ChatPattern: Layout Pattern Customization via Natural Language

Zixiao Wang\textsuperscript{1}, Yunheng Shen\textsuperscript{2}, Xufeng Yao\textsuperscript{1}, Wenqian Zhao\textsuperscript{1}, Yang Bai\textsuperscript{1}, Farzan Farnia\textsuperscript{1}, Bei Yu\textsuperscript{1}

\textsuperscript{1}Chinese University of Hong Kong
\textsuperscript{2}Tsinghua University
Background
VLSI layout patterns provide critical resources in various designs for manufacturability research. Pattern Generation task aims to mimic the distribution of existing patterns.
The requirements on layout pattern distributions can vary in real cases. Pattern Customization task aims to generate patterns to meet specialized requirements.
Let’s Employ a LLM

- Training a LLM from scratch? **NO**, Too expensive.
- Utilizing Pre-trained LLM? **Yes, but, how can LLM get access to the Layout Patterns?**
  - Encoding a pattern as a sequence of direction and distance?
  - Embedding a pattern as a pattern token?
  - Manipulating pattern-generation tools?

Legal Pattern

Tokenizer & Detokenizer

Illegal Pattern
ChatPattern
ChatPattern seamlessly integrates a front-end powered by a Large Language Model with a back-end that employs a conditional discrete diffusion model for layout pattern generation.
ChatPattern seamlessly integrates a front-end powered by a Large Language Model with a back-end that employs a conditional discrete diffusion model for layout pattern generation.
The LLM agent is designed to communicate with users via natural language, and is able to:

- Auto-Format Requirement
- Plan and Execute Task
- Learn and Apply Tool Functions
- Learn from Documents and Experience
To construct a pattern library, certain fundamental tools or APIs are indispensable:

- Random Topology Generation
- Topology Legalization

---

To construct a pattern library, certain fundamental tools or APIs are indispensable:

- **Conditional Topology Generation**
- Topology Legalization\(^1\)
- **Topology Modification**
- **Topology Extension**

---

Property-Conditional Topology Generation

\[ p_\theta(T_0|T_K, c) = p_\theta(T_0|T_1, c) \prod_{k=2}^{K} p_\theta(T_{k-1}|T_k, c), \quad \text{(1)} \]

\[ L = D_{KL} (q(x_{k-1}|x_k, x_0) \parallel p_\theta(x_{k-1}|x_k, c)) - \lambda \log p_\theta(x_0|x_k, c). \quad \text{(2)} \]
Property-Conditional Topology Generation

\[ p_\theta(T_0|T_K, c) = p_\theta(T_0|T_1, c) \prod_{k=2}^{K} p_\theta(T_{k-1}|T_k, c), \]  

(1)

\[ L = D_{KL} (q(x_{k-1}|x_k, x_0) \parallel p_\theta(x_{k-1}|x_k, c)) - \lambda \log p_\theta(x_0|x_k, c). \]  

(2)
Pattern Modification

\[
\begin{align*}
T_{k-1}^{\text{known}} & \sim q \left( T_{k-1}^{\text{known}} | T_{0}^{\text{known}} \right), \\
T_{k-1}^{\text{unknown}} & \sim p_\theta \left( T_{k-1}^{\text{unknown}} | T_{k}, c \right), \\
T_{k-1} & = M \odot T_{k-1}^{\text{known}} + (1 - M) \odot T_{k-1}^{\text{unknown}},
\end{align*}
\]
Pattern Modification

Existing Pattern

\[ T_k \]

\( T_{k-1}^{\text{known}} \sim q \]

w/ Mask

\[ T_{k-1}^{\text{unknown}} \sim p_\theta \]

w/ \((1\text{-Mask})\)

Next Iteration

\[ T_{k-1}^{\text{known}} \sim q \left( T_{k-1}^{\text{known}} \mid T_0^{\text{known}} \right), \]

\[ T_{k-1}^{\text{unknown}} \sim p_\theta \left( T_{k-1}^{\text{unknown}} \mid T_k, c \right), \]

\[ T_{k-1} = M \odot T_{k-1}^{\text{known}} + (1 - M) \odot T_{k-1}^{\text{unknown}}, \]
Pattern Modification

\[
\begin{align*}
    T_{k-1}^{\text{known}} & \sim q \left( T_{k-1}^{\text{known}} | T_0^{\text{known}} \right), \\
    T_{k-1}^{\text{unknown}} & \sim p_\theta \left( T_{k-1}^{\text{unknown}} | T_k, c \right), \\
    T_{k-1} & = M \odot T_{k-1}^{\text{known}} + (1 - M) \odot T_{k-1}^{\text{unknown}},
\end{align*}
\]
Pattern Extension

- Working Space
- Existing Pattern
- Newly Generated

Extend by Out-Painting
Extend by In-Painting
Example-Pipeline

#1 Agent Setting
You are a layout designer and are required to produce a well-designed layout pattern according to the user's requirements. There are some rules you must follow:

- (Rules)

#2 Tool Learning
During the design process, you have access to the following functions:

- (functions and descriptions)

#3 Document Learning
There is a standard working pipeline you can refer to:

- (Standard Pipeline)
- (Documents)

#4 User Requirement
Generate a layout pattern library, there are 100k layout patterns in total. The physical size fixed as 1.5um * 1.5um. The topology size should be chosen from 200*200 and 500*500. They should be in style of 'Layer-10001'.

#5.1 Standard Arguments
- Physical size: [1500,1500] nm
- Topology size: [200,200]
- Style: Layer-10001
- Count: 50000
- Extending Method: None
- Drop Allowed: True

#5.2 Standard Arguments
- Physical size: [1500,1500] nm
- Topology size: [500,500]
- Style: Layer-10001
- Count: 50000
- Extending Method: None
- Drop Allowed: True

#6.1 Task Planning and Function Call
- # generate basic topology with fixed size
  topology = topology_gen(seed, style)

- # extend topology to desired size
  topology = extension(topology, [200,200])

- # first attempt to legalize the topology
  layout, failed, log = legalize(topology,[1500,1500])

- # modification un-solvable region for failed case
  topology = modification(failed_topology, style)

- When Processing Finished) Return layout

#6.2 Task Planning and Function Call
- Return layout
#1 Agent Setting
You are a layout designer and are required to produce a well-designed layout pattern according to the user's requirements. There are some rules you must follow:

(Rules)

#2 Tool Learning
During the design process, you have access to the following functions:

(functions and descriptions)

#3 Document Learning
There is a standard working pipeline you can refer to: (Standard Pipeline)

(Documents)

#4 User Requirement
Generate a layout pattern library, there are 100k layout patterns in total. The physical size fixed as 1.5um * 1.5um. The topology size should be chosen from 200*200 and 500*500. They should be in style of 'Layer-10001'.

#5 Standard Arguments

#5.1 Standard Arguments
Physical size: [1500,1500] nm
Topology size: [200,200]
Style: Layer-10001
Count: 50000
Extending Method: Out
Drop Allowed: True

#5.2 Standard Arguments
Physical size: [1500,1500] nm
Topology size: [500,500]
Style: Layer-10001
Count: 50000
Extending Method: None
Drop Allowed: True

#6 Task Planning and Function Call

#6.1 Task Planning and Function Call

# generate basic topology with fixed size
topology = topology_gen(seed, style)

# extend topology to desired size
topology = extension(topology, [200,200])

# first attempt to legalize the topology
layout, failed, log = legalize(topology, [1500,1500])

# modification un-solvable region for failed case
topology = modification(failed_topology, style)

#6.2 Task Planning and Function Call

(Return layout)

#7 Finished
Summary results and return, save history if necessary

Example-Pipeline
#1 Agent Setting
You are a layout designer and are required to produce a well-designed layout pattern according to the user's requirements. There are some rules you must follow:

- Physical size: [1500,1500] nm
- Topology size: [500,500]
- Style: Layer-10001
- Count: 50000
- Extending Method: None
- Drop Allowed: True

#2 Tool Learning
During the design process, you have access to the following functions:

- Functions and descriptions

#3 Document Learning
There is a standard working pipeline you can refer to: (Standard Pipeline)
There are some experience you can refer to: (Documents)

#4 User Requirement
Generate a layout pattern library, there are 100k layout patterns in total. The physical size fixed as 1.5um * 1.5um. The topology size should be chosen from 200*200 and 500*500. They should be in style of 'Layer-10001'.

#5 Standard Arguments

## 5.1 Standard Arguments
- Physical size: [1500,1500] nm
- Topology size: [200,200]
- Style: Layer-10001
- Count: 50000
- Extending Method: Out
- Drop Allowed: True

## 5.2 Standard Arguments
- Physical size: [1500,1500] nm
- Topology size: [500,500]
- Style: Layer-10001
- Count: 50000
- Extending Method: None
- Drop Allowed: True

#6 Task Planning and Function Call

## 6.1 Task Planning and Function Call
- Generate basic topology with fixed size: topology = topology_gen(seed, style)
- Extend topology to desired size: topology = extension(topology, [200,200])
- First attempt to legalize the topology: layout, failed, log = legalize(topology, [1500,1500])
- Modification un-solvable region for failed case: topology = modification(failed_topology, style)

## 6.2 Task Planning and Function Call
- Return layout (When Processing Finished) Return layout

#7 Finished
Summary results and return, save history if necessary
Experiments
• Pattern Diversity. Shannon entropy of the pattern complexity.

\[ H = - \sum_i \sum_j P(c_{xi}, c_{yj}) \log P(c_{xi}, c_{yj}), \]  

(4)

• Pattern Legality.

\[ L = \frac{\# \text{ Legal Patterns}}{\# \text{ All Patterns}}. \]  

(5)
# Free-size Pattern Generation

<table>
<thead>
<tr>
<th>Task</th>
<th>Set/Method</th>
<th>Training Set*</th>
<th>Size</th>
<th>Layer-10001</th>
<th>Layer-10003</th>
<th>Total†</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Legality (†)</td>
<td>Diversity (†)</td>
<td>Legality (†)</td>
</tr>
<tr>
<td>Fixed-size</td>
<td>Real Patterns</td>
<td>/</td>
<td>128²</td>
<td>/</td>
<td>10.731</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>CAE+LegalGAN [ICCAD’20]</td>
<td>Layer-10001</td>
<td></td>
<td>3.74%</td>
<td>5.814</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>VCAE+LegalGAN [ICCAD’20]</td>
<td>Layer-10001</td>
<td></td>
<td>84.51%</td>
<td>9.867</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>LayoutTransformer [ICCAD’22]</td>
<td>Layer-10001</td>
<td></td>
<td>8.97%</td>
<td>10.527</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>ChatPattern</td>
<td>Layer-10001/10003</td>
<td></td>
<td>99.97%</td>
<td>10.796</td>
<td>99.99%</td>
</tr>
<tr>
<td>Free-size</td>
<td>Real Patterns</td>
<td>/</td>
<td>256²</td>
<td>/</td>
<td>12.702</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>[DAC’23] w/ Concatenation</td>
<td>Layer-10001/10003</td>
<td></td>
<td>57.78%</td>
<td>10.179</td>
<td>93.69%</td>
</tr>
<tr>
<td></td>
<td>ChatPattern</td>
<td>Layer-10001/10003</td>
<td></td>
<td>87.36%</td>
<td>11.154</td>
<td>99.78%</td>
</tr>
<tr>
<td></td>
<td>Real Patterns</td>
<td>/</td>
<td>512³</td>
<td>/</td>
<td>13.435</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>[DAC’23] w/ Concatenation</td>
<td>Layer-10001/10003</td>
<td></td>
<td>0.29%</td>
<td>5.714</td>
<td>40.83%</td>
</tr>
<tr>
<td></td>
<td>ChatPattern</td>
<td>Layer-10001/10003</td>
<td></td>
<td>36.42%</td>
<td>10.401</td>
<td>98.86%</td>
</tr>
<tr>
<td></td>
<td>Real Patterns</td>
<td>/</td>
<td>1024²</td>
<td>/</td>
<td>13.573</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>[DAC’23] w/ Concatenation</td>
<td>Layer-10001/10003</td>
<td></td>
<td>0.00%</td>
<td>0.000</td>
<td>0.64%</td>
</tr>
<tr>
<td></td>
<td>ChatPattern</td>
<td>Layer-10001/10003</td>
<td></td>
<td>1.19%</td>
<td>6.438</td>
<td>94.96%</td>
</tr>
</tbody>
</table>

## Insights

When pattern size is larger than 256², the legality of ChatPattern sometimes can reach more than 100× higher than DiffPattern [DAC’23].
Example of $256 \times 256$ topology matrix generated by In-Painting.
Out-Painting

(a) Layer-10001 style

(b) Layer-10003 style

512×512 topology matrix generated by Out-Painting.
User Input

Generate a layout pattern library, there are 100k layout patterns in total. The physical size fixed as 1.5um * 1.5um. The topology size should be chosen from 200*200 and 500*500. They should be in style of ‘Layer-10001’.

# Requirement - subtask 1
## Basic Part: Topology Size: [200, 200], Physical Size: [1500, 1500] nm, Style: Layer-10001, Count: 50000,
## Advanced Part: Extension Method: Out (Default: Out), Drop Allowed: True (Default: True), Time Limitation: None (Default: None).
We present a case where the LLM agent adeptly navigates unanticipated mistakes during legalization. It is noteworthy that such error-handling expertise is not pre-coded within the standard operational procedures or the experiential documentation.

% In previous attempts, one pattern can’t pass the legalization.
Thought: Since legalization has failed twice in the same region, I will try to in-paint that specific area with the same style and then attempt legalization again.
Action: Topology_Modification
Action Input: "topology_path":${path}, "upper": 12, "left": 56, "bottom": 33, "right": 73, "style": "Layer-10001", "seed": 42
% Continue Processing
Thanks!