

Reinforcement learning assisted Automated analog Layout design Flow

- Chapter 1+4 (Overview+PDK)-

AIST Solutions 岡村 淳一

第1章 (概要)

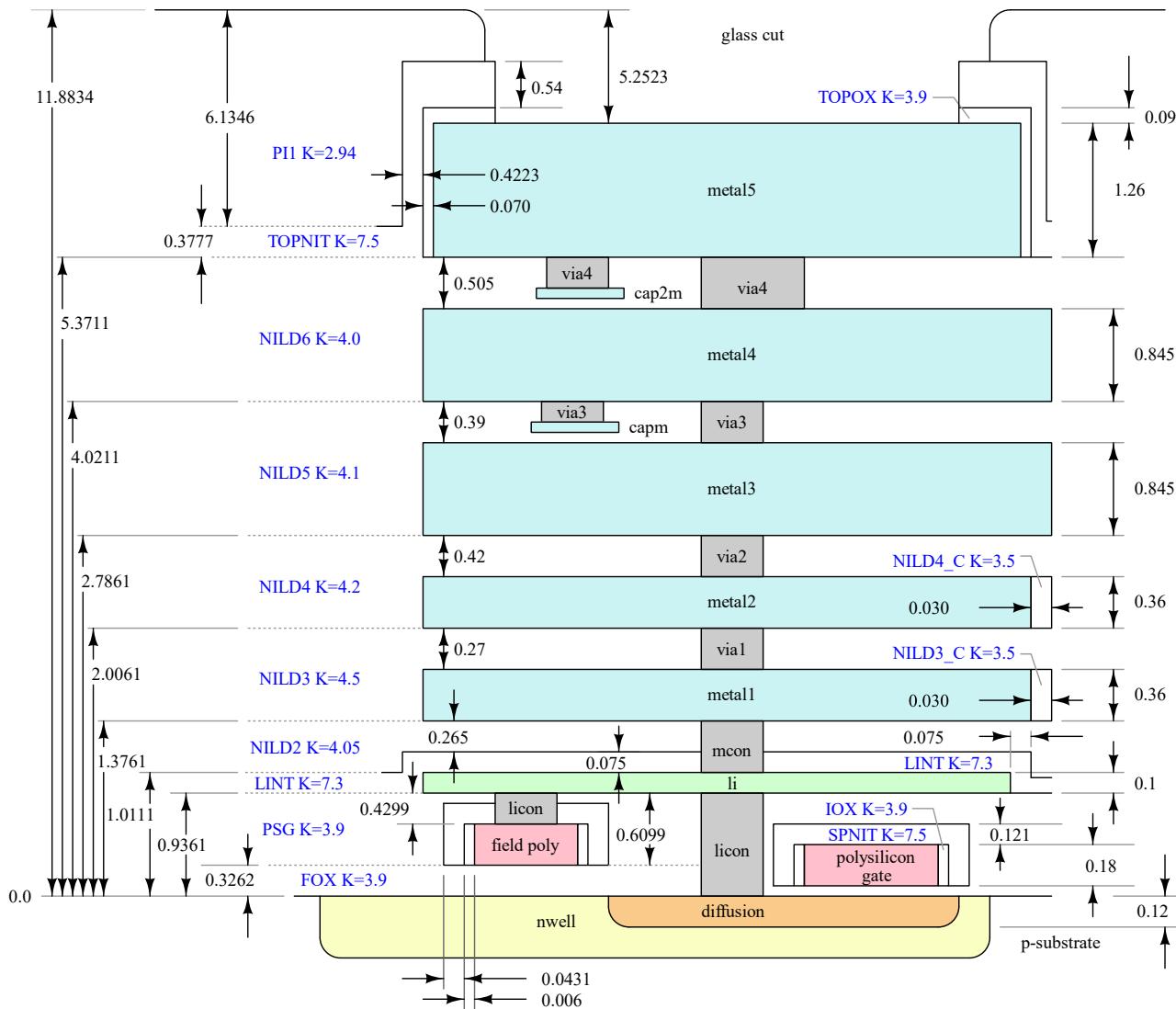
本書は、アナログ回路レイアウトのための自動オープンソースEDAツールの実装詳細を紹介することで、アナログツールの領域に貢献することを目的としています。このツールは、SKY130 PDK およびオープンソースVLSIレイアウトツール「Magic」を使用する、強化学習支援による自動アナログレイアウト設計フロー(RALF)に基づいています。すべてのソースコードはオープンソースで、GitHubで公開されています。

Reinforcement learning assisted
Automated analog
Layout design
Flow

ラルフ（発音は /rælf/ または /reɪf/）[1] はゲルマン起源の男性名で、古英語の Rædwulf と古ドイツ語の Radulf に由来し、古ノルウェー語の Raðulfr (rað 「顧問」と ulfr 「狼」) と同語源です。

第1章 (Sky130nm PDK)

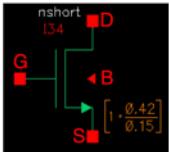
(Diagram not to scale!)



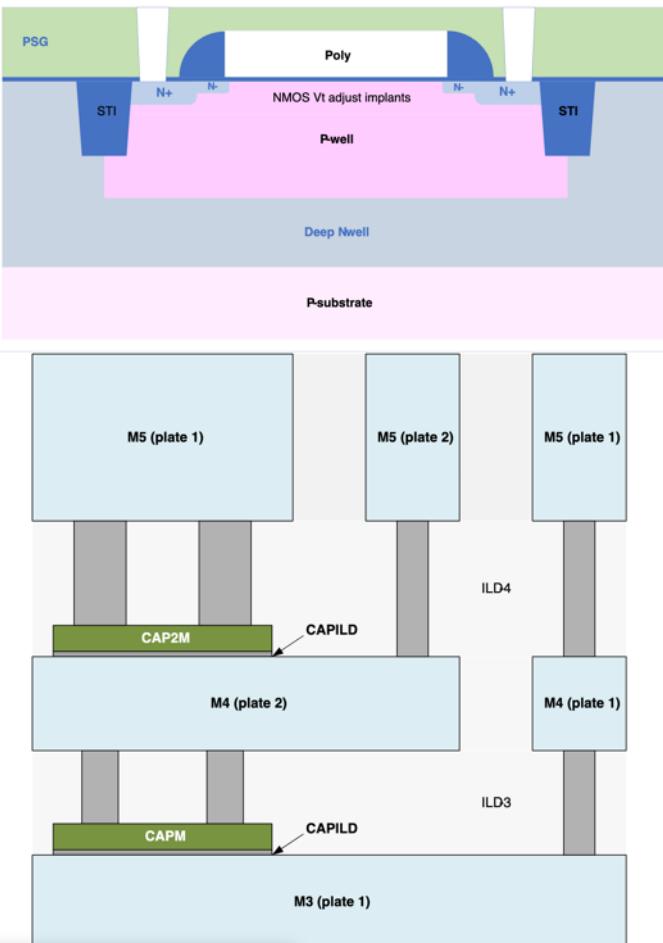
<https://github.com/google/skywater-pdk/blob/main/docs/rules/device-details.rst>

第1章 (Sky130nm PDK)

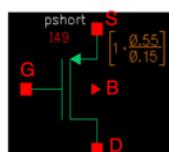
The symbol of the `:model:'sky130_fd_pr_nfet_01v8'` (1.8V NMOS FET) is shown below:



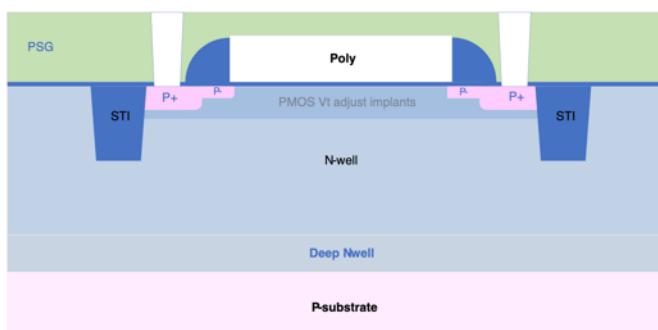
The cross-section of the NMOS FET is shown below:



The symbol of the `:model:'sky130_fd_pr_pfet_01v8'` (1.8V PMOS FET) is shown below:



The cross-section of the PMOS FET is shown below:



P- poly precision resistors

Spice Model Information

- Cell Name: `:cell:'res_xhigh_po_XpXX'`, `:cell:'sky130_fd_pr_res_xhigh_po'`
- Model Type: subcircuit

Operating ranges where SPICE models are valid

- $|V_{r0} - V_{r1}| = 0$ to 5.0V
- Currents up to 500 μ A/ μ m of width (preferred use $\leq 100 \mu$ A/ μ m)

Details

The resistors have 5 different fixed widths, plus a variable W/L option.

- 0.35 (0p35)
- 0.69 (0p69)
- 1.41 (1p41)
- 2.85 (2p83)
- 5.73 (5p73)

They are modeled as subcircuits, using a conventional resistor model combined with the capacitance under the resistor, as well as matching parameters and temperature coefficients. The fixed-width resistors may only be used in the above configurations. Each resistor end is contacted using a slot licon. Length is variable and measured between the front ends of the slot licons.

The resistors are modeled using the same equations as for the P+ poly resistors. In the case of the P- poly resistors, a separate implant is used to set the sheet resistance to 2000 ohm/sq.

Fixed value resistors have the same layout footprints as their P+ poly counterparts. Electrical and e-test specs are still TRD once sufficient silicon has been evaluated. More details on the use of the precision resistors, and their

<https://github.com/google/skywater-pdk/blob/main/docs/rules/device-details.rst>

第1章 (Sky130nm PDK)

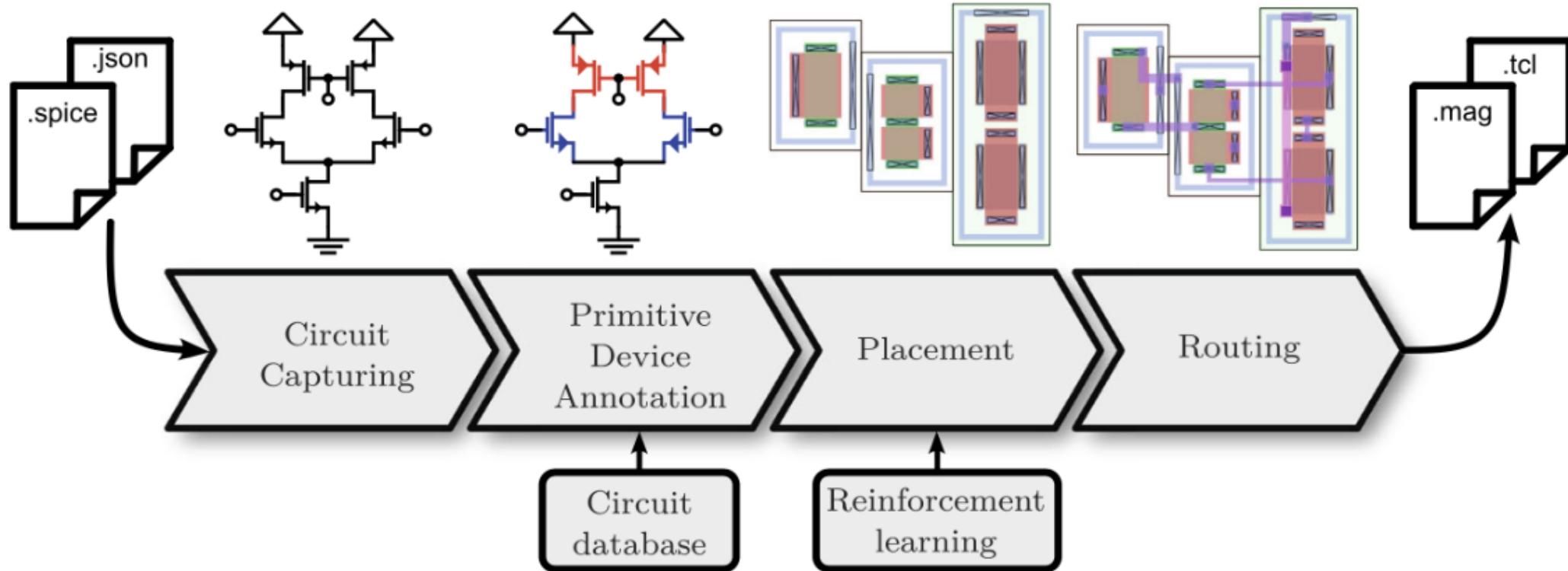


図 1.2: 強化学習支援による自動アナログレイアウト設計フロー

PDKとは？

【PDKの定義の確認】

Level1：ファブと会話して作るもの

1. SPICEモデル
2. DRC/LVS/LPE ルール
3. シンボルライブラリ

ファブの情報とEDAツールの橋渡し情報

Technology File

Level2：L1で作る基本部品

1. スタンダードセル
2. IO セル
3. SRAM

Logic chip を作る為の基本部品

Cell Library

回路設計の知識が必要

RALF では Level#1 の情報を使う

【DRC】

デザインルール = 微細加工に関するルール

Layer : 加工対象層 = 素材

配線 = メタル・ポリシリコン等

穴 = SiO₂

拡散層 = Well・アクティブ等

ルール : Line/Space

穴径・フリンジ・スペース

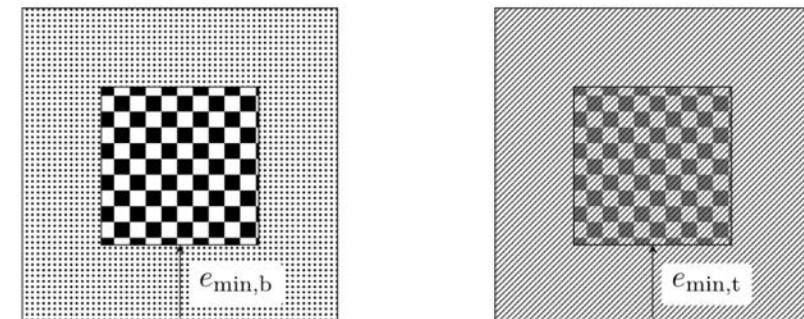
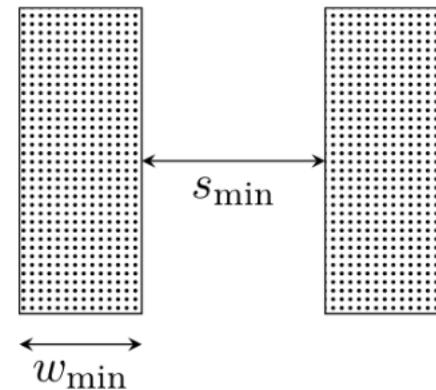
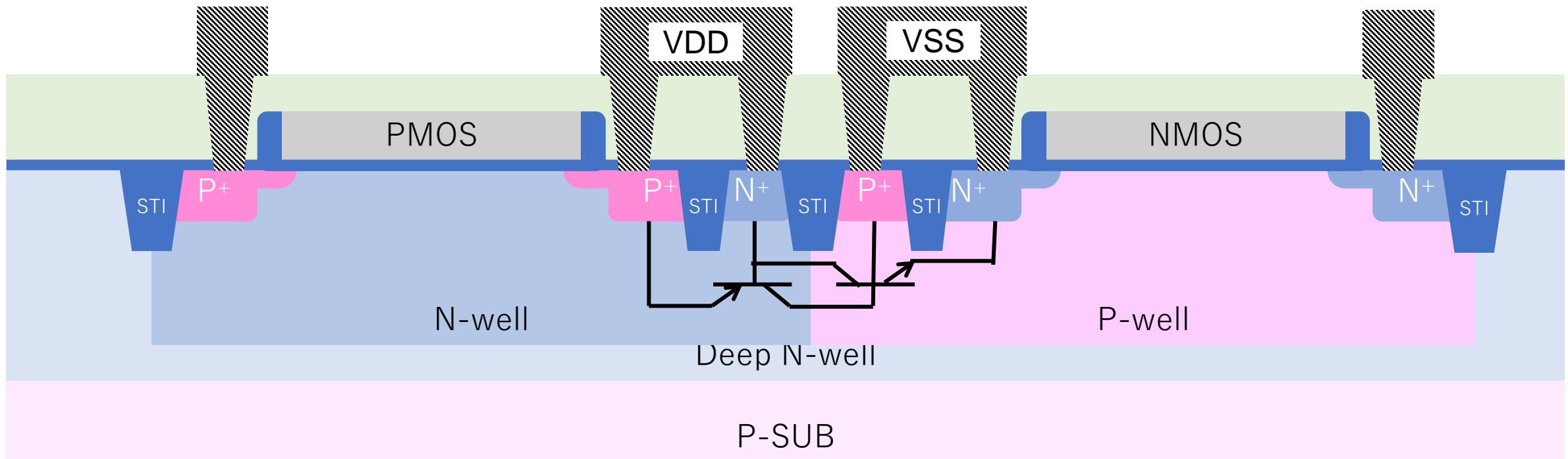


図 4.1: VIA に対する上下層の最小フリンジルールの例

第4章 (Placement Rule)

【DRC】

拡散層 = Well・アクティブが律速 ラッチアップ



第4章 (PDK.py layers.json)

The image shows two adjacent browser windows side-by-side.

Left Window (PDK.py): The title bar says "IIC-RALF/PDK/PDK.py at main". The page content displays the Python file "PDK.py". The code defines a class "PDK" with methods for initializing layers, getting scale factors, and returning metal and via layers as dictionaries. It also includes methods for device layers and layers, and handles layer aliases.

```
25
26     class PDK:
27         """Class to store a PDK.
28             A PDK defines the layers for the routing.
29         """
30     def __init__(self, pdk_file_path : str | Path) -> None:
31         self._init_layers()
32
33     @property
34     def scale_factor(self) -> int:
35         return self._scale_factor
36
37     @property
38     def metal_layers(self) -> dict[str, Layer]:
39         return self._metal_layers
40
41     @property
42     def via_layers(self) -> dict[str, Layer]:
43         return self._via_layers
44
45     @property
46     def device_layers(self) -> dict[str, Layer]:
47         return self._device_layers
48
49     @property
50     def layers(self) -> dict[str, Layer]:
51         return self._layers
52
53     def __init_layers(self):
54         self._layers.update(self._via_layers)
55
56     def _get_name_from_alias(self, layer_alias : str) -> str | None:
57         return None
58
59     def get_layer(self, layer : str) -> Layer:
60         raise ValueError(f"Layer {layer} not in PDK!")
61
62     def get_layer_number(self, layer : str) -> int:
63         raise ValueError(f"Layer {layer} has no number!")
64
65     def get_lower_metal_layer(self, layer : str) -> MetalLayer:
66         return None
```

Right Window (layers.json): The title bar says "IIC-RALF/PDK/layers.json at main". The page content displays the JSON file "layers.json". The file defines layer mappings and specific layer properties like width, space, and resistivity.

```
1 {
2     "ScaleFactor": 10,
3     "device_layers": ["nwell", "dnwell", "mimcap"],
4     "layer_stack": ["l1", "m1", "m2", "m3", "m4", "m5"],
5     "via_stack": ["mcon", "vial1", "via2", "via3", "via4"],
6     "aliases": {
7         "l1": ["l1i", "locali"],
8         "m1": ["met1", "metali"],
9         "m2": ["met2", "metal2"],
10        "m3": ["met3", "metal3"],
11        "m4": ["met4", "metal4"],
12        "m5": ["met5", "metal5"]
13    },
14    "via_map": {
15        "lim1": "mcon",
16        "m1i": "mcon",
17        "m1m2": "vial1",
18        "m2m1": "vial1",
19        "m2m3": "via2",
20        "m3m2": "via2",
21        "m3m4": "via3",
22        "m4m3": "via3",
23        "m4m5": "via4",
24        "m5m4": "via4"
25    },
26    "l1": {
27        "Width": 180,
28        "Space": 180,
29        "MinArea": 57600,
30        "Resistivity": 125
31    }
32},
33{
34    "mcon": {
35        "Stack": ["l1", "m1"],
36        "Space": 190,
37        "Width": 180,
38        "min_enclosure": 60,
39        "Resistivity": 11925
40}
```

第4章 (PlacementRule.py RoutingRule.py)

The image shows two side-by-side browser windows displaying Python code. Both windows have a header bar with a back/forward button, refresh, search, and other standard browser controls. The left window is titled "IIC-RALF/Rules/PlacementRules.py" and the right window is titled "IIC-RALF/Rules/RoutingRules.py".

IIC-RALF / Rules / PlacementRules.py

```
Code Blame 309 lines (250 loc) · 10.4 KB
22 from typing import TYPE_CHECKING
23
24 if TYPE_CHECKING:
25     from Magic.Cell import Cell
26     from Magic.MacroCell import MacroCell
27     from PDK.Layers import Layer
28     from SchematicCapture.Net import Net
29
30     import abc
31     from collections import deque
32
33     class PlacementRules:
34         """Class to store placement rules for a cell.
35         """
36     def __init__(self, cell : Cell, rules:list[PlacementRule]) -> None:
37         self._rules = rules
38
39         @property
40         def cell(self) -> Cell:
41             return self._cell
42
43         @property
44         def rules(self) -> list[PlacementRule]:
45             return self._rules
46
47         def generate_rule(self, other_rules : PlacementRules) -> tuple[float, float, float]:
48             bound = ...
49             return bound
50
51
52         class PlacementRule(metaclass = abc.ABCMeta):
53             return tuple()
54
55         class Spacing(PlacementRule):
56             return tuple(self.cell.get_bounding_box())
57
58         class MacroSpacing(Spacing):
59             ...
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
```

IIC-RALF / Rules / RoutingRules.py

```
Code Blame 102 lines (82 loc) · 3.17 KB
7 # you may not use this file except in compliance with the License.
8 # You may obtain a copy of the License at
9 #
10 # http://www.apache.org/licenses/LICENSE-2.0
11 #
12 # Unless required by applicable law or agreed to in writing, software
13 # distributed under the License is distributed on an "AS IS" BASIS,
14 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
15 # See the License for the specific language governing permissions and
16 # limitations under the License.
17 # SPDX-License-Identifier: Apache-2.0
18 =====
19
20 from __future__ import annotations
21
22 from typing import TYPE_CHECKING
23
24 if TYPE_CHECKING:
25     from Magic.Cell import Cell
26     from PDK.Layers import Layer
27
28     import abc
29     from Rules.Rule import Rule
30
31     class RoutingRule(Rule, metaclass = abc.ABCMeta):
32         return self._cell
33
34     class AreaRule(RoutingRule, metaclass = abc.ABCMeta):
35         raise NotImplementedError
36
37     class LayerRule(AreaRule, metaclass = abc.ABCMeta):
38         return self._layer
39
40     class ObstacleRule(LayerRule):
41         return tuple(area)
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
```

第4章 (net_rule_DiffAmp.json net_rule.json)

The image shows two side-by-side browser windows displaying JSON configuration files for a net rule system.

Left Tab (net_rules_DiffAmp.json):

- File Path: IIC-RALF/NetRules/net_rules_...
- Author: JakobRat
- Blame: 30 lines (30 loc) · 461 Bytes
- Content:

```
[{"MinNetWireWidth": {"net": "vmid", "min_width": 28}, {"MinNetWireWidth": {"net": "Vss", "min_width": 28}, {"MinNetWireWidth": {"net": "Vdd", "min_width": 28}, {"Ports": {"nets": ["Vss", "Vdd", "Vp", "Vn", "Vbp", "Vbn"]}, {"PowerNets": {"nets": ["Vdd", "Vss"]}}}], [{"Ports": {"nets": ["Vss", "Vdd", "Vp", "Vn", "Vbp", "Vbn", "Vop", "Von"]}], [{"PowerNets": {"nets": ["Vdd", "Vss"]}}]
```

Right Tab (net_rules.json):

- File Path: IIC-RALF/NetRules/net_rules.json
- Author: JakobRat
- Blame: 18 lines (18 loc) · 279 Bytes
- Content:

```
[{"MinNetWireWidth": {"net": "vmid", "min_width": 20}, {"Ports": {"nets": ["Vss", "Vdd", "Vp", "Vn", "Vbp", "Vbn", "Vop", "Von"]}}, {"PowerNets": {"nets": ["Vdd", "Vss"]}}]
```

Reinforcement learning assisted Automated analog Layout design Flow

- Chapter 5 (Placement)-

AIST Solutions 岡村 淳一

第5章 (RALF の肝)

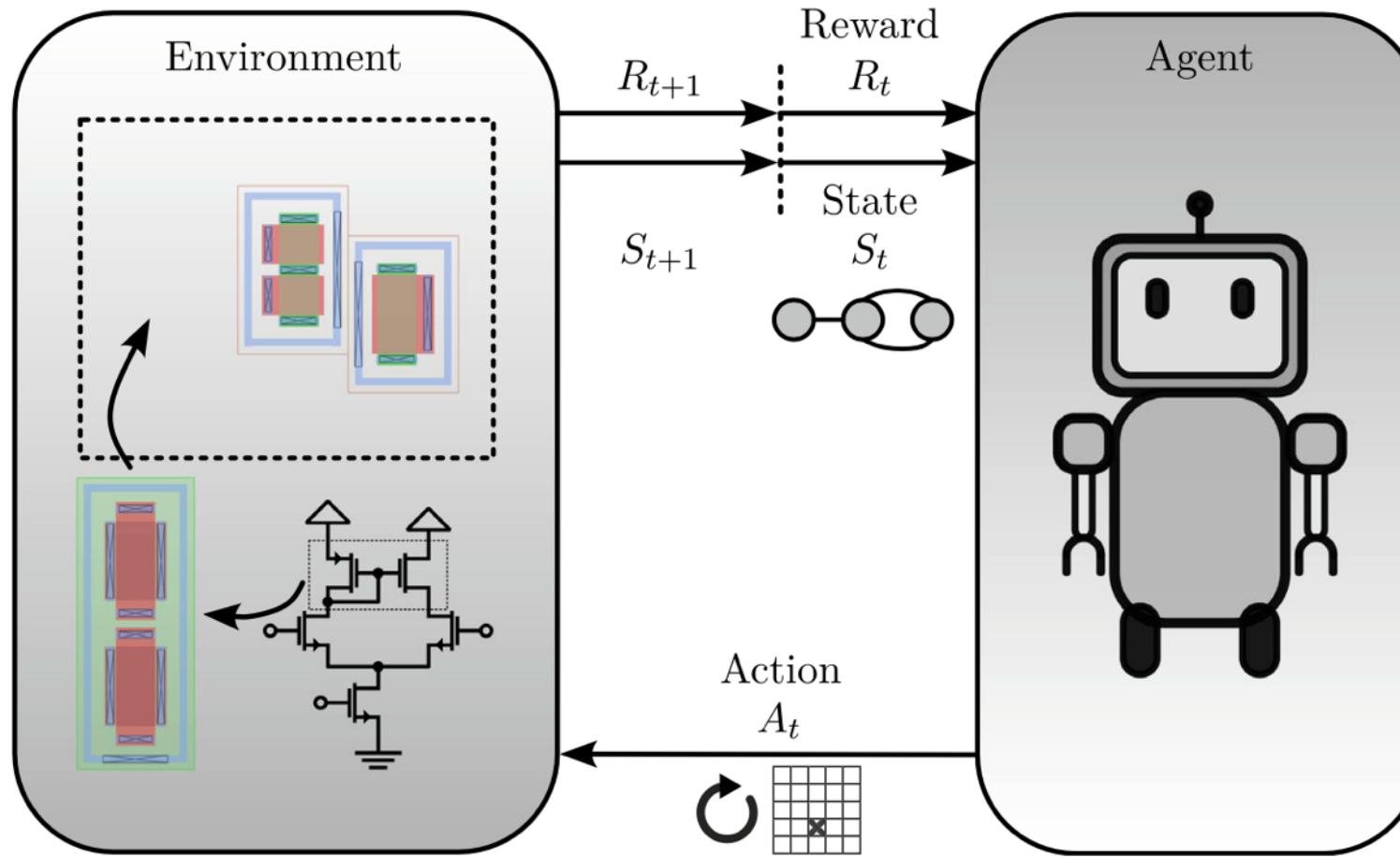
強化学習 (Reinforcement Learning) を用いてレイアウトの配置 (placement)を行います。

エージェント(Agent)は、素子を順番にレイアウトへ配置することで、与えられた回路の配置 (Environment)問題を解決するポリシー (policy)を学習します。

この学習(Learning)プロセスでは、推定配線長や配線混雑度を評価指標(Reward)として、複数の試行を繰り返します。

比較として、シミュレーテッド・アニーリング(SA)法による最適化手法を説明します。

Agent, Action, Environment, Reward



強化学習(RL)とは、環境と相互作用を繰り返しながら最適な行動を学習する機械学習の手法のことです。数理最適化とは異なるアプローチであり、**強化学習**では、**報酬やペナルティといったフィードバック**を受け取りながら、**試行錯誤を繰り返しながら最適な行動方策を見つけるアプローチ**です。

言葉の意味

・環境 :

エージェントが行動する場所.

・エージェント :

環境内を行動する主体.

・状態 $s \in S$:

環境があるタイムステップにおいてどうなっているか.

・行動 $a \in A$:

エージェントがある状態に対して起こす行動.

・報酬 $r \in R$:

ある状態においてエージェントがある行動を行った結果得られる利益. 強化学習はこの報酬を最大化することを目的とする.

・価値 :

報酬に繋がる状態や行動のことで、以下の行動価値と状態価値がある.

- **Q値(行動価値)** $Q\pi(s,a)$:

ある状態 s においてある行動 a をとった時の価値

- **状態価値** $V\pi(s)$:

取った行動とは無関係に決まる、ある状態に達したときの価値.

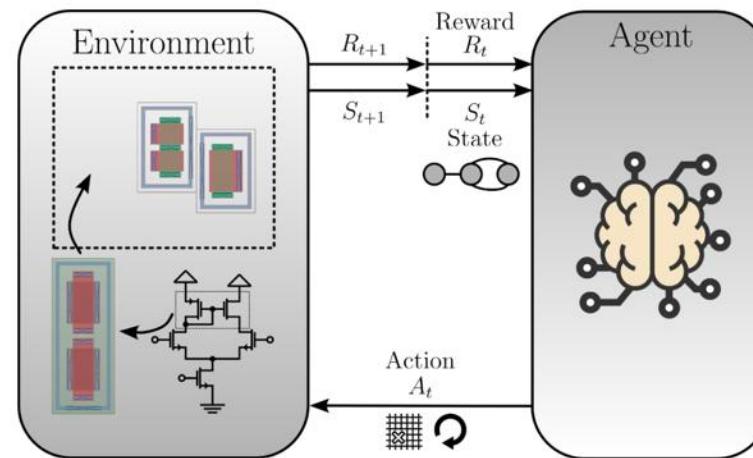
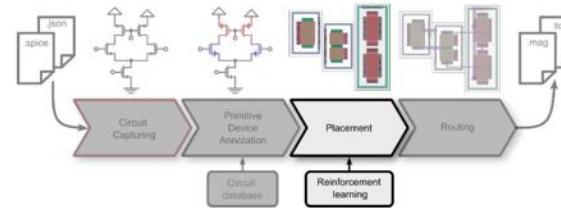
・方策 π :

ある状態が s になると確率的にある行動 a をとる、と決めた関数.

Action, Environment

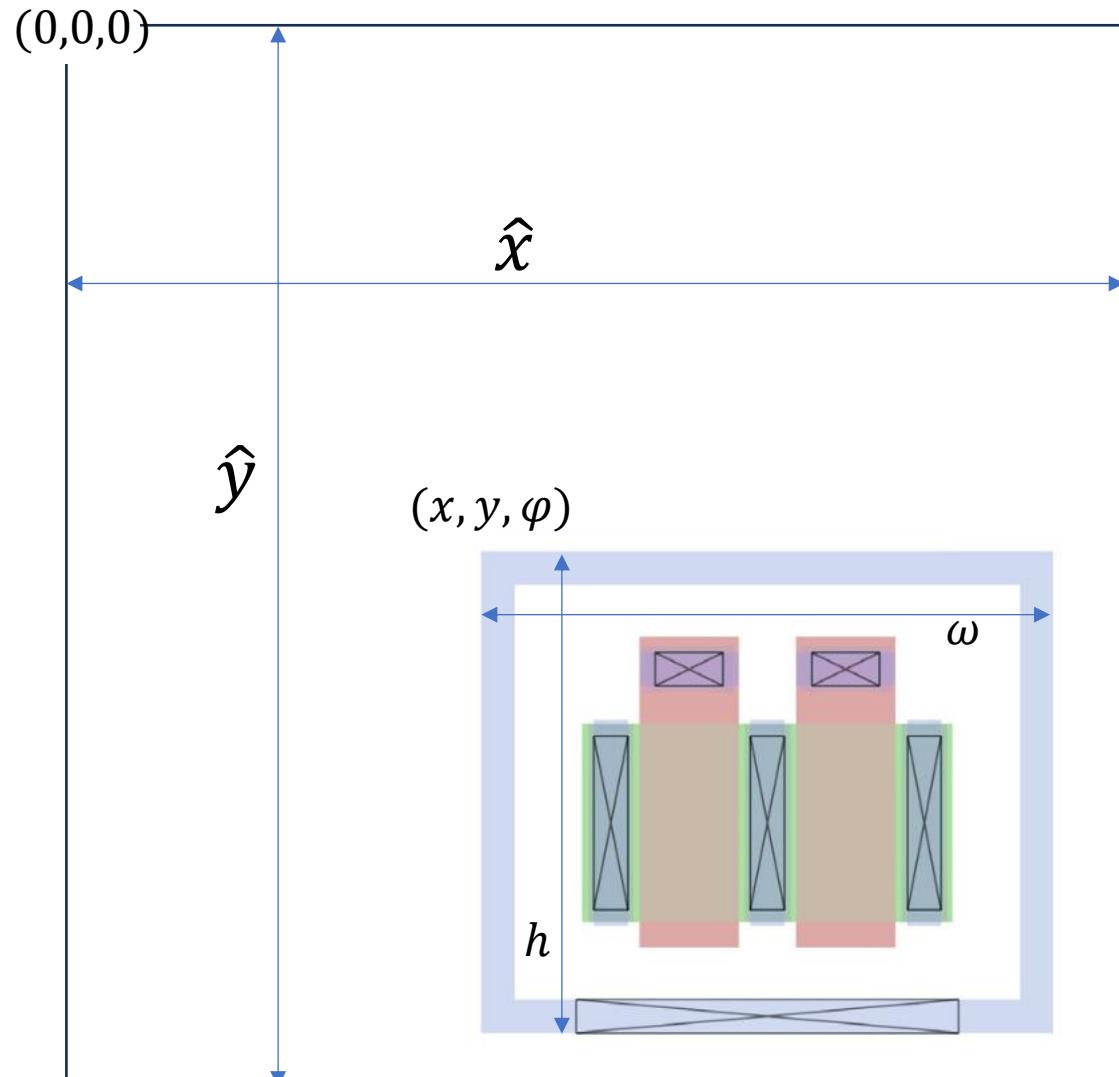
Placement

- Cell views generated in Magic
- Bottom-up approach
- Reinforcement learning based
 - Sequential placement on the die
 - Action
 - x - and y - coordinate
 - Rotation θ
 - State
 - Circuit graph
 - Node features
 - Edge features
 - Final reward
 - Estimated wire length and routing congestion



Action, Environment

環境(Environment)はチップ上のエリアとして定義され、その上にセルが次々と配置(Action)されます。

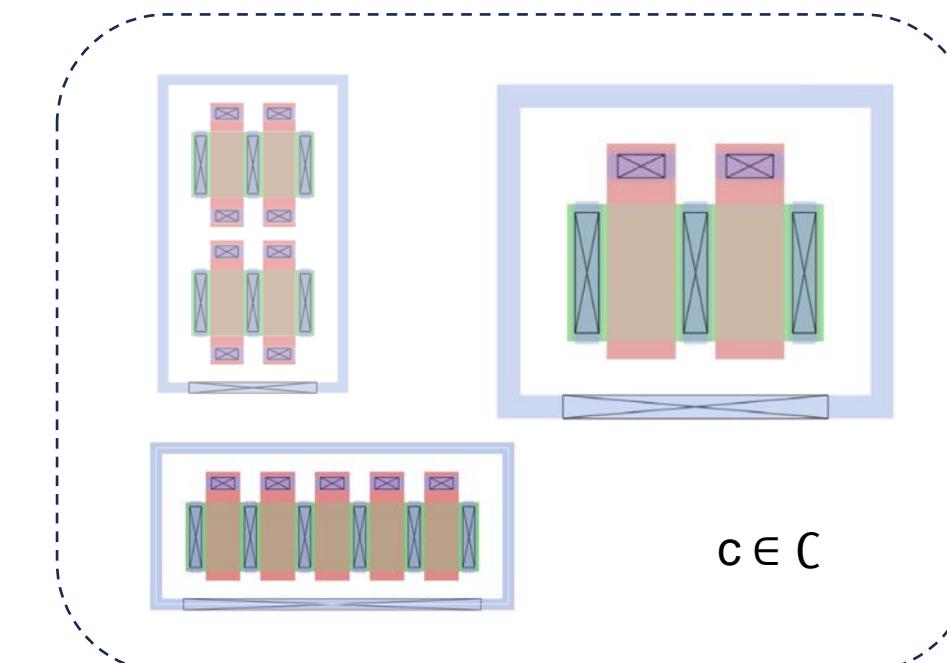


$$\hat{x} = \hat{y} = \left\lfloor \sum_{c \in \mathcal{C}} \max(h_c, w_c) \right\rfloor \quad (5.1)$$

$$\mathcal{P} = \{(x, y) \mid \forall x \in \{0, 1, \dots, \hat{x}\}, \forall y \in \{0, 1, \dots, \hat{y}\}\} \quad (5.2)$$

$$\Phi = \{0, 90, 180, 270\} \quad (5.3)$$

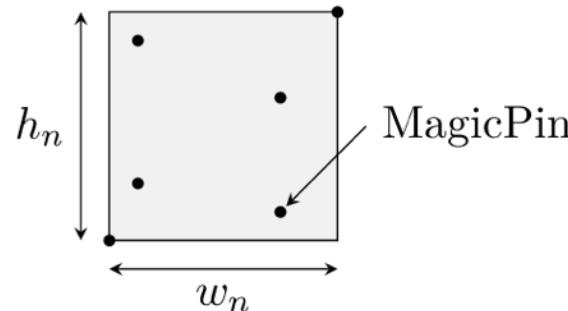
$$\mathcal{A} = \mathcal{P} \cup \Phi = \{(x, y, \varphi) \mid (x, y) \in \mathcal{P}, \varphi \in \Phi\} \quad (5.4)$$



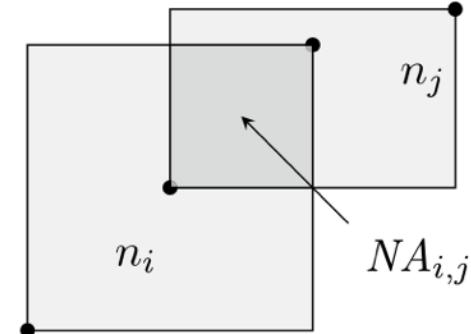
$c \in \mathcal{C}$

State, Reward

最終報酬RT (式(5.8)) は、全体の半周長配線長(HPWL) と配線混雑の平方根の和として定義されます。この式は、全体のネット長を最小化しながら、十分な配線可能性を提供するという考えに基づいています。



(a) Width w_n and height h_n of a net n .
最外周のMagicPin(接続Pin)で
 w, h を定義



(b) Net-overlap-area $NA_{i,j}$, between
the nets n_i and n_j .

Vt : セルのパラメータのベクトル
Et : セル間の接続情報

状態 S_t は時刻 t におけるグラフ $G_t = (V_t, E_t)$ として定義されます。まず時刻 $t = 0$ の初期状態があります。これは、この状態ではすべてのセルが点 $(0, 0)^T$ に位置し、回転角 $\Phi = 0$ 、かつ配置済フラグがリセットされた状態です。もう一つの特別な状態は時刻 $t = T$ で、ここで $T = |C|$ となり、すべてのセルが配置され完了したことを示しています。

$$\text{HPWL} = \sum_{n \in \mathcal{N}} \text{HPWL}(n) = \sum_{n \in \mathcal{N}} (h_n + w_n) \quad (5.5)$$

$$d_n = \frac{(w_n + h_n) p}{w_n h_n} \quad (5.6)$$

$$\text{congestion} = \sum_{n_i \in \mathcal{N}} d_{n_i} \sum_{\substack{n_j \in \mathcal{N} \\ n_i \neq n_j}} NA_{i,j} \quad (5.7)$$

$$R_T = - \left(\text{HPWL} + \sqrt{\text{congestion}} \right) \quad (5.8)$$

$\bar{p} = p/l$ のミス

proximal policy optimization :PPO

エージェントは、各タイムステップ t において、現在の状態 S_t に基づき、1つの配置エピソード中に報酬を最大化するアクション A_t を選択しようと試みます。このタスクを解決するために、Schulmanによって開発された近接方策最適化(proximal policy optimization :PPO)アルゴリズムが実装されました。これは、rollout(), evaluate(), learn() の3つの主要なメソッドを定義しており、配置環境で動作するように適応されています。

近接方策最適化 (proximal policy optimization :PPO) は強化学習のアルゴリズムの一種である。2017年にジョン・シュルマンによって発明され[\[1\]](#)、OpenAIのデフォルトの強化学習アルゴリズムとなった

PPOは **Actor-Critic**ベースの代表的アルゴリズムで、ポリシーを「少しずつ」更新しながら学習の安定性を高めています。

なぜ分けるのか？

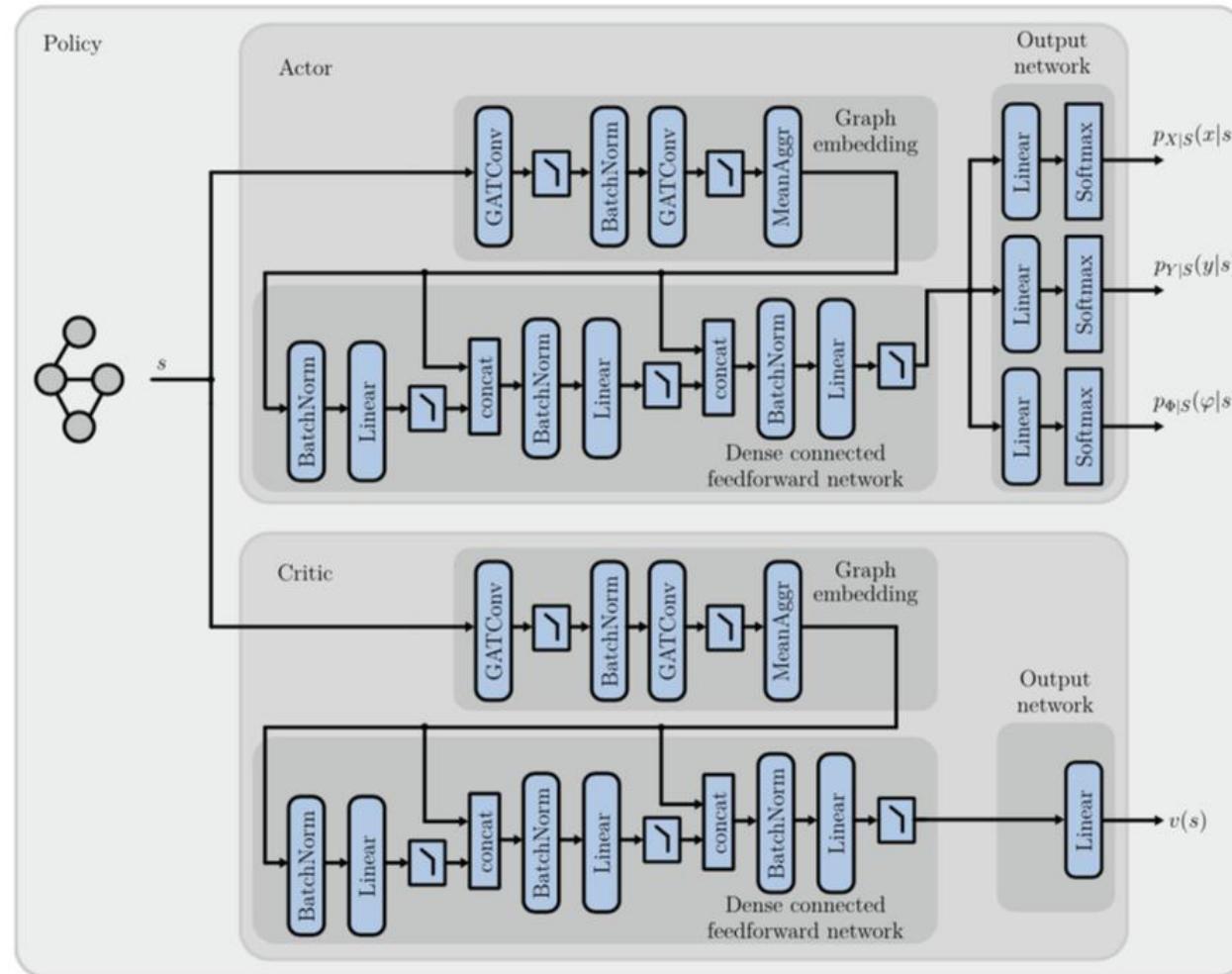
- Policy Gradient (Actorだけ) だと、勾配にはらつきが多くて不安定
- Value-based (Criticだけ) だと、行動選択に制限あり (e.g. ϵ -greedy)

Actor-Criticはその両者の良いとこ取りを目指して、

- **Criticで安定化（価値評価）**
- **Actorで柔軟な行動選択**

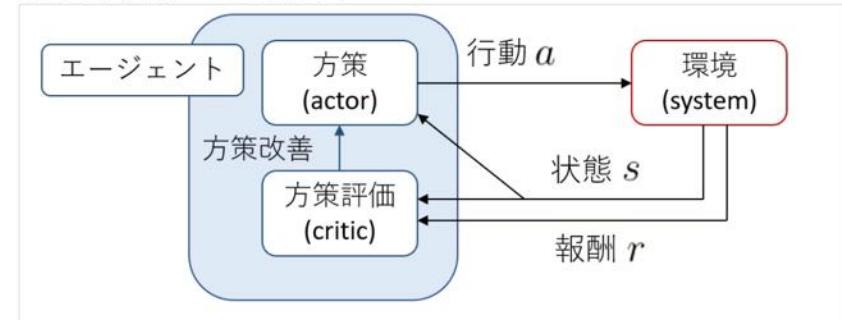
ができる仕組みです。

Policy network



- Actor-Criticとは？(再掲)

価値推定において、方策を選択する際のモジュール(actor)と方策を評価する際のモジュール(critic)を別々になっている。これにより、推定価値を過大評価しないようになったことがメリット、方策ベースで方策オン。



<https://qiita.com/aiueola/items/d195532251f482571b1d>

強化学習における Actor-Critic (アクター・クリティック) の学習の流れ (ざっくり)

1. **Actor**が状態 s に対して行動 a を選ぶ
2. 環境から報酬 r と次の状態 s' を受け取る
3. **Critic**が状態 s の価値を評価 → TD誤差を計算
4. **Critic**はTD誤差が小さくなるように更新
5. **Actor**は「良い行動 (TD誤差がプラス)」をより選びやすくなるようにポリシーを更新

図 5.3: Actor-critic 型 ポリシーネットワーク

Action (rolleout, evaluate, learn)

rollout()

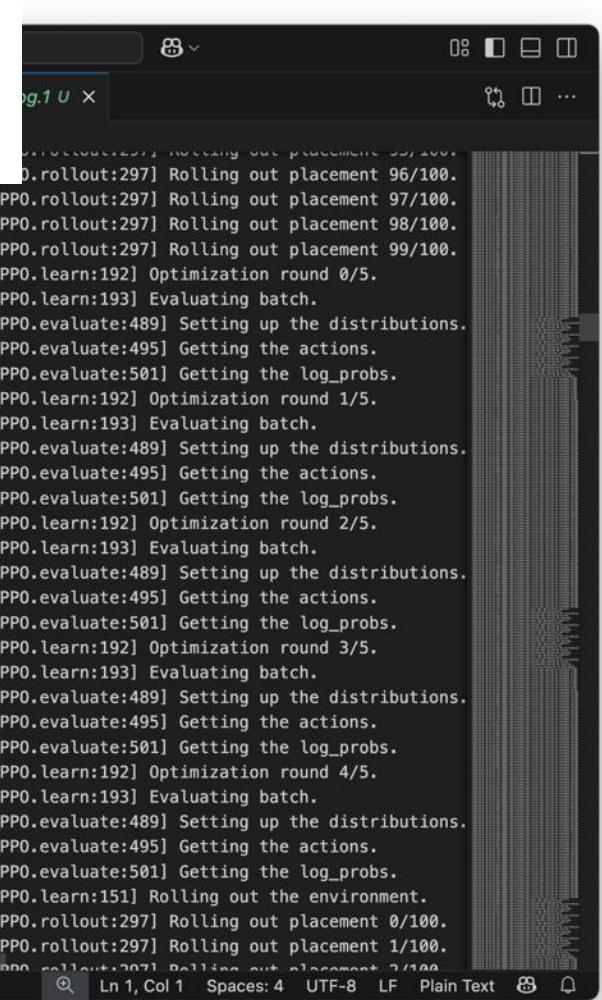
ポリシーに従い n_p 回の配置を試すこと = rollout

evaluate(S, A)

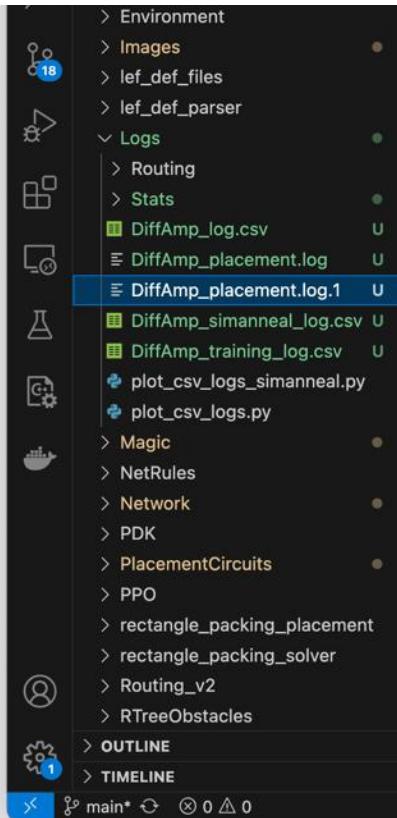
状態Sをポリシーに入力することでポリシーを評価する。

learn()

PPOアルゴリズムに従ってポリシーを更新して配置タスクの学習を行う。



```
0. rollout:297] Rolling out placement 0/100.
0. rollout:297] Rolling out placement 96/100.
0. rollout:297] Rolling out placement 97/100.
0. rollout:297] Rolling out placement 98/100.
0. rollout:297] Rolling out placement 99/100.
14-01 06:34:32,312 DEBUG [PPO.Placement_PPO.learn:192] Optimization round 0/5.
14-01 06:34:32,312 DEBUG [PPO.Placement_PPO.learn:193] Evaluating batch.
14-01 06:34:32,331 DEBUG [PPO.Placement_PPO.evaluate:489] Setting up the distributions.
14-01 06:34:32,336 DEBUG [PPO.Placement_PPO.evaluate:495] Getting the actions.
14-01 06:34:32,336 DEBUG [PPO.Placement_PPO.evaluate:501] Getting the log_probs.
14-01 06:34:32,371 DEBUG [PPO.Placement_PPO.learn:192] Optimization round 1/5.
14-01 06:34:32,371 DEBUG [PPO.Placement_PPO.learn:193] Evaluating batch.
14-01 06:34:32,386 DEBUG [PPO.Placement_PPO.evaluate:489] Setting up the distributions.
14-01 06:34:32,388 DEBUG [PPO.Placement_PPO.evaluate:495] Getting the actions.
14-01 06:34:32,388 DEBUG [PPO.Placement_PPO.evaluate:501] Getting the log_probs.
14-01 06:34:32,420 DEBUG [PPO.Placement_PPO.learn:192] Optimization round 2/5.
14-01 06:34:32,420 DEBUG [PPO.Placement_PPO.learn:193] Evaluating batch.
14-01 06:34:32,434 DEBUG [PPO.Placement_PPO.evaluate:489] Setting up the distributions.
14-01 06:34:32,435 DEBUG [PPO.Placement_PPO.evaluate:495] Getting the actions.
14-01 06:34:32,435 DEBUG [PPO.Placement_PPO.evaluate:501] Getting the log_probs.
14-01 06:34:32,454 DEBUG [PPO.Placement_PPO.learn:192] Optimization round 3/5.
14-01 06:34:32,454 DEBUG [PPO.Placement_PPO.learn:193] Evaluating batch.
14-01 06:34:32,468 DEBUG [PPO.Placement_PPO.evaluate:489] Setting up the distributions.
14-01 06:34:32,469 DEBUG [PPO.Placement_PPO.evaluate:495] Getting the actions.
14-01 06:34:32,470 DEBUG [PPO.Placement_PPO.evaluate:501] Getting the log_probs.
14-01 06:34:32,485 DEBUG [PPO.Placement_PPO.learn:192] Optimization round 4/5.
14-01 06:34:32,485 DEBUG [PPO.Placement_PPO.learn:193] Evaluating batch.
14-01 06:34:32,500 DEBUG [PPO.Placement_PPO.evaluate:489] Setting up the distributions.
14-01 06:34:32,502 DEBUG [PPO.Placement_PPO.evaluate:495] Getting the actions.
14-01 06:34:32,502 DEBUG [PPO.Placement_PPO.evaluate:501] Getting the log_probs.
14-01 06:34:32,517 DEBUG [PPO.Placement_PPO.learn:151] Rolling out the environment.
14-01 06:34:32,518 DEBUG [PPO.Placement_PPO.rollout:297] Rolling out placement 0/100.
14-01 06:34:32,535 DEBUG [PPO.Placement_PPO.rollout:297] Rolling out placement 1/100.
```



Only for ISHI会

Program の流れ

[SchematicCapture.utils.setup_circuit]

Setting up circuit DiffAmp from netlist-file Circuits/Examples/DiffAmp.spice.

Excluding nets [], and using rules NetRules/net_rules_DiffAmp.json.

Parsing netlist.

Building netlist.

Building circuit.

Including net-rules.

Top-Circuit: Circuit(name=DiffAmp) created.

Successfully created circuit Circuit(name=DiffAmp).

[Magic.utils.instantiate_circuit:]

Instantiating Circuit(name=DiffAmp) in magic.

Instantiation topology: [(1, Circuit(name=DiffAmp))]

Instantiating devices of Circuit(name=DiffAmp) in magic. Devices-path: Magic/Devices

Instantiated devices of Circuit(name=DiffAmp) at topological layer 1.

[Magic.utils.generate_cell:]

Generated cell Cell(name=XM5, device=None).

Generated cell Cell(name=XDP_XM1_XM2, device=None).

Generated cell Cell(name=XDL_XM3_XM4, device=None).

[Environment.utils.do_bottom_up_placement:]

Performing bottom-up placement on circuit DiffAmp for 10 placements per circuit/subcircuit.

Circuit: type <class 'SchematicCapture.Circuit.Circuit'>, id 281472027679856

Placement order: [('DiffAmp', 281472027679856)]

Program の流れ

[Environment.utils.do_placement:]

Doing placement DiffAmp of circuit DiffAmp for 2.0 rollouts.

Circuit: type <class 'SchematicCapture.Circuit.Circuit'>, id 281472027679856

Set up placement environment with size (1558, 1558).

Environment: type <class 'Environment.Environment.Placement'>, id 281472026616240

[PPO.Placement_PPO.learn:]

Learning to place for N placements.

Doing NNN placements per batch.

Rolling out the environment.

[PPO.Placement_PPO.rollout:]

Rolling out placement 0/NNN.

...

Rolling out placement NNN/NNN.

[PPO.Placement_PPO.learn:]

Optimization round 0/5

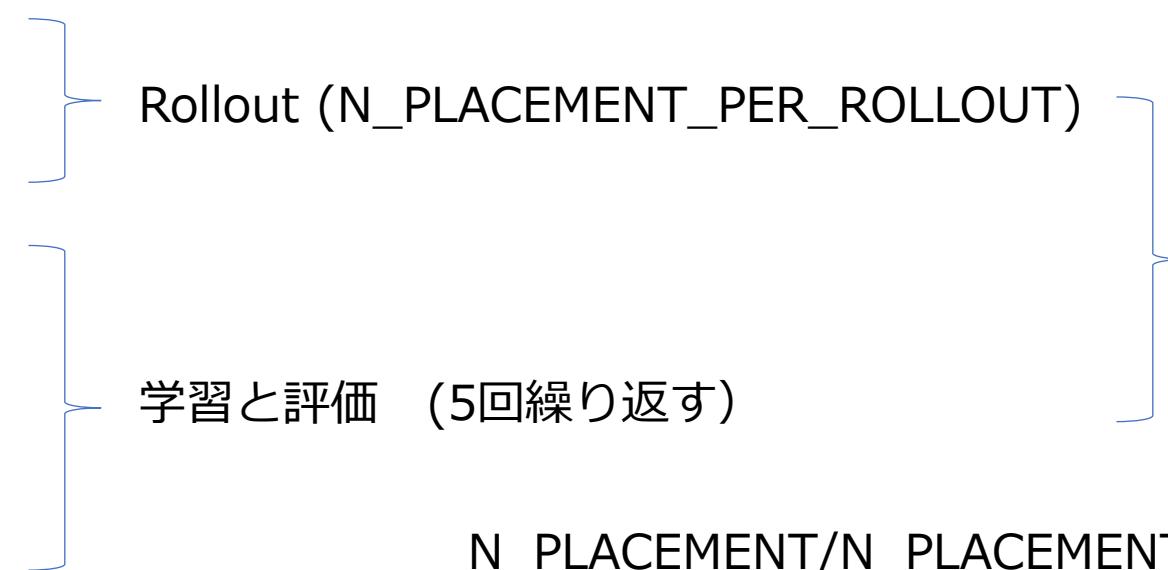
Evaluating batch.

[PPO.Placement_PPO.evaluate:]

Setting up the distributions.

Getting the actions.

Getting the log_probs.



Program の流れ

[Environment.utils.do_placement:]

Placement of DiffAmp done with reward -2631.9847663518703.

Circuit of best placement: type <class 'SchematicCapture.Circuit.Circuit'>, id 281472027120640

[Environment.utils.do_bottom_up_placement:]

Best-circuit: type <class 'SchematicCapture.Circuit.Circuit'>, id 281472027120640

Updating cell of device XM5.

Best-location (np.float64(708.0), np.float64(835.0)), Best-rotation 270.

Updated cell to location (np.float64(708.0), np.float64(835.0)) and rotation 270.

Updating cell of device XDP_XM1_XM2.

Best-location (np.float64(429.0), np.float64(820.5)), Best-rotation 270.

Updated cell to location (np.float64(429.0), np.float64(820.5)) and rotation 270.

Updating cell of device XDL_XM3_XM4.

Best-location (np.float64(429.0), np.float64(1181.0)), Best-rotation 270.

Updated cell to location (np.float64(429.0), np.float64(1181.0)) and rotation 270.

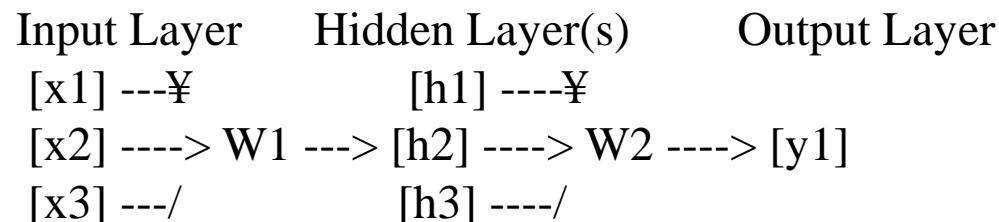
Policy network

Graph Embedding は、ノード（点）、エッジ（辺）、構造情報などを持つ グラフ構造データ を、機械学習で扱いやすいように、ベクトル空間に写像（埋め込み）する技術です。

- グラフ内のノードの類似性や構造情報を保ったまま、低次元の連続空間に変換する。
- 強化学習やその他の機械学習モデルで直接使えるようにする（たとえばQ関数やポリシーに使う状態表現として）。

全結合フィードフォワードネットワークは、ニューラルネットワークの最も基本的な構造の一つです。

- 全結合（Dense / Fully Connected）：各ニューロン（ノード）が前の層のすべてのニューロンとつながっていることを意味します。

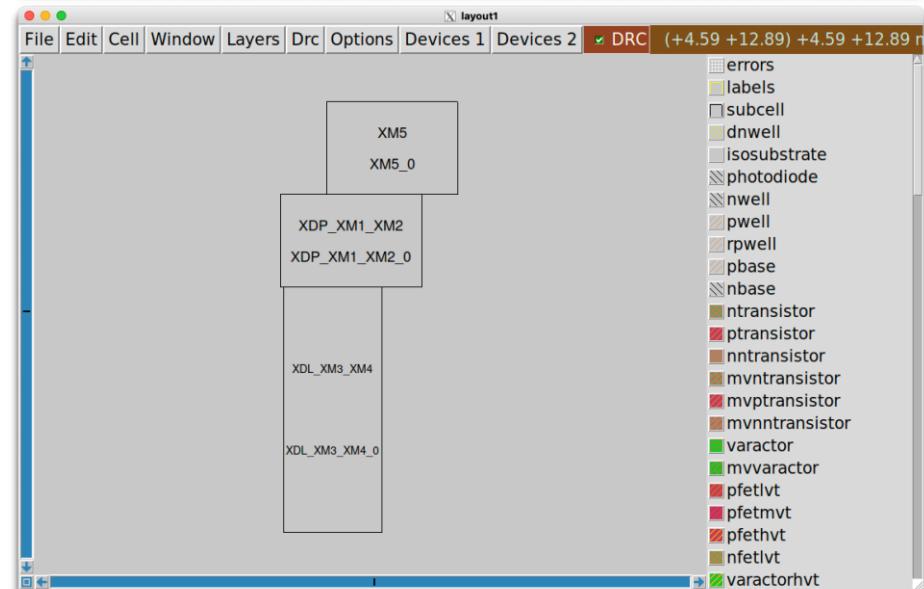
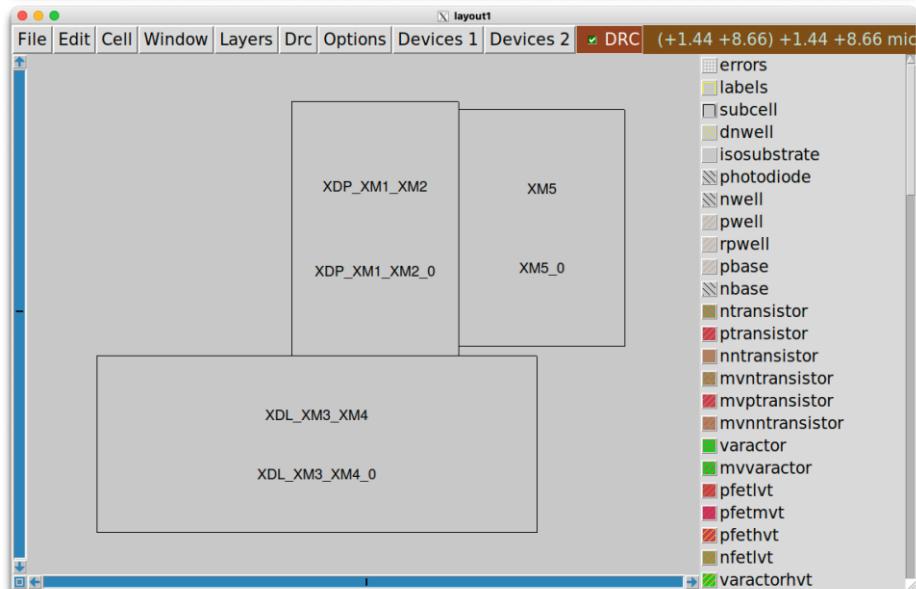
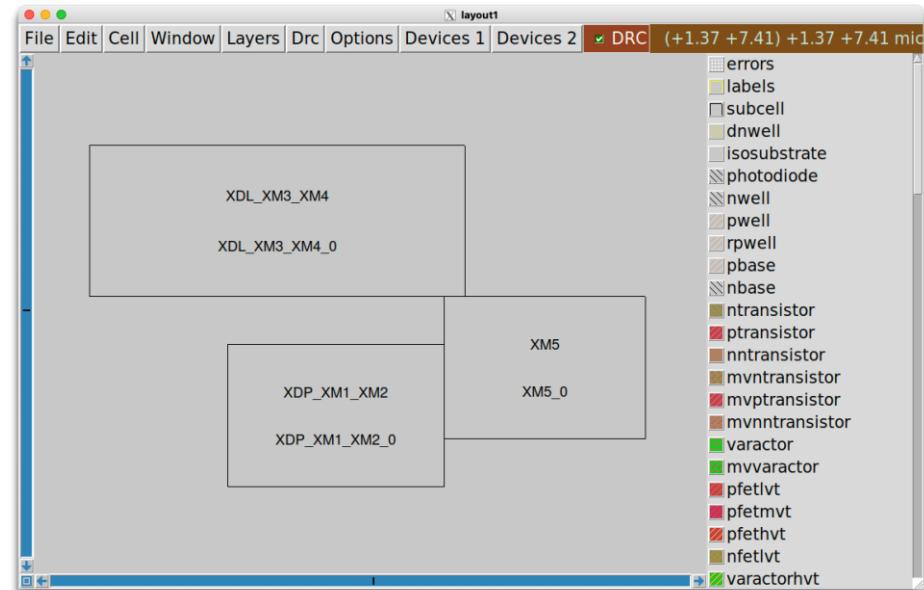
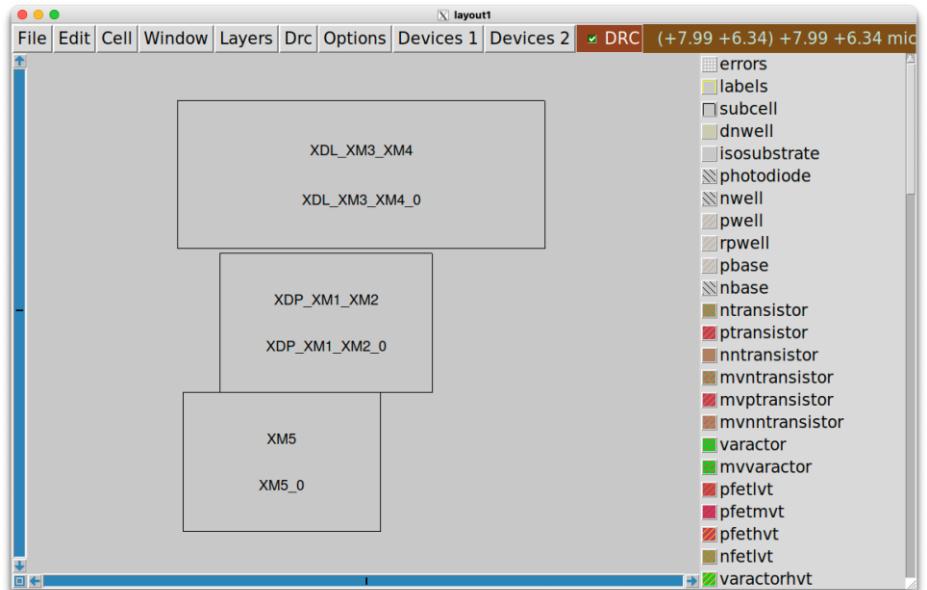


- フィードフォワード（Feedforward）：情報が一方向（入力→出力）にのみ流れる構造です。つまり、ループや再帰はなし。

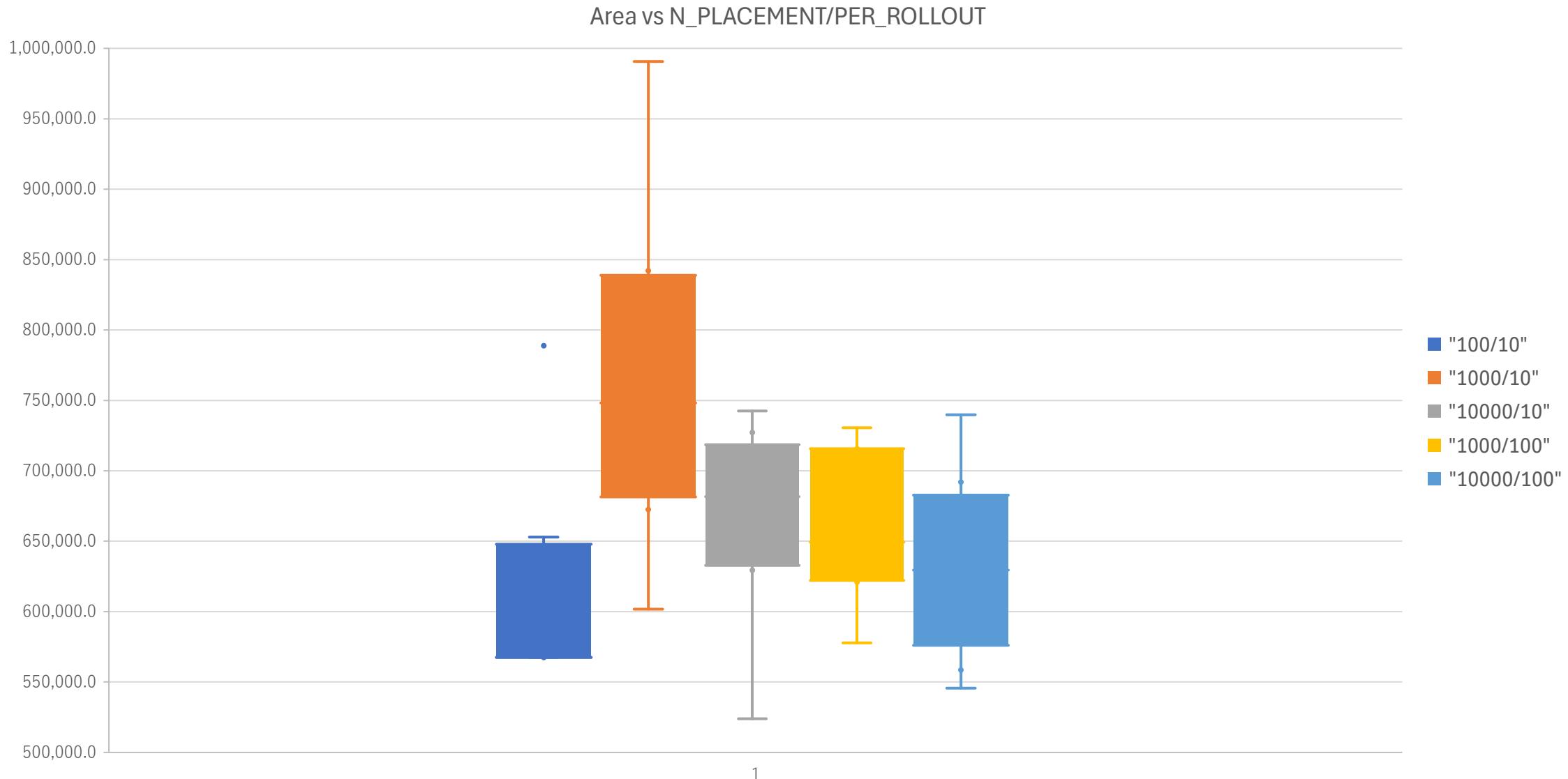
出力ネットワーク（Output Network） とは、ニューラルネットワークにおいて最終的な出力を生成する部分（層）のことを指します。

- ネットワーク全体の構造の中で、
 入力層 → 隠れ層（中間層） → 出力ネットワーク
 という位置づけで、最終的な「予測」や「行動の選択」など、ネットワークの目的に応じたアウトプットを提供します。

最終結果のバラツキの様子(1000/100)



最終結果(Area)のバラツキの様子



Simulated Annealing

焼きなまし法（英: Simulated Annealing SAと略記）は、大域的最適化問題への汎用の乱択アルゴリズムである。SAアルゴリズムは、解を繰り返し求め直すにあたって、現在の解のランダムな近傍の解を求めるのだが、その際に与えられた関数の値とグローバルなパラメータ T （温度を意味する）が影響する。そして、物理過程との相似によって、 T （温度）の値は徐々に小さくなっていく。By Wikipedia

配置タスクを矩形パッキング問題(RPP) に変換し、シミュレーテッドアニーリング(SA) 最適化を適用することで解決します。RPP の解決には、Terada によって開発された矩形パッキングソルバー(RPS) ライブラリが使用されました。ここでは、パッキングすべき矩形のリストによって、Problem クラスとして表現されるRPPが設定され、各矩形はその幅と高さで指定されます。オプションとして、これらの矩形は回転可能(rotable) として設定でき、パッキング中に幅と高さを入れ替えることが可能です。パッキング内の矩形を識別するために、各矩形には固有の識別番号が割り当てられています。パッキングの状態は、Murata (22) によって定義されたシーケンスペアで表現され、これに矩形の回転情報が拡張されています。したがって、各状態 $s \in S$ は、三つ組 $(\Gamma+, \Gamma-, \Theta)$ によって記述されます。

Program の流れ

[SchematicCapture.utils.setup_circuit]

Setting up circuit DiffAmp from netlist-file Circuits/Examples/DiffAmp.spice.

Excluding nets [], and using rules NetRules/net_rules_DiffAmp.json.

Parsing netlist.

Building netlist.

Building circuit.

Including net-rules.

Top-Circuit: Circuit(name=DiffAmp) created.

Successfully created circuit Circuit(name=DiffAmp).

[Magic.utils.instantiate_circuit:]

Instantiating Circuit(name=DiffAmp) in magic.

Instantiation topology: [(1, Circuit(name=DiffAmp))]

Instantiating devices of Circuit(name=DiffAmp) in magic. Devices-path: Magic/Devices

Instantiated devices of Circuit(name=DiffAmp) at topological layer 1.

[Magic.utils.generate_cell:]

Generated cell Cell(name=XM5, device=None).

Generated cell Cell(name=XDP_XM1_XM2, device=None).

Generated cell Cell(name=XDL_XM3_XM4, device=None).

[rectangle_packing_placement.utils.do_bottom_up_placement:147]

Circuit: type <class 'SchematicCapture.Circuit.Circuit'>, id 281472823294960

Placement order: [('DiffAmp', 281472823294960)]

[rectangle_packing_placement.utils.do_placement:66]

PlacementProblem: type <class 'SchematicCapture.Circuit.Circuit'>, id 281472823294960

PlacementSolver: circuit name DiffAmp

Program の流れ

```
[rectangle_packing_placement.placement_solver._solve_with_strategy:160]
    PlacementRPP Hard
[rectangle_packing_placement.rectangle_packing_solver.solver.update:228]
    RPP Anneal: update
[rectangle_packing_placement.rectangle_packing_solver.solver.move:269]
    RPP Hard:move
    ...
    RPP Hard:move
[rectangle_packing_placement.rectangle_packing_solver.solver.update:228]
    RPP Anneal: update
[rectangle_packing_placement.rectangle_packing_solver.solver.move:269]
    RPP Hard:move
    ...
    RPP Hard:move
[rectangle_packing_placement.rectangle_packing_solver.solver.update:228]
    RPP Anneal: update

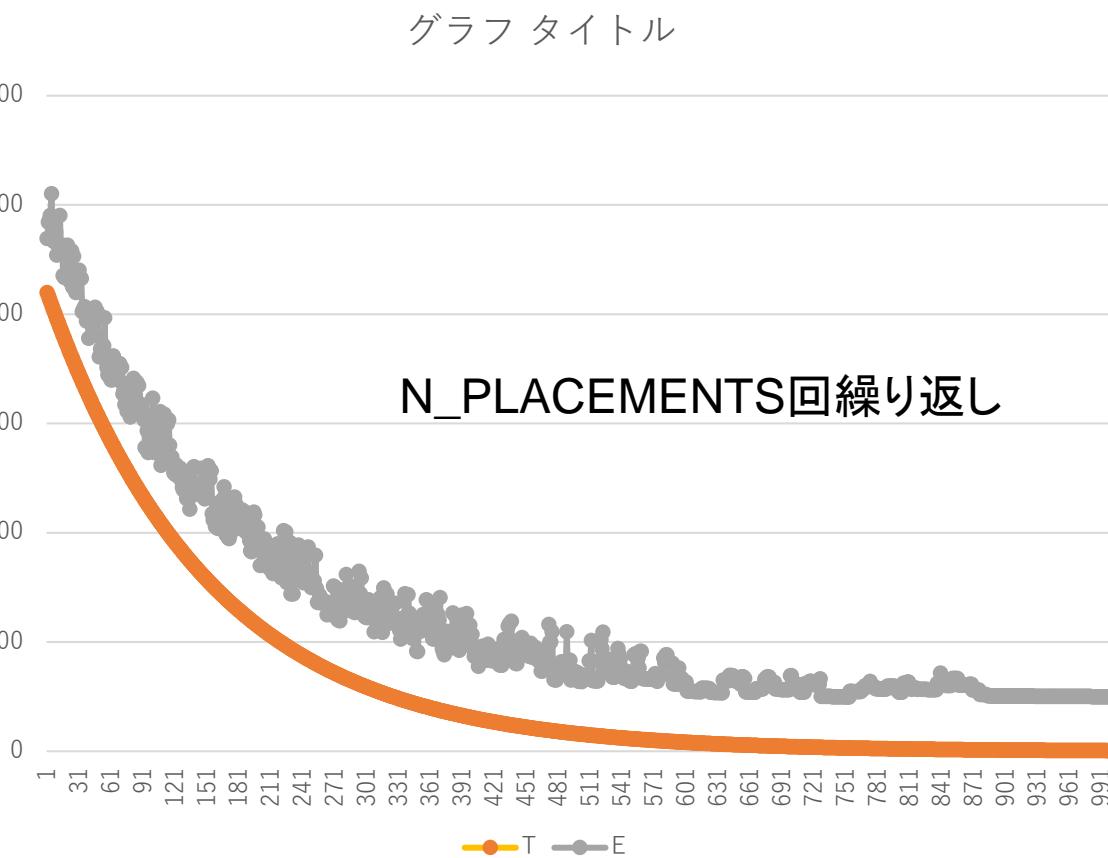
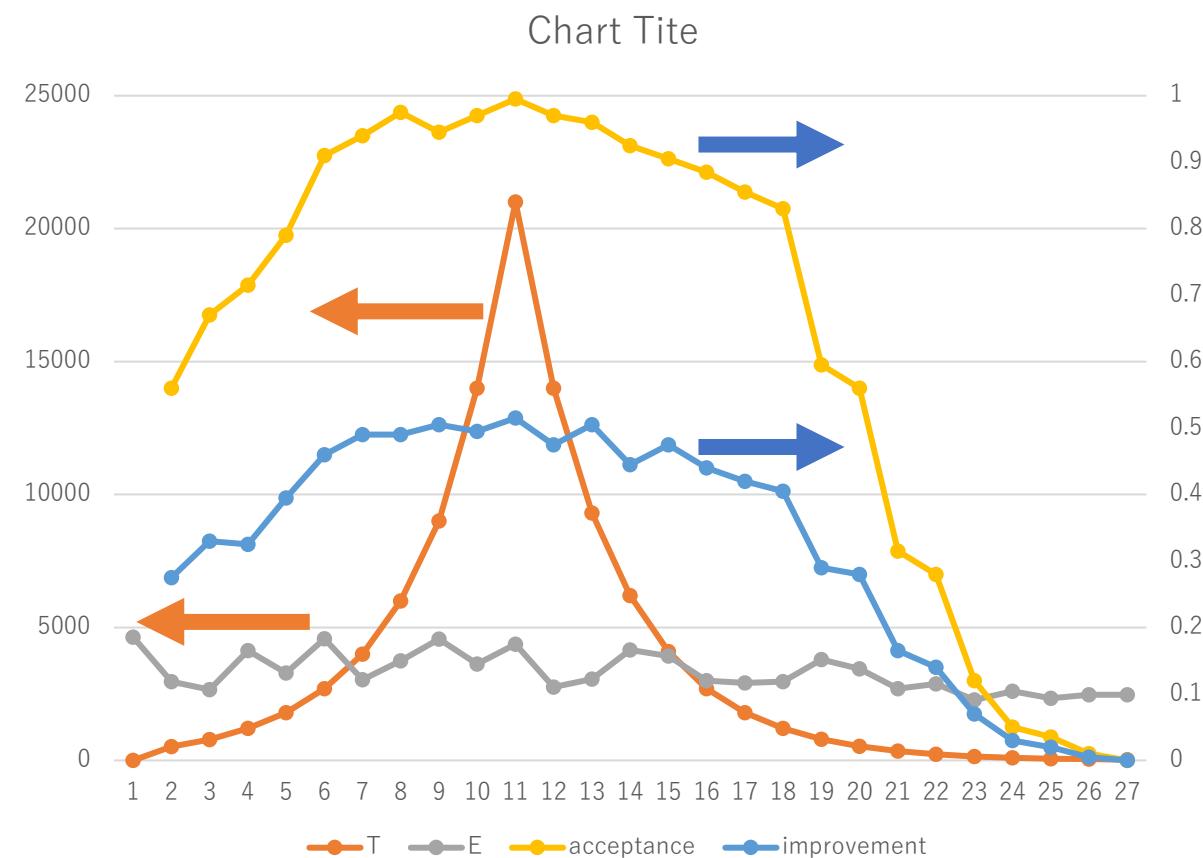
[rectangle_packing_placement.rectangle_packing_solver.solver.anneal:239]
    RPP Anneal: anneal

[rectangle_packing_placement.rectangle_packing_solver.solver.move:269]
    RPP Hard:move
[rectangle_packing_placement.rectangle_packing_solver.solver.move:269]
    RPP Hard:move
```

探索 . . .

N_PLACEMENTS 回繰り返す

Temp/Energy の様子



アニール時間(SIM_ANEL_STEP/N_PLACEMENT)での違い

Name	Value
Circuit	DiffAmp
Time taken [s]	3
Placements	1000
Total HPWL	2169.0
Congestion	10992.22
Total width	721.0
Total height	737.0
Area	531377.0

20/1000

Name	Value
Circuit	DiffAmp
Time taken [s]	18
Placements	1000
Total HPWL	2214.0
Congestion	10814.54
Total width	692.5
Total height	737.0
Area	510372.5

200/1000

Name	Value
Circuit	DiffAmp
Time taken [s]	228
Placements	1000
Total HPWL	2275.0
Congestion	3819.86
Total width	1016.0
Total height	721.0
Area	732536.0

2000/1000

Name	Value
Circuit	DiffAmp
Time taken [s]	14
Placements	100
Total HPWL	2279.0
Congestion	16194.73
Total width	737.0
Total height	721.0
Area	531377.0

200/100

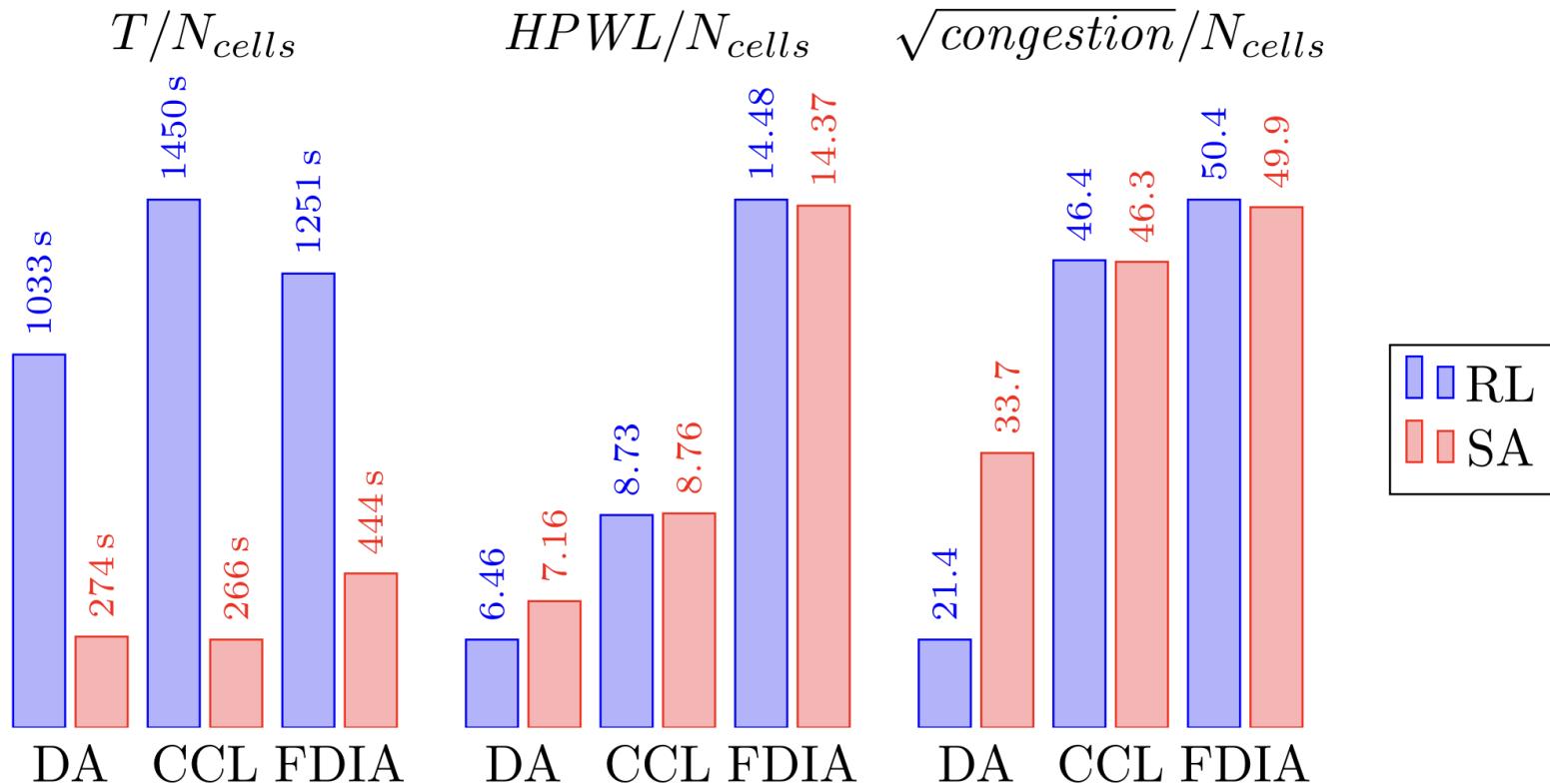
Name	Value
Circuit	DiffAmp
Time taken [s]	23
Placements	1000
Total HPWL	2424.0
Congestion	4628.38
Total width	985.5
Total height	869.0
Area	856399.5

200/1000

Name	Value
Circuit	DiffAmp
Time taken [s]	41
Placements	10000
Total HPWL	2141.0
Congestion	9158.11
Total width	721.0
Total height	810.0
Area	584010.0

200/10000

Experimental results – Placement comparison



Summary & Outlook

- Developed an electronic design automation (EDA) tool
- Open-source automated analog layout generation
- Reinforcement learning based placement
- Two stage routing flow
- Outlook
 - Adding more primitive devices
 - Adding more circuits to the database
 - Symmetrical placement
 - Symmetrical routing