MLCAD Workshop 2022

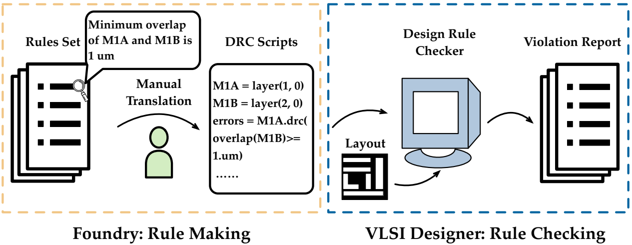
Efficient Design Rule Checking Script Generation via Key Information Extraction

Binwu Zhu[1], Xinyun Zhang[1], Yibo Lin[2], Bei Yu[1], Martin Wong[1]

[1]The Chinese University of Hong Kong
[2]Peking University

Sep. 12, 2022

Background and Motivation



- **Rule making**: Manufacturers first specify the essential design rules based on their manufacturing capability and then convert them into executable DRC scripts manually.
- **Rule checking**: These scripts are provided to the designer and will be input into a design rule checker, such as KLayout, to verify the correctness of the layout design.

Challenges of DRC

- With the continuous scaling-down of circuit feature size, rules to be checked for layout are increasing, i.e., thousands of rules should be checked for 7nm design. The whole process is very time-consuming.
- Some design rules can be very complicated, i.e., with complex logic, which may easily lead to misunderstanding.
- Different checkers require different script languages, which means all scripts must be re-implemented when transferring to other checkers.

KLayout Script

```
M1A = layer(1, 0)
M1B = layer(2, 0)
errors = M1A.drc(
overlap(M1B)>=
1.um)
```

Guardian Script

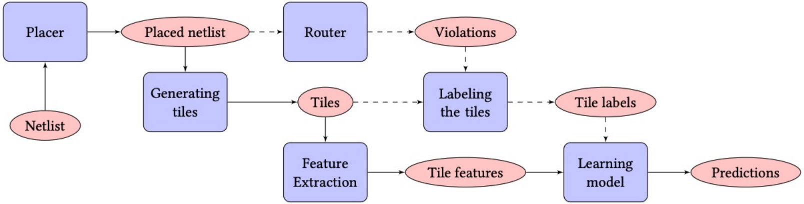
```
Layer:
  M1A (1, 0),
  M1B (2, 0)
InDistance:
  layer1 = M1A,
  layer2 = M1B,
  limits >= 1
```

LayoutEditor Script

```
M1A = 1
M1B = 2
layout->drcTool->
minimumOverlap
(M1A, M1B, 1)
```

Prior Work

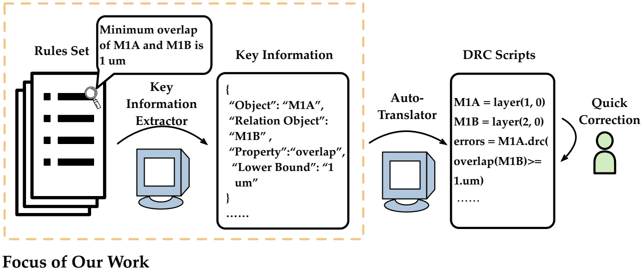
- Current DRC related works have tried to replace the traditional DRC checker with CNN based models to make a performance prediction.



Violation Detection Framework[1]

- However, practical industrial designs still rely on design rule checkers, which requires DRC scripts converted from natural language rules.

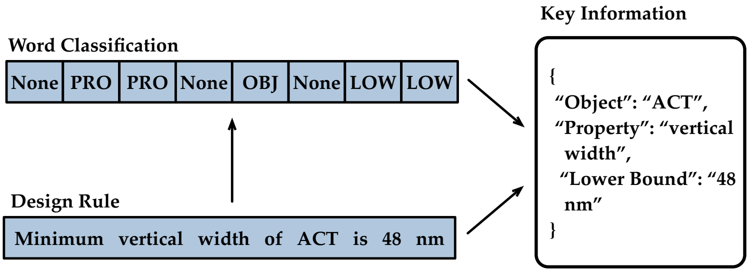
[1] Aysa Fakheri Tabrizi et al. (2018). "A Machine Learning Framework to Identify Detailed Routing Short Violations from a Placed Netlist". In: *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, pp. 1–6.



- A deep learning based key information extractor is designed to automatically identify the essential arguments of the scripts.
- The followed script translator will organize the extracted arguments into the final scripts. When switching to other checkers, we can keep the extractor unchanged and only need to modify the translator.
- The final manual correction is to ensure the complete correctness of the scripts.

Method

- Extracting key information from natural language design rules can be converted to such a problem that a specific word should be classified into a particular category.
- The category is closely related to its semantic role in rules.



- Key information extraction is converted to a word classification problem.

Semantic Roles

- All semantic roles are illustrated as follows:

Semantic Roles	Meanings	Examples
Object	Target layer of checking rules.	Minimum vertical width of **ACT** is 48 nm.
Relation Object	Additional layer that have relationships with target layer .	Minimum extension of GATEAB past **ACT** is 38 nm.
Property	Property to be checked of the target layer.	Minimum **vertical width** of ACT is 48 nm.
Condition	Logical conditions for particular layers.	GATEC shape bottom or top must be aligned if **distance is less than 192 nm**.
Restriction	Geometric restrictions that layers should follow.	GIL may **not bend**.
Lower Bound	Minimum value of the property to be checked.	Minimum vertical width of ACT is **48** nm.
Upper Bound	Maximum value of the property to be checked.	Maximum distance of GATEAB to neighboring shape is **236** nm.
Exact Value	Exact value of the property to be checked.	Exact horizontal spacing of ACT is **80** nm.

Rule Data Generation - Motivation

- Open-source design rules for academic research are relatively rare. Our dataset, FreePDK15[2], only includes around 130 rules, which is not enough for model training.
- Different from image data augmentation techniques such as rotation and clipping, we are supposed to ensure all generated rules are both syntactically and semantically correct.
- As our task is a classification task, semantic role labels need to be assigned to each word, which is extremely expensive.

[2] Kirti Bhanushali (May 2014). "Design Rule Development for FreePDK15: An Open Source Predictive Process Design Kit for 15nm FinFET Devices". PhD thesis.

Rule Data Generation - Methods

- Word Order Adjustment

Original	V0 shape must be rectangular if enclosing M1[A | B] wire width is equal or greater than 40nm
Reordered	If enclosing M1[A | B] wire width is equal or greater than 40nm, V0 shape must be rectangular

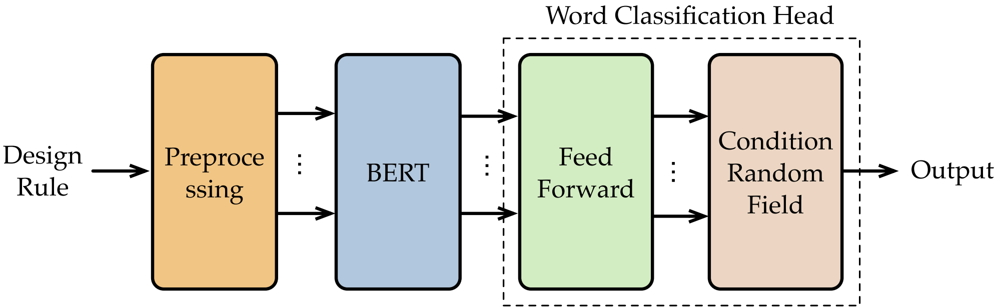
- Paraphrasing

Original	Minimum vertical width of ACT is 48 nm.
Synonym replacement	Vertical width of ACT is at least 48 nm.

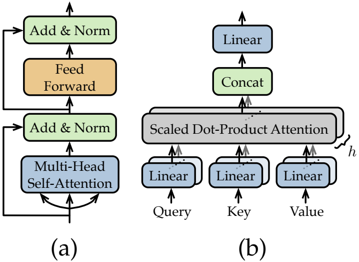
- Template Filling

Minimum	**Property**	of	**Object**	is	**Lower Bound**	nm

- Objective: Identify the essential arguments from design rules to generate the final scripts.
- Extractor Architecture



- Recently, Transformer[3] has made much progress in language processing tasks.



(a) Transformer Encoder; (b) Multi-Head Self-Attention.

- $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n \times d_m}$, $\{\mathbf{Q}, \mathbf{K}, \mathbf{V}\} = \{\mathbf{X}\mathbf{W}^Q, \mathbf{X}\mathbf{W}^K, \mathbf{X}\mathbf{W}^V\}$
- $\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}\left(\mathbf{H}_1, \ldots, \mathbf{H}_h\right)\mathbf{W}^O$, where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{n \times d_m}$
- $\mathbf{H}_i = \text{Attention}\left(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V\right) = \text{softmax}\left[\mathbf{Q}\mathbf{W}_i^Q \left(\mathbf{K}\mathbf{W}_i^K\right)^\top / \sqrt{d_k}\right]\mathbf{V}\mathbf{W}_i^V$.

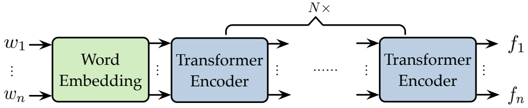
[3] Ashish Vaswani et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems*. Vol. 30.

Preprocessing & Backbone

- **Preprocessing:** Split the rule $\boldsymbol{r}$ into a list of words and extend list length to L by padding a special word "[PAD]", since different rules vary in length.

$$[w_1, w_2, \ldots, w_{len(\boldsymbol{r})}, \underbrace{\text{[PAD]}, \ldots, \text{[PAD]}}_{L-len(\boldsymbol{r})}] = \text{Preprocess}(\boldsymbol{r})$$

- **Backbone:** Determining the semantic role of each word is closely related to its sentence, and one word may have different semantic roles in different rules.



Bidirectional Encoder Representations from Transformers (BERT)

- BERT[4] is a strong language representation model constructed from Transformer and it has been fully pretrained by various language tasks on large datasets.

[4]Jacob Devlin et al. (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding". In:

Word Classification Head

- Word Classification Head is composed of a two-layer MLP and a condition random field (CRF) module.

$$\boldsymbol{P}^{wc} = \mathrm{MLP}(\boldsymbol{F}^{o}) \in \mathbb{R}^{L \times N_{wc}}$$

- Although a two-layer MLP can finish the classification task, it does not consider the label relationship within the rule.

Minimum extension of **GATEAB** past **ACT** is 38 nm.

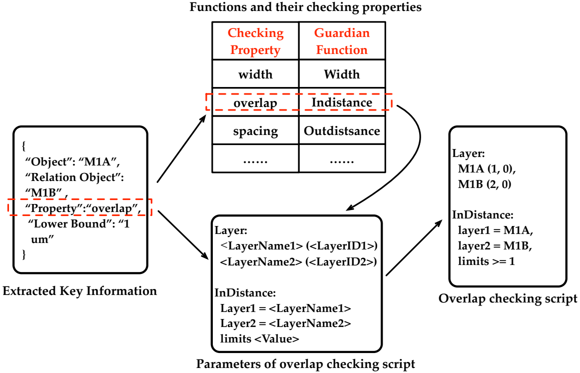
- CRF is a probabilistic model with parameter $\boldsymbol{K} \in \mathbb{R}^{(N_{wc}+2)\times(N_{wc}+2)}$, which describes the probability of transiting from i to j. Two additional states represent "Start" and "End". Relying on CRF, the probability of a sequence is computed as follows:

$$S_{\boldsymbol{r}}(\boldsymbol{y}) = \sum_{i=0}^{L} \boldsymbol{K}_{\boldsymbol{y}_i, \boldsymbol{y}_{i+1}} + \sum_{i=0}^{L} \boldsymbol{P}^{wc}_{i, \boldsymbol{y}_i}.$$

$$p(\boldsymbol{y}|\boldsymbol{r}) = \frac{\exp S_{\boldsymbol{r}}(\boldsymbol{y})}{\sum_{\hat{\boldsymbol{y}} \in \boldsymbol{Y}_{\boldsymbol{r}}} \exp S_{\boldsymbol{r}}(\hat{\boldsymbol{y}})},$$

Second Stage of Our Flow - Script Translator

- Objective: Organize the extracted arguments into the final scripts.
- Translator Architecture



- DRC script is composed of function calling statements. To conveniently search the required function, we can pair the properties and functions together.
- To automatically pass the key information to the function, we need to connect the parameters of different functions with our semantic roles.

Experimental Results

Benchmark Statistics

- Training Set: FreePDK15[5] and augmented data. Test Set: ASAP7[6]

Semantic Roles	Training Set		Test Set	
	#	Percent (%)	#	Percent (%)
Object	6054	15.68	491	14.74
Relation Object	2561	6.63	181	5.43
Property	4705	12.18	300	9.01
Condition	5061	13.10	596	17.89
Restriction	1404	3.64	159	4.77
Lower Bound	2482	6.43	326	9.79
Upper Bound	1144	2.96	8	0.24
Exact Value	1430	3.70	40	1.20
None	13778	35.68	1230	36.93
Total	38619	100	3331	100

[5]Kirti Bhanushali (May 2014). "Design Rule Development for FreePDK15: An Open Source Predictive Process Design Kit for 15nm FinFET Devices". PhD thesis.

[6]Lawrence T Clark et al. (2016). "ASAP7: A 7-nm finFET predictive process design kit". In: *Microelectronics Journal* 53, pp. 105–115.

Word Classfication Results

- Since there are almost no prior related works, we implement two widely used models for NLP tasks, Bi-RNN[7] and Bi-LSTM[8], as baseline.

Categories	Bi-RNN			Bi-LSTM			Ours		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Object	0.609	0.666	0.636	0.774	0.662	0.714	0.853	0.804	0.828
Relation Object	0.436	0.674	0.529	0.422	0.674	0.519	0.849	0.896	0.872
Property	0.879	0.970	0.922	0.894	0.953	0.923	0.892	0.900	0.896
Condition	0.786	0.767	0.776	0.669	0.757	0.710	0.818	0.838	0.828
Restriction	0.389	0.453	0.419	0.538	0.403	0.460	0.789	0.704	0.744
Lower Bound	0.947	0.871	0.907	0.960	0.883	0.920	0.967	0.907	0.936
Upper Bound	0.500	1.000	0.667	0.429	0.750	0.545	0.889	1.000	0.941
Exact Value	0.371	0.650	0.473	0.750	0.900	0.818	0.741	1.000	0.851
None	0.874	0.714	0.775	0.893	0.825	0.858	0.892	0.894	0.893
Average	0.649	0.759	0.685	0.703	0.756	0.719	**0.853**	**0.880**	**0.863**
Ratio	0.761	0.863	0.794	0.824	0.859	0.833	**1.000**	**1.000**	**1.000**

[7] Jie Zhou and Wei Xu (2015). "End-to-end learning of semantic role labeling using recurrent neural networks". In: pp. 1127–1137.

[8] Matthew E Peters et al. (2018). "Deep contextualized word representations". In: *Proceedings of NAACL-HLT*, pp. 2227–2237.

- The key information extractor averagely takes only **5.0ms** to process a single rule from ASAP7 dataset.
- The script translator is simply responsible for deciding which function to call and passing the arguments. It spends only **0.46ms** processing one item of key information.
- In conclusion, by combining the extractor and translator, our framework can generate a single script in **5.46ms** on average.

THANK YOU!