**API documentation for IC placement by using reinforcement learning and simulated annealing method**

1. **Placement problem registration**

GET, POST

/ICPlacement/

**Request body param**

Content-Type:application/json

Accept:application/json

{

"problems":

{

"TopLayer":

{

"Components":

[

{

"width": 30.5,

"height": 40.5,

},

{

"width": 20.5,

"height": 50.5,

},

{

"width": 70.5,

"height": 80.5,

},

],

"PreDefinedBlocks":

[

{

"top": 180,

"left": 150,

"bottom":80,

"right":210,

},

],

"BottomLayer":

{

"Components":

[

{

"width": 40.5,

"height": 50.5,

},

{

"width": 50,

"height": 80,

},

],

"PreDefinedBlocks":

[

{

"top": 180,

"left": 150,

"bottom":80,

"right":210,

},

],

}

"Epoch": 12

"SACount": 5000

"RLSteps": 200

}

}

**Response**

{

"problems\_id": 10

}

1. **Get the optimized placement result with problem id.**

GET, POST

/ICPlacement/

**Request body param**

{

"problems\_id" : 10

}

**Response**

{

"status" : string (this one will say the current calculation status on the server. For example, this message might be like this. “Still running”)

"result" :

[

{

“id” : 0

“top” : 23.5

“left” : 34.7

“right” : 60

“bottom” : 15.0

},

]

}

In order to get the placement, I have created two APIs with the same endpoint, but different parameters.

In the first API call, the placement problem will be registered to the server because it can not reply to the result quickly due to the long computation.

Instead, it will reply to the problem id if this task was successfully registered.

In the client side, it needs to send the request periodically to check if the registered task was computed or not. This is what the second API call is needed.

In the second API call, it will request with the problem id that was taken from the first API call. Once the server calculated the placement, it will return the result and the result will include the positions (top, left, right bottom) of all the components.