer Electronics*, vol. 56, no. 4, 2010.

[30] T. Sun, Z. Ren, and S. Ding, "Region-based semi-supervised clustering image segmentation," in *Int'l Conf. on Natural Computation*, vol. 4, 2011, pp. 1855–1858.

[31] B. Tejaswi, P. Sivasakthivel, and M. Venkatesan, "Image processing algorithm for droplet measurement after impingement," in *IEEE Int'l Conf. on Computat. Intell. and Comput. Research*, 2016, pp. 1–4.

[32] G. Wu and C. Chu, "Detailed placement algorithm for VLSI design with double-row height standard cells," *IEEE Trans. on CAD of Integrated Circ. and Systems*, vol. 35, no. 9, pp. 1569–1573, 2016.

[33] W. Zhao, J. Ma, M. Tehranipoor, and S. Chakravarty, "Power-safe application of transition delay fault patterns considering current limit during wafer test," in *IEEE Asian Test Symp.*, 2010, pp. 301–306.

[34] W. Zhao and M. Tehranipoor, "Peak power identification on power bumps during test application," in *Int'l Green Comput. Conf. and Workshop*, 2011, pp. 1–3.