DiffPattern: Layout Pattern Generation via Discrete Diffusion

Zixiao Wang[1*], Yunheng Shen[2*], Wenqian Zhao[1], Yang Bai[1], Guojin Chen[1], Farzan Farnia[1], Bei Yu[1]

[1]The Chinese University of Hong Kong, [2]Tsinghua University

Highlights

- We develop a novel layout pattern generation method based on discrete denoising for synthesizing layout topology.
- We propose a lossless layout pattern representation strategy, Deep Squish Pattern, which accelerates pixel-based layout pattern generation schemes.
- We utilize a nonlinear system for pattern assessment where the system provides a reliable method to legalize the layout patterns.
- We extensively evaluate our methodology on benchmark showing that **DiffPattern** can achieve SOTA performance.

Background

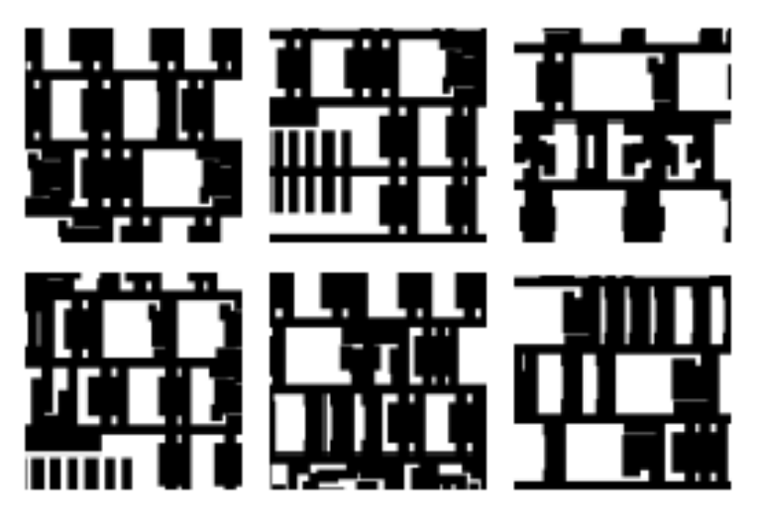
Figure 1. Realistic Layout Patterns[3]

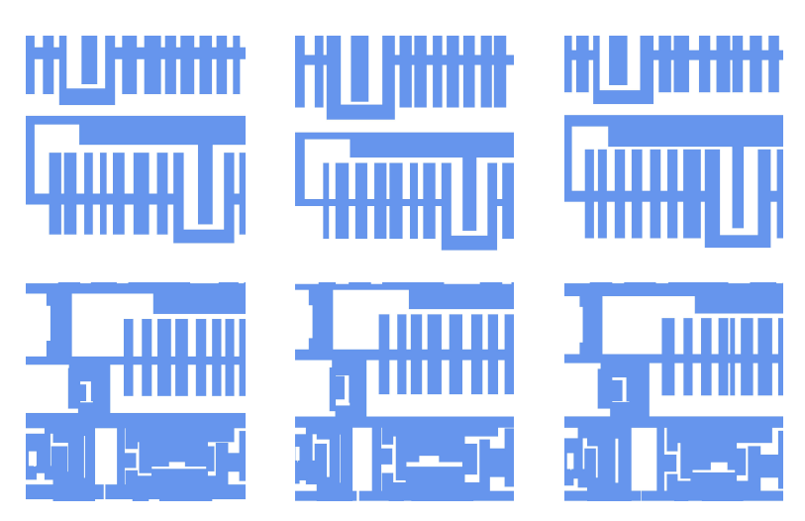
Figure 2. Generated Layout Patterns

Layout Pattern Generation. VLSI layout patterns provide critical resources in various designs for manufacturability research. The target of pattern generation is to synthesize novel and legal layout patterns.

Denoising Diffusion Probabilistic Models (DDPM). DDPM is a kind of strong and widely-used generative model in multiple domains.

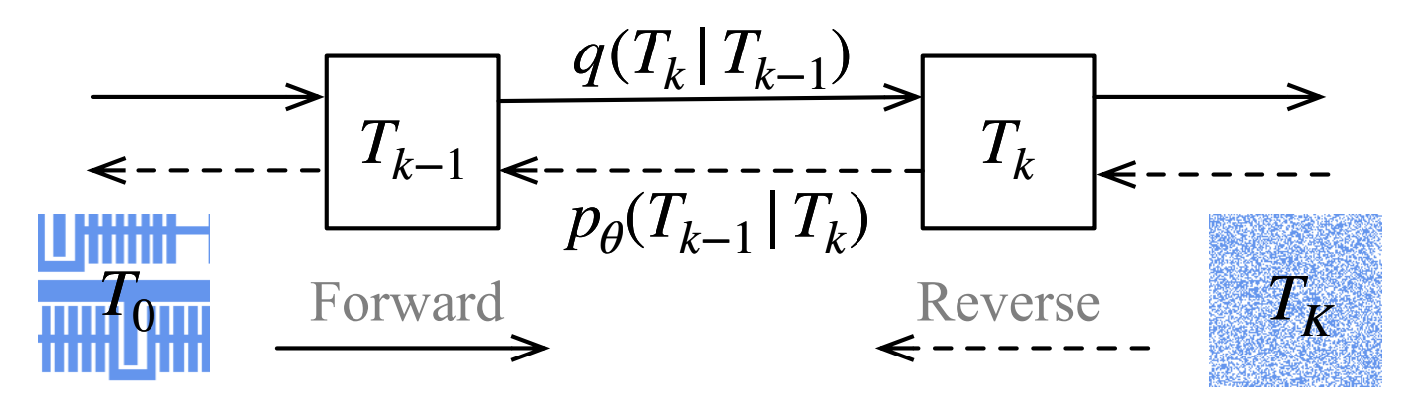

Figure 3. Illustration of denoising diffusion process.

Forward: $\boldsymbol{q}\left(\boldsymbol{T}_k|\boldsymbol{T}_{k-1}\right) := \mathcal{N}\left(\boldsymbol{T}_k; \sqrt{1-\beta_k}\boldsymbol{T}_{k-1}, \beta_k\boldsymbol{I}\right)$.

Reverse: $\boldsymbol{p}_{\boldsymbol{\theta}}\left(\boldsymbol{T}_{k-1}|\boldsymbol{T}_k\right) := \mathcal{N}\left(\boldsymbol{T}_{k-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}\left(\boldsymbol{T}_k, k\right), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\left(\boldsymbol{T}_k, k\right)\right)$.

Motivation

Existing Problems Three main challenges in current learning-based pattern generation methods:

- The information density of layout pattern representation is not satisfactory.
- Current methods generate continuous topology first and convert it into a binary one, which wastes model capacity.
- Learning-based pattern legalization is not reliable and unexplainable.

In this work, we address these challenges by,

- **Deep Squish Pattern Representation**: A more compact layout pattern representation method with balanced entry weight.
- **Discrete Topology Generation**: A discrete diffusion model to synthesize binary layout topology directly.
- **White-box Pattern Legalization**: A flexible and rule-based legalization for reliable pattern recovery.

Pipeline

As illustrated in Fig. 4, **DiffPattern** consists of three phases.

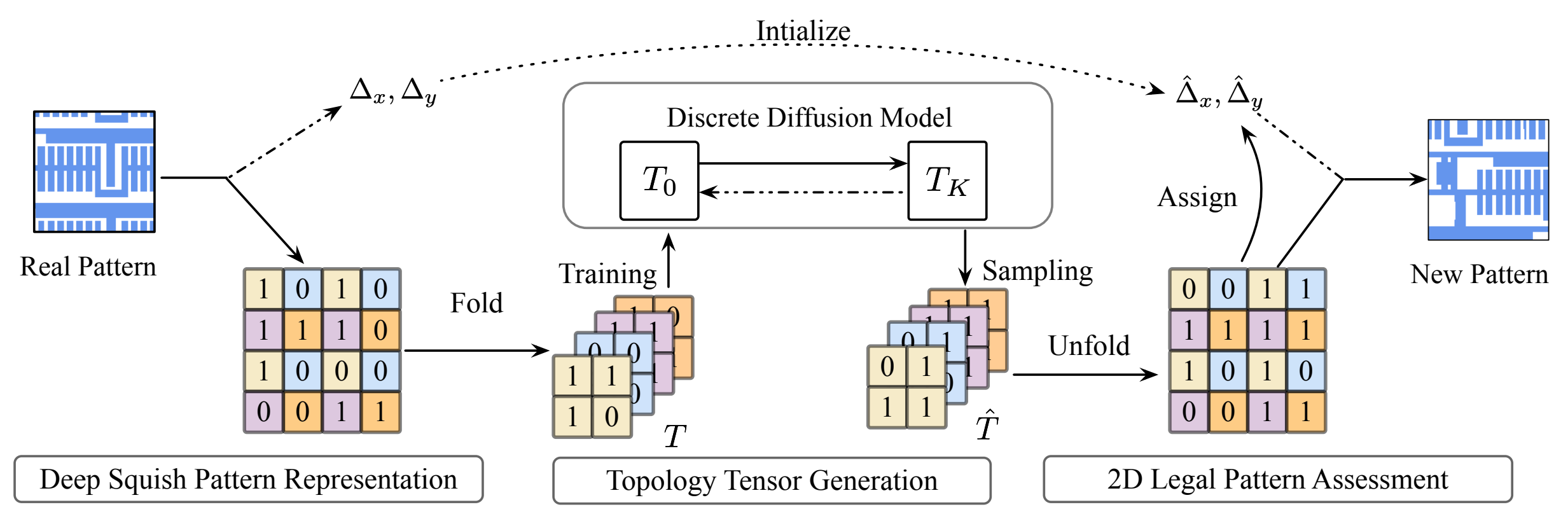

Figure 4. The framework of the proposed pattern generation method.

Methodology

Deep Squish Pattern Representation is a lossless and compact representation of the topology matrix, which allocates equal contribution to each entry in the topology matrix and folds the topology matrix into a tensor.

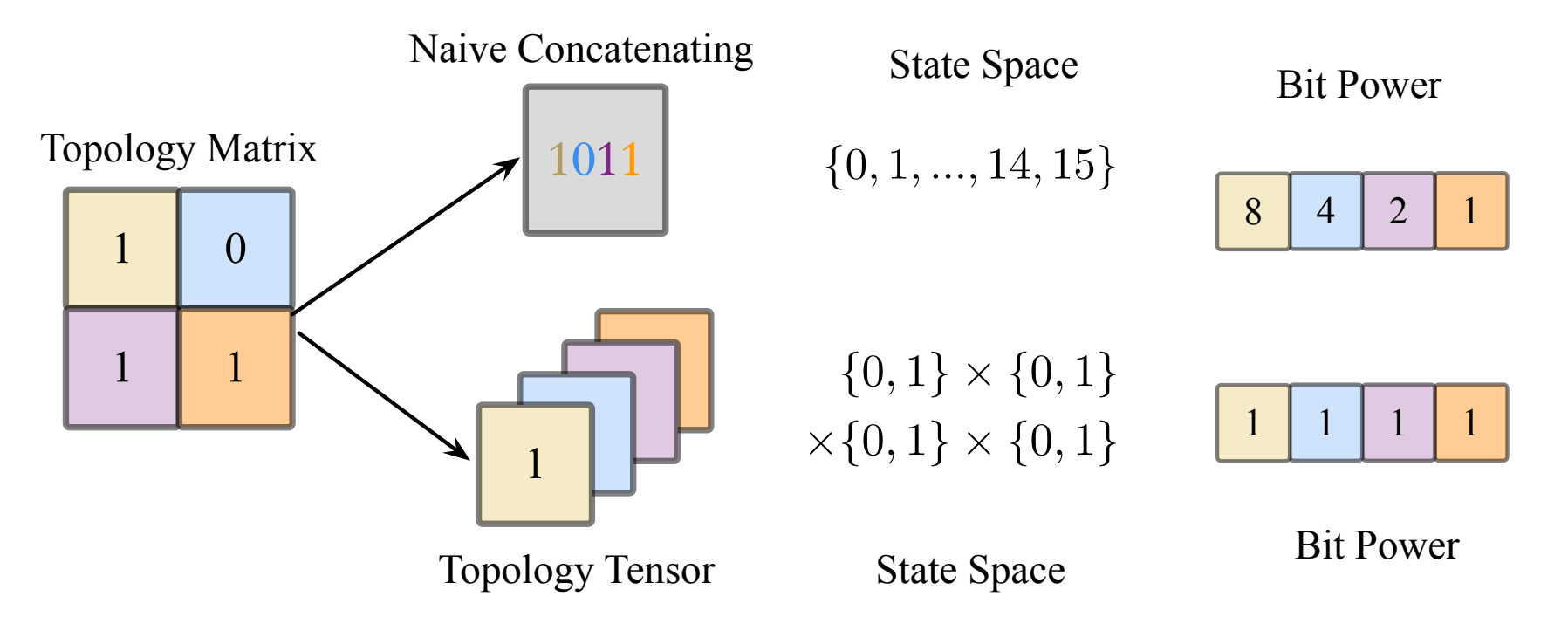

Figure 5. An illustration of Deep Squish Pattern Representation.

Topology Tensor Generation phase utilizes a discrete diffusion model, where every entry in the topology belongs to a discrete state. In forward process, every entry randomly flips between pre-defiend discrete states,

$$\boldsymbol{q}\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}\right) := \mathrm{Cat}\left(\boldsymbol{x}_k; \boldsymbol{p} = \boldsymbol{x}_{k-1}\boldsymbol{Q}_k\right), \tag{1}$$

where $\boldsymbol{Q}_k$ defines state transition probability for each $\boldsymbol{x}$ at the k-th diffusion step. In order to make sure we have $\boldsymbol{q}(\boldsymbol{x}_k|\boldsymbol{x}_0) \rightarrow [0.5, 0.5]$, when $k \rightarrow K$. We choose an increasing β_k and the transition matrix is defined as,

$$\boldsymbol{Q}_k = \begin{bmatrix} 1-\beta_k & \beta_k \\ \beta_k & 1-\beta_k \end{bmatrix}. \tag{2}$$

Therefore, discrete topology tensor can be directly generate from random noise by flipping entry state according to model prediction as shown in Fig. 6.

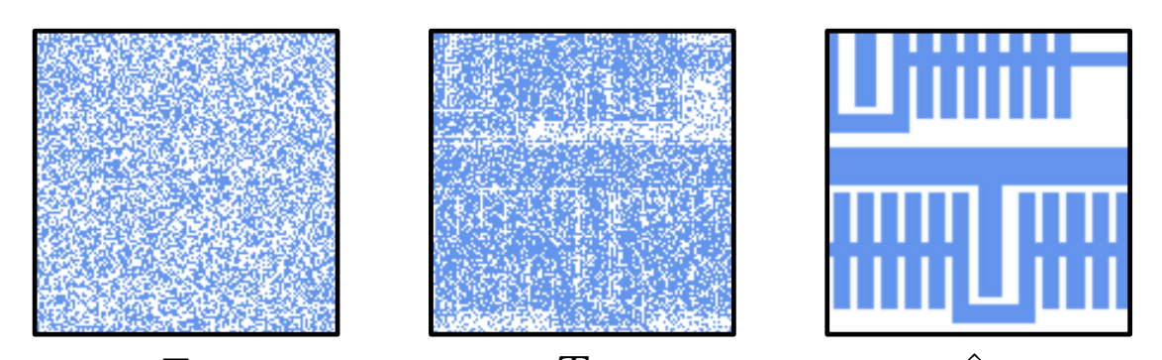

Figure 6. An illustration of the (flattened) samples from our Discrete Diffusion Model.

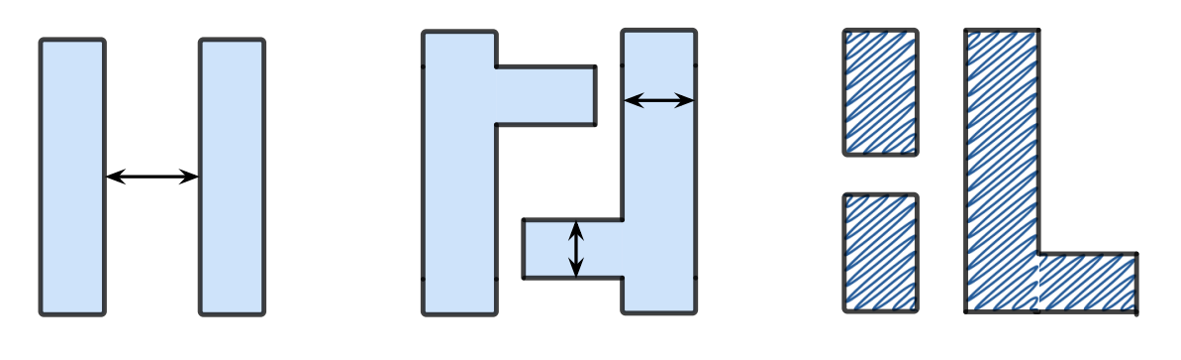

Figure 7. An illustration of design rules.

2D Legal Pattern Assessment translates each design rule into a sequence of linear/non-linear constraints. The solution is usually not unique and each legal solution of the non-linear system is a valid layout pattern.

$$\begin{cases} \delta_{xi}, \delta_{yj} > 0, & \forall \delta_{xi}, \delta_{yj}; \\ \sum \delta_{xi} = \sqrt{C}M, \quad \sum \delta_{yj} = \sqrt{C}M; & \\ \sum_{i=a}^{b} \delta_i \geq \mathit{Space}_{\mathit{min}}, & \forall (a,b) \in Set_S; \\ \sum_{i=a}^{b} \delta_i \geq \mathit{Width}_{\mathit{min}}, & \forall (a,b) \in Set_W; \\ \sum \delta_{xi}\delta_{yj} \in \left[\mathit{Area}_{\mathit{min}}, \mathit{Area}_{\mathit{max}}\right], & \forall \text{ Polygon}; \end{cases} \tag{3}$$

Evaluation Results

Table 1. Comparison on pattern diversity and legality. All results are copied from previous work. 'Real Patterns' refer to the whole dataset (Training set + Test set). '-' refer to Not Applicable.

Set/Method	Generated Topology	Legal Patterns Legality (↑)	Legal Patterns Diversity (↑)
Real Patterns	-	13869	10.777
CAE[2]	100000	19	3.7871
VCAE[3]	100000	2126	9.9775
CAE+LegalGAN[3]	100000	3740	5.8142
VCAE+LegalGAN[3]	100000	84510	9.8669
LayouTransformer[1]	10.532	89726	10.527
DiffPattern-S	100000	**100000**	**10.815**
DiffPattern-L	100000	**10000000**	**10.815**

Observations

- **Legality**: DiffPattern achieve perfect performance (i.e. 100%) under the metric of legality in the standard settings. With the well-designed topology generation method.
- **Diversity**: DiffPatten gets reasonable improvement (10.527→10.815) on the diversity of generated pattern compared with the previous best method.

Flexibility of Reliable Legalization The decomposition of topology generation and legal pattern assessment brings flexibility to DiffPattern. Without re-training the network, we can both generate different patterns from single topology Fig. 8 and generate legal patterns with different design rules Fig. 9.

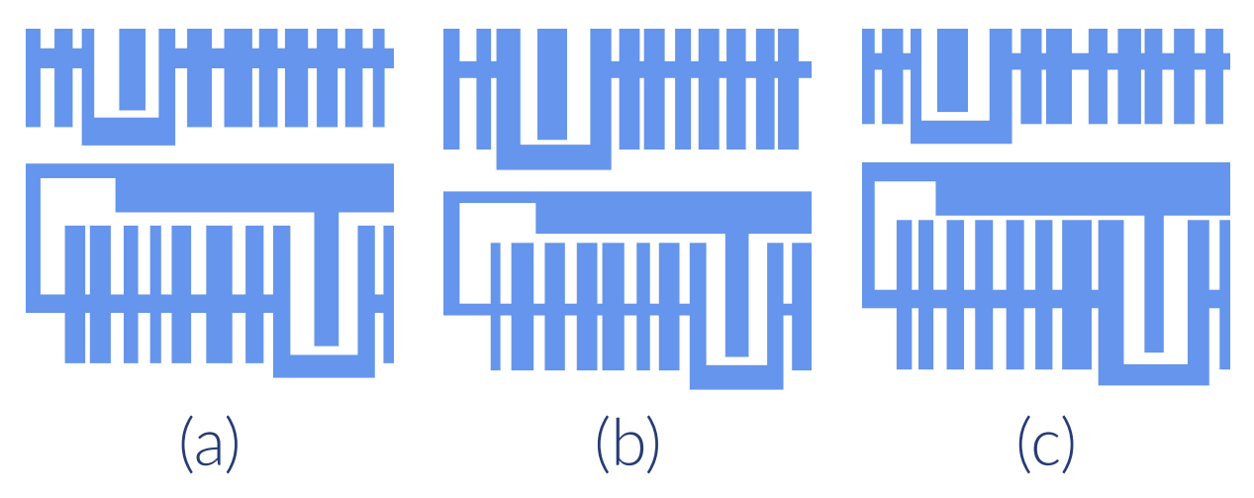

Figure 8. Different layout patterns that are generated from a single topology with the same design rule.

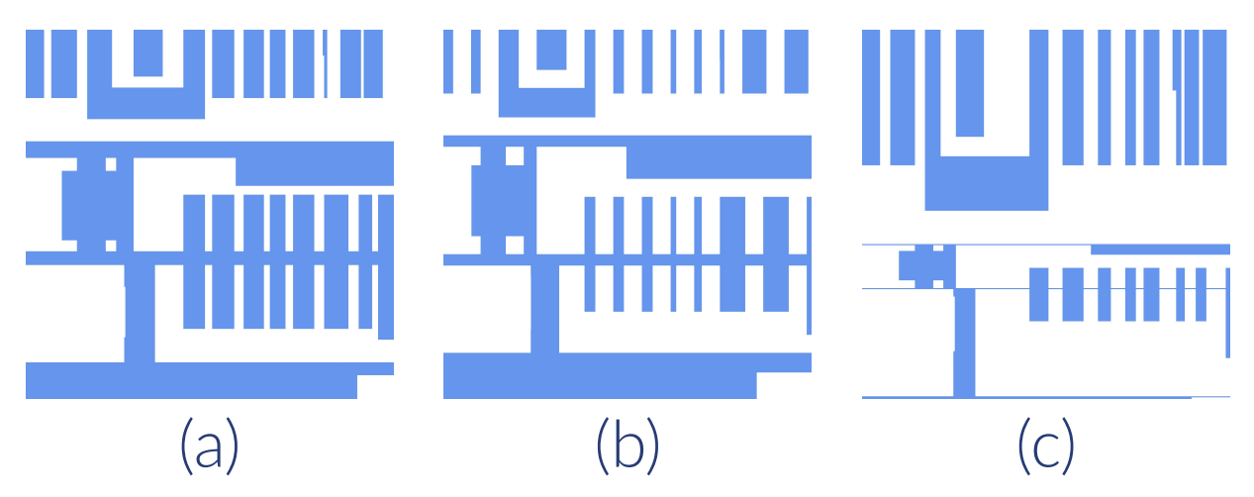

Figure 9. Layout patterns that are generated from the same topology with different design rules: (a) Normal rule; (b) Larger $\mathit{space}_{\mathit{min}}$; (c) Smaller $\mathit{Area}_{\mathit{max}}$.

Conclusion

We aim to synthesize diverse and legal layout patterns, and propose a practical method named DiffPattern. Based on the given design rules our approach allows us to stably generate theoretically infinite legal layout patterns.

References

[1] Liangjian Wen, Yi Zhu, Lei Ye, Guojin Chen, Bei Yu, Jianzhuang Liu, and Chunjing Xu. Layoutransformer: Generating layout patterns with transformer via sequential pattern modeling. In *ICCAD*. IEEE, 2022.

[2] Haoyu Yang, Piyush Pathak, Frank Gennari, Ya-Chieh Lai, and Bei Yu. Deepattern: Layout pattern generation with transforming convolutional auto-encoder. In *DAC*, pages 1–6, 2019.

[3] Xiaopeng Zhang, James Shiely, and Evangeline FY Young. Layout pattern generation and legalization with generative learning models. In *ICCAD*, pages 1–9. IEEE, 2020.