Memristive Crossbar Mapping for Neuromorphic Computing Systems on 3D IC

Qi Xu
EST Department, USTC
xuqi@mail.ustc.edu.cn

Song Chen
EST Department, USTC
songch@ustc.edu.cn

Bei Yu
CSE Department, CUHK
byu@cse.cuhk.edu.hk

Feng Wu
EEIS Department, USTC
fengwu@ustc.edu.cn

ABSTRACT

In recent years, neuromorphic computing systems based on memristive crossbar have provided a promising solution to enable acceleration of neural networks. Meanwhile, most of the neural networks used in realistic applications are often sparse. If such sparse neural network is directly implemented on a single memristive crossbar, it would result in inefficient hardware realizations. In this work, we propose 3D-FNC, a 3D floorplanning framework for neuromorphic computing systems in consideration of both crossbar utilization and design cost. 3D-FNC groups neurons that connect more common neurons into one cluster, where the optimal number of clusters is determined by L-method. As a result, the connections of a neural network can be effectively mapped to memristive crossbars or discrete synapses. Finally, a 3D floorplanning for memristive crossbars and neurons is developed to reduce area and wirelength cost. Experimental results show that 3D-FNC can achieve highly hardware-efficient designs, compared to state-of-the-art.

ACM Reference Format:
Qi Xu, Song Chen, Bei Yu, and Feng Wu. 2018. Memristive Crossbar Mapping for Neuromorphic Computing Systems on 3D IC . In *Proceedings of Great Lakes Symposium on VLSI (GLSVLSI'18)*. ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.1145/3194554.3194636

1 INTRODUCTION

Neuromorphic computing systems (NCS) based on hardware designs are intended to mimic neuro-biological architectures [1]. However, the CMOS technology implementation has been shown to suffer from mismatch between NCS building blocks (neuron and synapse) and CMOS primitives (Boolean logic) [2]. Moreover, based on the conventional CMOS technology, a large number of transistors are required to mimic a single synapse [3]. To address this problem, many neuromorphic designs on device and architecture level have been explored. The emerging memristor device inspires the approaches of using memristors to implement synapse circuit due to the similarity between the memristive and synaptic behaviors [4]. In order to achieve area efficient design, a memristive crossbar structure that connects all its **pre-synaptic** (i.e. input) neurons to all its **post-synaptic** (i.e. output) neurons is developed [5]. In addition, NCS with high complexity and high connectivity is required to implement increasingly demanding computational tasks. Traditional two-dimensional (2D) integration may be hard to meet these requirements, as longer signal transmission distances are introduced due to a large number of connections in 2D integration. On the other hand, three-dimensional integrated circuits (3D ICs) involve vertically stacking multiple dies connected by through silicon vias (TSVs), providing a promising way to enable high density and high computation speed in NCS [6]. Besides, 3D design in NCS can emulate real biophysical processing in human brain [7].

GLSVLSI'18, May 23–25, 2018, Chicago, IL USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5724-1/18/05...$15.00
https://doi.org/10.1145/3194554.3194636

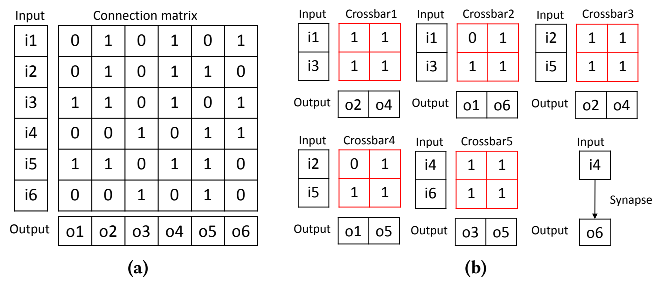


Figure 1: (a) Connection matrix of a feed-forward neural network with six pre-synaptic neurons $\{i_1, \ldots, i_6\}$ and six post-synaptic neurons $\{o_1, \ldots, o_6\}$; (b) The connections are mapped to five memristive crossbars with high utilizations and one discrete synapse.

Many works for NCS implementation on memristive crossbars have been investigated. Wen *et al.* [8] proposed a design automation framework for large scale hybrid neuromorphic computing systems. Wu *et al.* [9] developed a thermal optimization for memristor-based hybrid neuromorphic computing systems. In [8, 9], an iterative spectral clustering is repeatedly performed to group the connections into memristive crossbars. However, since the spectral clustering is executed in every iteration, the methods could be time consuming when the neural network is large. Cui *et al.* [10] proposed a sparse matrix reordering method, which uses both row and column permutation matrices to group the connections into crossbars. However, crossbar utilization is not considered, thus the generated memristive crossbars with low utilization may result in highly area-inefficient designs. In addition, so far all previous works are based on 2D integration, thus some insurmountable obstacles are inevitably introduced by 2D circuits like larger chip area and longer wirelength etc.

To overcome the above issues, in this paper we propose 3D-FNC, a 3D floorplanning framework for neuromorphic computing systems in consideration of both crossbar utilization and design cost. Figure 1 illustrates an example how the proposed framework can effectively map the connections into crossbars with high utilizations. To represent a neural network, in this paper we use connection matrix, which is (0, 1)-matrix that "1" indicates a connection exists between two corresponding neurons and "0" means two corresponding neurons have no connection. As shown in Figure 1(a), if the sparse connection matrix is directly mapped to a crossbar, the utilization of the crossbar is low. However, through several techniques in our framework (e.g. hierarchical clustering), the mapped crossbars with high utilizations are shown in Figure 1(b). The corresponding 3D floorplan is shown

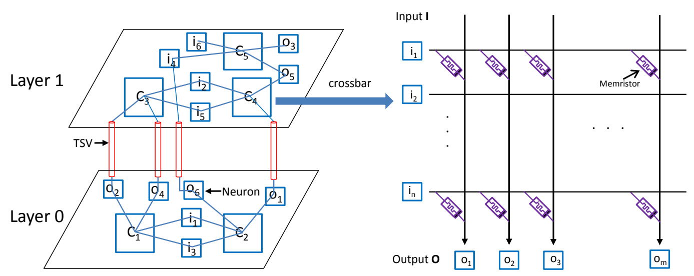


Figure 2: 3D floorplan of the mapping results of the neural network in Figure 1(a) and the memristive crossbar.

in Figure 2. Key technical contributions of this work are listed as follows.

- Hierarchical clustering groups neurons of a neural network into clusters. And the proposed distance metric can ensure that neurons connecting more common neurons are grouped into one cluster.
- An L-method is developed to determine the optimal number of clusters.
- A 3D floorplanning is implemented to minimize the hardware cost.

The remainder of this paper is organized as follows. Section 2 presents preliminaries of NCS. Section 3 is to describe the proposed 3D-FNC framework in details. Section 4 lists experimental results, followed by conclusion in Section 5.

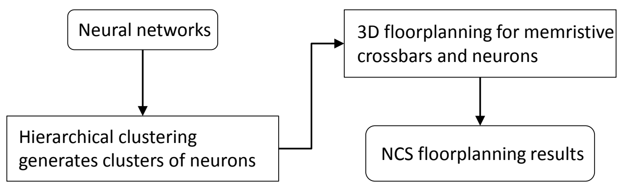
2 PRELIMINARIES

In a neural network, the pre-synaptic neurons send signals into the network and the post-synaptic neurons receive information from the pre-synaptic neurons through the synapses [11]. The synapses apply different weights on the information during the transmission, which can be expressed as $\boldsymbol{O} = \boldsymbol{IC}$. An element c_{ij} in the connection matrix $\boldsymbol{C}$ represents the weight of a synapse between the pre-synaptic neuron i in $\boldsymbol{I}$ and the post-synaptic neuron j in $\boldsymbol{O}$. Since the resistance of memristor is programmed by applying current or voltage, the memristor can be used to implement the weight of the synapse [5]. A memristive crossbar example is shown in Figure 2.

In realistic applications, the size of neural networks is often very large. For instance, AlexNet proposed by Krizhevsky *et al.* [12] in 2012 contains 650K neurons and 60M synapses. In addition, most of large neural networks are sparse. If a sparse neural network is directly implemented on a memristive crossbar, the crossbar utilization can be low, resulting in highly area-inefficient designs. In this paper, we define the utilization of a memristive crossbar as the ratio between the utilized number of connections and the total available connections in the crossbar. In order to enable energy and area efficient design, the large neural network with high sparsity should be implemented by using smaller size crossbars or discrete synapses.

We define the problem of 3D floorplanning for neuromorphic computing systems (3D-FNC) as follows.

Problem 1 (3D-FNC). Given a sparse neural network with n neurons, we map the connections into memristive crossbars with high utilization or discrete synapses. Then we implement the 3D floorplanning of memristive crossbars and neurons to minimize chip area, wirelength, and TSV numbers.


Neural networks
Hierarchical clustering generates clusters of neurons
3D floorplanning for memristive crossbars and neurons
NCS floorplanning results
Figure 3: The flow of 3D-FNC.

Algorithm 1 Hierarchical clustering to cluster neurons

Input: Connection matrix C.
Output: A dendrogram of neurons.

```
1: Assign each neuron to a cluster;
2: Calculate distances among clusters;              ▷ Equation (1)
3: for $i \leftarrow 1$ to $n - 1$ do
4:     Merge two clusters $\{r\}$ and $\{s\}$ with the shortest distance;
5:             ▷ Update distances between $\{r,s\}$ and other clusters $\{k\}$
6:     $d_{\{k\},\{r,s\}} \leftarrow \min[d_{\{k\},\{r\}}, d_{\{k\},\{s\}}]$;
7: end for
8: All neurons are clustered into one cluster;
```

3 3D-FNC FRAMEWORK

Given a sparse neural network, the proposed hierarchical clustering partitions all the neurons into clusters, so that the connections can be easily mapped to set of crossbars with high utilization. Based on the cluster results, we implement a 3D floorplanning of neuromorphic computing systems to achieve a minimum chip area, wirelength, and TSV numbers under fixed-outline and no-overlapping constraints [13]. In this work, the memristive crossbars and the neurons are considered as blocks. The partitioned sequence pair (P-SP) [14] is used to represent 3D floorplans. The chip area is evaluated by the method in [15], which takes the fixed-outline constraint into account. To evaluate the wirelength, we adopt the calculation model in [16] to decompose the nets spanning multiple device layers into sub-nets, one on each device layer, by introducing dummy pins corresponding to TSVs. Besides, the number of TSVs of a net is $|j - i|$ if the net spans layer i to layer j. The flow of 3D-FNC is shown in Figure 3.

3.1 Hierarchical Clustering

Hierarchical clustering is used to generate clusters in a bottom-up iterative manner [17]. In every iteration, two clusters with the shortest distance are merged. Implication of "distance" varies with specific applications in practice. The iterative merging is repeated until all data points are formed into one cluster.

In our proposed framework, we aim to generate clusters that can be mapped to memristive crossbars with high utilization. Therefore, the neurons that connect more common neurons should be grouped into one cluster. In this work, the hierarchical clustering is adopted to cluster neurons. We redefine the distance between two neurons, i_p and i_q ($1 \leq p, q \leq n$), as follows:

$$\mathrm{dist}(i_p, i_q) = \sqrt{\sum_{j=1}^{n} (c_{pj} \,\&\, c_{qj} - 1)^2}, \tag{1}$$

where c_{pj} represents the connection between neuron i_p and neuron o_j (same for c_{qj}). Since two clusters with the shortest distance are merged in every iteration, the distance metric in Equation (1) can ensure the neurons connecting more common neurons are grouped into cluster.

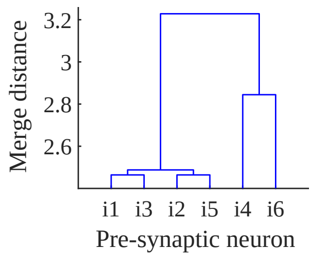


Figure 4: A hierarchical clustering dendrogram.

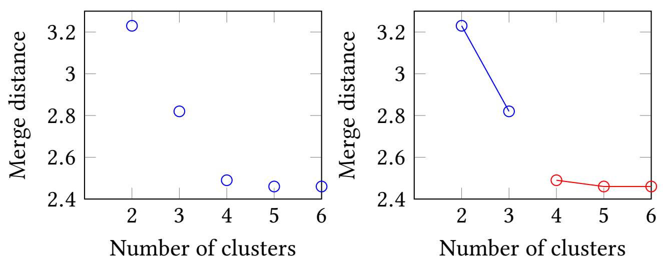


Figure 5: (a) The evaluation graph of the connection matrix shown in Fig. 1(a). (b) Two best-fit lines are obtained at the transition point x=3 by the L-method.

The flow of hierarchical clustering is shown in Algorithm 1. Starting from n data points, hierarchical clustering first treats each point as a single cluster (line 1). Then the iterative merging steps are performed n-1 times and eventually build a dendrogram (cluster tree) (lines 3–8). Therefore, the data points that are close will be merged first, and far away clusters will not be merged until the late iterations [18]. Figure 4 is used to illustrate the process of hierarchical clustering. For the connection matrix of one layer in a feed-forward neural network shown in Figure 1(a), it contains six pre-synaptic neurons and six post-synaptic neurons. During the first iteration, pre-synaptic neurons i_1 and i_3 are merged into a cluster since the distance between them is the smallest (connect three common post-synaptic neurons o_2, o_4, and o_6). Next, i_2 and i_5 are merged to form a cluster. Finally, two clusters $\{i_1, i_2, i_3, i_5\}$ and $\{i_4, i_6\}$ are merged into one cluster. The generated dendrogram is shown in Figure 4.

3.2 Optimal Number of Clusters

Although the hierarchical clustering can generate a cluster tree, however, if the number of clusters is not given, the hierarchical clustering cannot build the final clustering result. In this work, the L-method is proposed to find the optimal number of clusters. We summarize the major steps of the L-method for selection of the number of clusters in Algorithm 2. The L-method is based on an evaluation graph where the x-axis is the number of clusters and the y-axis is the evaluation metric used by the clustering algorithm. For the hierarchical clustering, the y-axis values are the merge distance between the last two merged clusters at x clusters. Since the hierarchical clustering merges a pair of clusters in every iteration, the evaluation graph can be produced by running the clustering algorithm only once (line 1). Note that the sharp transition point on the evaluation graph is considered to be the optimal number of clusters [19].

To illustrate how the L-method works, we use the connection matrix shown in Figure 1(a) as an example. Based on the dendrogram generated by hierarchical clustering, we plot the evaluation graph in Figure 5(a), where x-axis shows the number of clusters and y-axis represents the merge distance d. The L-method tries to determine the optimal number by searching for two lines that can best-fit the evaluation graph, and the intersection of these two lines represent the transition point of the graph. In Figure 5(b), the two lines can accurately fit the graph where one line fits the data in the interval $x \in [2, 3]$ and the other line fits the data in the interval $x \in [4, 6]$. It can be noticed that the sharp transition point is located at x=3, meaning that the neurons should be grouped into 3 clusters.

Algorithm 2 L-method for selection of cluster numbers

Input: A cluster tree generated by the hierarchical clustering.
Output: The optimal number of clusters $\hat{t}$.

```
1: Construct the evaluation graph based on the cluster tree;
2: for each point $t \leftarrow 3$ to $n-2$ do
3:     Solve the regression problem at $t$;          ▷ Equation (2)
4:     Calculate RMSE$_t$ at $t$;                     ▷ Equation (3)
5: end for
6: Determine $\hat{t}$;                              ▷ Equation (4)
```

Mathematically, the L-method can be formulated as the regression problem [18] (lines 2–5). Considering the evaluation graph where the number of clusters varies from 2 to n (n-1 data points in the graph), and the data points are partitioned into a left region and a right region at transition point x=t. The left region has points with x∈[2, t], and the right region has points with x∈[t+1, n]. To ensure each region contains at least two points, the value of t ranges from 3 to n-2. Then the regression problem is solved respectively to achieve two best-fit lines for the left and the right regions as follows:

$$\begin{aligned} \text{RMSE}_l &= \min_{s_l, b_l} \|\boldsymbol{d}_l - s_l \cdot \boldsymbol{x}_l - \boldsymbol{b}_l\|_2, \\ \text{RMSE}_r &= \min_{s_r, b_r} \|\boldsymbol{d}_r - s_r \cdot \boldsymbol{x}_r - \boldsymbol{b}_r\|_2, \end{aligned} \tag{2}$$

where $\boldsymbol{d}_l$ and $\boldsymbol{d}_r$ are the merge distance values for left region $\boldsymbol{x}_l \in [2, t]^\intercal$ and right region $\boldsymbol{x}_r \in [t+1, n]^\intercal$, respectively. ($s_l$, b_l) and (s_r, b_r) represent the slope and bias for left-fit and right-fit lines. RMSE$_l$ and RMSE$_r$ are the root mean square error of the two best-fit lines. Then the total root mean square error (RMSE$_t$) is defines as follows:

$$\text{RMSE}_t = \frac{t-1}{n-1} \cdot \text{RMSE}_l + \frac{n-t}{n-1} \cdot \text{RMSE}_r. \tag{3}$$

We solve the regression problems for all values of t, and the transition point that achieves the minimum RMSE$_t$ is assumed as the optimal number of clusters (line 6):

$$\hat{t} = \underset{t \in [3, n-2]}{\text{argmin}} \ \text{RMSE}_t. \tag{4}$$

After solving the regression problem for the evaluation graph in Figure 4, the transition point x=3 can achieve minimum total root mean square error for the two lines. The three clusters are $\{i_1, i_3\}$, $\{i_2, i_5\}$, and $\{i_4, i_6\}$, respectively.

4 EXPERIMENTAL RESULTS

We implement the hierarchical clustering and the L-method in MATLAB on a 2.20 GHZ 64 bit Windows 7 machine with 4 GB RAM. 3D floorplanning is implemented in C++ language on a 12-core 2.0 GHz Linux server with 64 GB RAM.

4.1 Impact of L-Method

In this work, based on the evaluation graph generated by hierarchical clustering, the L-method can find the optimal number of clusters. In

Table 1: Effectiveness of the proposed 3D-FNC.

Bench	Sparsity	AutoNCS [8]					3D-FNC w/o. LM					3D-FNC				
		Area (um^2)	Wire (um)	#TSV	util	RT(s)	Area (um^2)	Wire (um)	#TSV	util	RT(s)	Area (um^2)	Wire (um)	#TSV	util	RT(s)
n300	94.47%	3925.17	93398.64	353	0.65	20.37	4058.70	96424.28	349	0.45	6.46	3814.78	94389.73	358	0.67	5.21
n400	93.59%	7318.64	269163.17	427	0.76	35.28	7381.36	276821.31	420	0.57	13.23	7153.36	272542.70	426	0.77	11.17
n500	94.39%	10521.18	407018.25	491	0.70	69.43	10994.21	429515.68	477	0.47	32.14	10502.32	411046.51	485	0.68	29.62
avg.	94.15%	7255.00	256526.69	424	0.70	41.69	7478.09	267587.09	416	0.50	17.28	7156.82	259326.31	423	0.71	15.33
ratio	–	1.01	0.99	1.00	0.99	2.72	1.04	1.03	0.98	0.70	1.13	1.00	1.00	1.00	1.00	1.00

order to see the effect of the L-method, we compare the floorplanning results on the clustering results with and without the L-method. Since the size of crossbar, $s \times s$, is limited by the memristive technology, the maximum size of crossbars is set to 64×64 in this work [8]. Without the L-method, the number of clusters is determined by the way, in which iteratively increasing the number of clusters by one in hierarchical clustering until the size of the largest crossbar is below the size limitation. Then we run the 3D floorplanning on the clustering results of the case with and without L-method, respectively. The layer number is set to 2. The area of the neuron in $45nm$ technology is $2500um^2$ [20]. Besides each memristor and memristive crossbar use $4f^2$ and $s^2 \cdot 40f^2$ circuit area (f=feature size), respectively [11]. The experiment is tested on three Hopfield networks in [8] with the size of 300, 400, and 500. In Table 1, "3D-FNC w/o. LM" and "3D-FNC" list the average results. Column "util" gives the average utilization of all mapped crossbars. Columns "Area", "Wire" and "#TSV" represent chip area, total half-perimeter wirelength overhead, and the number of TSVs, respectively. We can see that in the case without L-method, the utilization of mapped crossbars are reduced by nearly 30%. In addition, area and wirelength are increased by around 4% and 3%, which demonstrates that the hierarchical clustering with L-method can result in highly hardware-efficient designs.

4.2 Comparison with Previous Work

We compare 3D-FNC with AutoNCS [8] on the hardware cost. In AutoNCS, the iterative spectral clustering is used for grouping the connections into the memristive crossbars. Since AutoNCS only generates 2D floorplanning output, to provide a fair comparison, we also implement a 3D floorplanning based on the iterative spectral clustering results of AutoNCS. The experiment is also tested on three Hopfield networks in [8] with the size of 300, 400, and 500. Table 1 lists the average statistic results. Column "RT" reports the total computational time in seconds. As shown in Table 1, compared with AutoNCS, 3D-FNC can achieve comparable area, wirelength cost and crossbar utilization, while the runtime of 3D-FNC is far less than AutoNCS. That is because in AutoNCS, the spectral clustering is executed in every iteration, thus the clustering method could be time consuming when the neural network is large.

5 CONCLUSION

In this paper, we have proposed an effective memristive crossbar mapping for neuromorphic computing systems on 3D IC, where both crossbar utilization and design cost are considered. Experimental results show that, compared with state-of-the-art, the proposed 3D-FNC can achieve highly hardware-efficient designs. Memristive crossbar gives hope for the anticipated efficient implementation of artificial neuromorphic networks, thus we expect to see a lot of researches to provide more efficient physical synthesis solutions.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China (NSFC) under grant No. 61732020 and The Research Grants Council of Hong Kong SAR (Project No. CUHK24209017). The authors would like to thank the Information Science Laboratory Center of USTC for hardware and software services, as well as Prof. Tsung-Yi Ho from National Tsing Hua University for providing test benches.

REFERENCES

[1] Y. Chen, H. H. Li, C. Wu, C. Song, S. Li, C. Min, H.-P. Cheng, W. Wen, and X. Liu, "Neuromorphic computing's yesterday, today, and tomorrow–an evolutional view," *Integration, the VLSI Journal*, 2017.

[2] A. Ankit, A. Sengupta, and K. Roy, "Trannsformer: Neural network transformation for memristive crossbar based neuromorphic system design," *arXiv preprint arXiv:1708.07949*, 2017.

[3] F. Akopyan, J. Sawada, A. Cassidy, and et.al, "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE TCAD*, vol. 34, no. 10, pp. 1537–1557, 2015.

[4] S. Acciarito, A. Cristini, G. Susi, and et.al, "Hardware design of lif with latency neuron model with memristive stdp synapses," *Integration, the VLSI Journal*, 2017.

[5] M. Hu, H. Li, Q. Wu, and G. S. Rose, "Hardware realization of bsb recall function using memristor crossbar arrays," in *Proc. DAC*, 2012, pp. 498–503.

[6] M. A. Ehsan, Z. Zhou, and Y. Yi, "Neuromorphic 3D integrated circuit: A hybrid, reliable and energy efficient approach for next generation computing," in *Proc. GLSVLSI*, 2017, pp. 221–226.

[7] H. An, M. A. Ehsan, Z. Zhou, F. Shen, and Y. Yi, "Monolithic 3D neuromorphic computing system with hybrid CMOS and memristor-based synapses and neurons," *Integration, the VLSI Journal*, 2017.

[8] W. Wen, C.-R. Wu, X. Hu, B. Liu, T.-Y. Ho, X. Li, and Y. Chen, "An eda framework for large scale hybrid neuromorphic computing systems," in *Proc. DAC*, 2015, pp. 1–6.

[9] C.-R. Wu, W. Wen, T.-Y. Ho, and Y. Chen, "Thermal optimization for memristor-based hybrid neuromorphic computing systems," in *Proc. ASPDAC*, 2016, pp. 274–279.

[10] J. Cui and Q. Qiu, "Towards memristor based accelerator for sparse matrix vector multiplication," in *Proc. ISCAS*, 2016, pp. 121–124.

[11] B. Liu, Y. Chen, B. Wysocki, and T. Huang, "The circuit realization of a neuromorphic computing system with memristor-based synapse design," in *Neural Information Processing*, 2012, pp. 357–365.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[13] Q. Xu and S. Chen, "Fast thermal analysis for fixed-outline 3D floorplanning," *Integration, the VLSI Journal*, vol. 59, pp. 157–167, 2017.

[14] P. H. Shiu, R. Ravichandran, S. Easwar, and S. K. Lim, "Multi-layer floorplanning for reliable system-on-package," in *Proc. ISCAS*, vol. 5, 2004, pp. V–69.

[15] S. Chen and T. Yoshimura, "Fixed-outline floorplanning: Enumerating block positions and a new objective function for calculating area costs," *IEEE TCAD*, vol. 27, no. 5, pp. 858–871, 2008.

[16] S. Chen, L. GE, M.-F. Chiang, and T. Yoshimura, "Lagrangian relaxation based inter-layer signal via assignment for 3-D ICs," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 92, no. 4, pp. 1080–1087, 2009.

[17] R. C. Dubes and A. K. Jain, *Algorithms for clustering data.* New Jersey: Prentice Hall Englewood Cliffs, 1988.

[18] M. B. Alawieh, F. Wang, and X. Li, "Identifying wafer-level systematic failure patterns via unsupervised learning," *IEEE TCAD*, 2017.

[19] S. Salvador and P. Chan, "Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms," in *Proc. ICTAI*, 2004, pp. 576–584.

[20] J.-s. Seo, B. Brezzo, Y. Liu, and et.al, "A 45nm CMOS neuromorphic chip with a scalable architecture for learning in networks of spiking neurons," in *Proc. CICC*, 2011, pp. 1–4.