NVCell 2: Routability-Driven Standard Cell Layout in Advanced Nodes with Lattice Graph Routability Model

Chia-Tung Ho*
Nvidia Research
Santa Clara, CA, USA
chiatungh@nvidia.com

Alvin Ho
Nvidia
Santa Clara, CA, USA
alho@nvidia.com

Matthew Fojtik
Nvidia Research
Durham, NC, USA
mfojtik@nvidia.com

Minsoo Kim
Nvidia
Austin, TX, USA
minsook@nvidia.com

Shang Wei
Nvidia
Santa Clara, CA, USA
shwei@nvidia.com

Yaguang Li
Nvidia
Austin, TX, USA
yaguangl@nvidia.com

Brucek Khailany
Nvidia Research
Austin, TX, USA
bkhailany@nvidia.com

Haoxing Ren*
Nvidia Research
Austin, TX, USA
haoxingr@nvidia.com

ABSTRACT

Standard cells are essential components of modern digital circuit designs. With process technologies advancing beyond the 5nm node, more routability issues have arisen due to the decreasing number of routing tracks, increasing number and complexity of design rules, and strict patterning rules. Automatic standard cell synthesis tools are struggling to design cells with severe routability issues. In this paper, we propose a routability-driven standard cell synthesis framework using a novel pin density aware congestion metric, lattice graph routability modelling approach, and dynamic external pin allocation methodology to generate routability optimized layouts. On a benchmark of 94 complex and hard-to-route standard cells, NVCell 2 improves the number of routable and LVS/DRC clean cell layouts by 84.0% and 87.2%, respectively. NVCell 2 can generate 98.9% of cells LVS/DRC clean, with 13.9% of the cells having smaller area, compared to an industrial standard cell library with over 1000 standard cells.

CCS CONCEPTS

• **Hardware → Standard cell libraries**; • **Computing methodologies → Machine learning**.

KEYWORDS

Standard cell design automation, Electronic design automation, Machine learning

ACM Reference Format:
Chia-Tung Ho, Alvin Ho, Matthew Fojtik, Minsoo Kim, Shang Wei, Yaguang Li, Brucek Khailany, and Haoxing Ren. 2023. NVCell 2: Routability-Driven Standard Cell Layout in Advanced Nodes with Lattice Graph Routability Model. In *Proceedings of the 2023 International Symposium on Physical Design (ISPD '23), March 26–29, 2023, Virtual Event, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3569052.3578920

2 This work is licensed under a Creative Commons Attribution International 4.0 License.

ISPD '23, March 26–29, 2023, Virtual Event, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9978-4/23/03.
https://doi.org/10.1145/3569052.3578920

1 INTRODUCTION

Standard cells are the essential building blocks of digital Very Large-Scale Integration (VLSI) designs. As process technologies relentlessly advance beyond $5nm$, the decreasing number of routing tracks, increasing number of design rules, and strict patterning rules are leading to severe routability issues in standard cell layouts. In addition, there are thousands of standard cells that need to be designed for each technology node. Experienced human standard cell designers are struggling to deliver standard cell libraries in time, because of the increasing difficulty of complying with all the Design Rule Checks (DRCs) in such a routing limited environment. Therefore, automatic standard cell synthesis with the consideration of routability is of great importance in advanced technology nodes.

Recently, automated standard cell synthesis tools such as NVCell [1] and BonnCell [2], have been shown to generate high quality cell layouts on advanced technology nodes. Due to routability issues, one of the key challenges is that the generated placement for any given cell could be unroutable or unable to be routed without DRC errors. Previous works use simpler heuristics or models to predict routability, but they are still struggling to generate routable device placements when considering complex design rules and multi-patterning rules in the actual routing stage. Moreover, the standard cell external pin assignment is also very critical to the routability because of limited routing resources.

In this paper, we develop a routability-driven standard cell design automation framework: NVCell 2, which enhances the original NVCell with device placement guided by a novel pin density aware (PDA) congestion metric and a lattice graph routability model, and routing with dynamic pin allocation. We demonstrate that NVcell 2 can successfully generate 98.9% LVS/DRC clean cell layouts over 1000 standard cells in an industrial standard cell library.

Our main contributions are as follows.

- We propose a novel Pin Density Aware (PDA) congestion metric to capture the routability of local areas in the standard

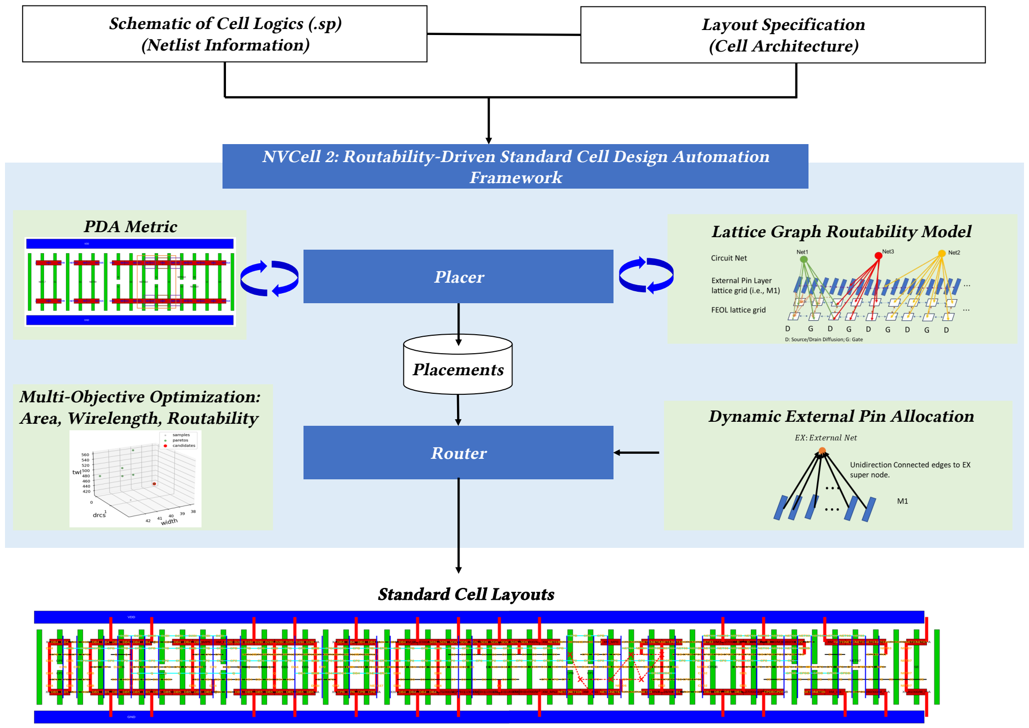

Schematic of Cell Logics (.sp)
(Netlist Information)
Layout Specification
(Cell Architecture)
NVCell 2: Routability-Driven Standard Cell Design Automation Framework
PDA Metric
Placer
Lattice Graph Routability Model
Placements
Multi-Objective Optimization: Area, Wirelength, Routability
Router
Dynamic External Pin Allocation
Standard Cell Layouts

Figure 1: NVCell 2: Routability-Driven Standard Cell Design Automation Framework Overview

cell. The PDA metric achieves correlations of 0.9543 and 0.8364 with the average routing congestion and the area of golden unrouted probability distribution, respectively, on a dataset of 2240 device placements.

- We develop a novel lattice graph routability modelling approach to capture the routability of local areas, routability impacts between local areas, and global net connections in the standard cell. A cell-level routabilty metric is extracted from the lattic graph model, which achieves correlations of 0.9608 and 0.8536 with the average routing congestion and the area of golden unrouted probability distribution, respectively.
- We propose a dynamic standard cell external pin allocation methodology to dynamically assign the external pins with the considerations of routability and design rules in the routing phase.
- We use multi-objective BOHB [3, 4] to explore the weights of cell width and routability to generate competitive cell layouts in terms of cell width, routability, number of DRC errors, and total wirelength.
- On a set of 94 cells with routability issues on the original NVCell, NVCell 2 can produce LVS/DRC clean layouts for 27.7%, 63.8% and 87.2% with PDA metric, lattice graph model and dynamic pin allocation, incrementally.
- NVCell 2 achieves cell layouts with lower area than the existing industrial standard cell library for 13.9% of over 1000 cells.

The remaining sections are organized as follows. Section 2 reviews prior works in standard cell layout automation and gives a brief overview of the original NVCell [1] that this work is built upon. Section 3 describes our routability-driven standard cell design automation framework. Section 4 presents our main experiments. Section 5 concludes the paper.

2 BACKGROUND

Standard cell layout automation includes placement and routing steps. The placement step places devices; the routing step connects device terminals and pins based on net connectivity. Sequential standard cell synthesis approach, such as [5], [2], [6], and [1], performs the placement step first and then the routing step. Placement techniques include heuristic based methods, exhaustive search based methods, and mathematical programming based methods. Routing techniques include channel routing, SAT, and Mixed-Integer Linear Programming(MILP) based routing methods. [5] leveraged MILP algorithms to find optimal device placement. [2] and [6] used branch and bound or dynamic programming techniques to explore optimal transistor placement exhaustively and then formulate the MILP for in-cell routing. Recently, Ren *et al.* [1] used the simulated annealing technique to generate optimal transistor placement, leveraged genetic algorithms for routing, and applied reinforcement learning to fix the design rule violation.

There are also several works proposing to solve the placement and routing problems simultaneously using Satisfiability-modulo-theory (SMT) in [7], [8], [9], and [10]. By encoding the design rules in the engine, simultaneously standard cell synthesis approach can generate routable standard cell layouts. However, their scalability is worse than sequential standard cell synthesis approaches on large and complex standard cell designs (i.e., more than 50 devices).

Cell external pin assignment is also important for cell routability. NVCell [1] assigns external pins in the placement stage. [9] supports a dynamic external pin allocation approach in its multi-commodity flow based MILP or SMT engines. In other works, such as [2], and [6], it is not clear what methodology is used to assign external pins.

2.1 NVCell

The NVCell framework [1] is a sequential standard cell automation approach, which consists of placement and routing stages. In the placement stage, given a set of PMOS and NMOS devices, the goal of placement is to place them on the PMOS row and NMOS row of the cell layout while satisfying technology constraints. Here, the conventional simulated annealing algorithm is selected because its adaptability to custom layout constraints and ease of implementation. The designed simulated annealing based algorithm does both pairing and ordering simultaneously. Simulated annealing makes moves on a placement representation which specifies the placement order of pins, ordering of NMOS and PMOS devices, and whether to flip a device orientation (switching the source and drain positions). It optimizes a scoring function which is a weighted sum of cell width, routability estimation, and estimated wirelength. These moves can be categorized either by the types of moves or by the targeted devices of the moves. The Flip changes all targeted devices flip flag. The Swap swaps targeted devices. The Move moves targeted devices to a specific location. The target devices can be either consecutive PMOS devices, consecutive NMOS devices, or consecutive PMOS/NMOS device pairs. The simulated annealing algorithm is implemented based on the modified Lam annealing schedule [11] that requires no hyperparameter tuning.

NVCell has two routability estimation methods: a heuristic method and a model-based method. The heuristic method estimates routing congestion with a simple heuristic. For each net, it draws a horizontal line from its left most terminal to its right most terminal. Then it compute the max and average number of crossing lines on each x position and use a weight average as the routability estimation for the placement. The model-based method is based on a machine learning model. On each device pair, it collects features such as the number of nets connected to each terminal of the PMOS and NMOS devices, the number of pins near it, and the number of estimated wire crossings over it, etc. It then concatenates the features of all the device pairs together into a tensor and use 1D convolution and max pooling to predict the routability of the given placement. The routability is labelled as $[routable, routablebutwithDRCs, notroutable]$, which are generated by the router. The routability estimation methods help improve the success rate of NVCell, but still fails to produce many cells with severe routing issues.

In the routing phase, there are two steps: a genetic algorithm-based routing step and a Reinforcement Learning (RL)-based DRC fixing step [12]. The genetic algorithm drives a maze router to create many routing candidates, and the DRC RL agent reduces the number of DRCs of a given routing candidate [1]. The genetic algorithm based routing algorithm uses routing segments as the genetic representation in that it ensures that good routing islands in the routing structure are preserved during genetic operations such as crossover and mutation. The fitness of each individual chromosome in a generation is evaluated based on two metrics: the number of unrouted terminal pairs and the number of DRCs.

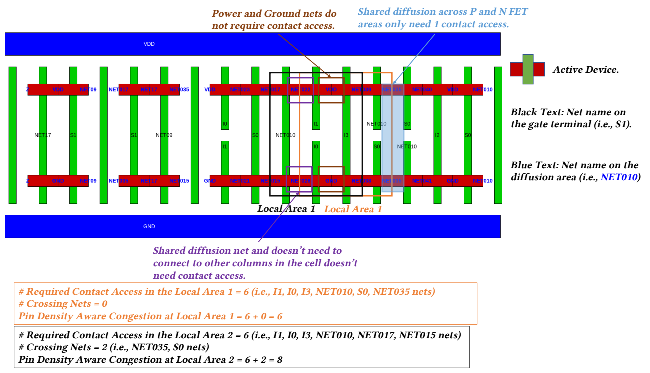
3 ROUTABILITY-DRIVEN FRAMEWORK

We introduce the novel pin density aware congestion (PDA) Metric, lattice graph routability model, dynamic external pin allocation, and multi-objective optimization approach here. Figure 1 shows the overview of our framework. Given a cell netlist and layout specification, our framework firstly uses a simulated annealing based placement engine [1] with PDA metric and lattice graph routability model to generate standard cell placements. Then, the genetic algorithm based router [12] using dynamic external pin allocation are leveraged to generate optimized cell layouts with the consideration of design rules. Moreover, we use multi-objective optimization approach to optimize the cell area, total wirelength, and routability together.

3.1 Pin Density Aware Congestion Metric

We introduce the Pin Density Aware Congestion (PDA) Metric for routability-aware device placement generation. Given a device placement information, the PDA metric is used to capture the number of required contacts to lowest metal layer (i.e., M0) in a local area and number of crossing nets, which exclude the nets within the local area.

In Figure 2, we count the number of required contacts for Local Area 1 and Local Area 2 based on the technology definition. For shared diffusion across the P and N FET area, there is only one contact needed for this technology. For power, ground, and shared diffusion nets that does not need to be connected to other transistors, the number of required contacts is 0. As a result, the number of required contacts in the local window 1 is 6, which include the contacts to access I1, I0, I3, NET010, S0, and NET035 nets. Since the crossing nets from the left side to the right side of the local area 1 are also the nets inside the local area 1, the number of crossing nets are 0. For the local area 2, The number of required contacts is 6 (i.e., I1, I0, I3, NET010, NET017, NET015 nets). Here, Net035 and S0 nets are not nets inside the local area 2 and they need to be connected from the left side to the right side of the local area 2. As a result, the number of crossing nets is 2 in this example. The Pin Density Aware Congestion at Local Area 2 is 8.


Power and Ground nets do not require contact access.
Shared diffusion across P and N FET areas only need 1 contact access.
VDD
Active Device.
Black Text: Net name on the gate terminal (i.e., S1).
Blue Text: Net name on the diffusion area (i.e., NET010)
Local Area 1
Local Area 1
GND
Shared diffusion net and doesn't need to connect to other columns in the cell doesn't need contact access.

# Required Contact Access in the Local Area 1 = 6 (i.e., I1, I0, I3, NET010, S0, NET035 nets)
# Crossing Nets = 0
Pin Density Aware Congestion at Local Area 1 = 6 + 0 = 6

# Required Contact Access in the Local Area 2 = 6 (i.e., I1, I0, I3, NET010, NET017, NET015 nets)
# Crossing Nets = 2 (i.e., NET035, S0 nets)
Pin Density Aware Congestion at Local Area 2 = 6 + 2 = 8

Figure 2: Pin Density Aware Congestion Example 1 with Comb14 design. There are 26 devices in the cell design.

In addition, the number of required contact access and crossing nets in the local area can be calculated with weights based on the net name or the transistor pin structure (i.e., split gate/diffusion). For example, we can make the weights of net I1 and I0 larger

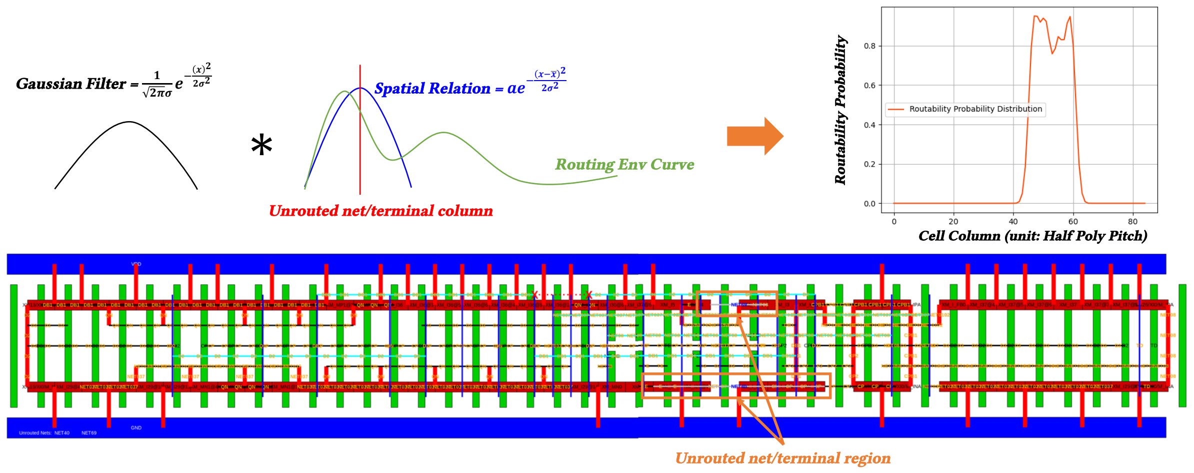

Gaussian Filter = $\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x)^2}{2\sigma^2}}$
*
Spatial Relation = $ae^{-\frac{(x-\bar{x})^2}{2\sigma^2}}$
Routing Env Curve
Unrouted net/terminal column
Routability Probability
Routability Probability Distribution
Cell Column (unit: Half Poly Pitch)
Unrouted net/terminal region

Figure 3: An example of generated routability probability distribution of a latch design for clock trees. There are 76 devices in this cell design.

than 1 in the Local Area 2 to penalize the split gate structure as shown in Figure 2. For the window size, We can adjust the width according to the contact rules and end-of-line spacing rules of the lowest metal layer (i.e., M0). Most of the routability problems happen in the region near large Pin Density Aware Congestion local areas in a standard cell placement. As a result, we can smooth the Pin Density Aware Congestion of the local areas to resolve the routability problems during transistor placement.

To obtain the cell-level Pin Density Aware Congestion Metric (i.e., P_{cell}), we can apply operations to PDA metric of each local area in the standard cell placement. The operation can be maximum, average, or top-k, etc. For example, we show the maximum, and average operations to obtain the P_{cell} in Equation (1) below.

$$\textit{Max: } P_{cell} = \max_{i=1...N_w} PDA_i / \textit{Average: } P_{cell} = \frac{\sum_i^{N_w} PDA_i}{N_w} \tag{1}$$

Where N_w is the number of local windows of PDA metric in the cell placement.

3.2 Lattice Graph Routability Model

The lattice graph routability model is used to capture the routability of local areas, interaction between local areas, and global net connections in the standard cell. Firstly, we introduce the routability probability of each column in the cell to provide a more global view of routability than the PDA metric, which considers the pin accessibility and congestion in a local window. Secondly, we show the architecture of lattice graph routability model and the loss function. Lastly, we introduce the calculation of predicted cell-level routability metric, R_{cell}, for optimizing the routability of device placement.

3.2.1 Routability Probability Distribution. We construct a routability probability distribution based on the unrouted nets or terminals and the routing environment (i.e., horizontal congestion). Firstly, we identify the cell columns that have unrouted nets or terminals (i.e., unrouted column set U) and set their unroutable probability to 1. Secondly, the routability probability is spread to adjacent cell columns considering the distance and routing environment (i.e., horizontal congestion) of cell columns to construct the probability distribution as shown in the following equation.

$$P_{rout,k} = r_k \sum_{u_i \in U} e^{-\frac{(k-u_i)^2}{2\sigma^2}}, 0 \le r_k \le 1 \tag{2}$$

Where k indicates the k^{th} column in the cell and u_i is the i^{th} element in unrouted column set U. σ is set based on the design rule of pin access metal layers (i.e., end-of-line spacing of M0 and contact rules) in half contacted poly pitch (CPP) unit. r_k is the routing environment value of k^{th} column in the cell. Here, the r_k is calculated based on the horizontal routing congestion ratio as shown in the following formulation.

$$r_k = \frac{t_{h,k}}{T_{h,k}} \tag{3}$$

Where $t_{h,k}$ is the number of used horizontal routing tracks at k^{th} column in the cell, and $T_{h,k}$ is the number of usable horizontal routing tracks at k^{th} column in the cell. Lastly, we use a gaussian kernel to smooth $P_{rout,k}$ distribution. Figure 3 shows an example of generated routability distribution of a high drive strength latch designed for clock trees. The generated routability probability distribution is used to train the lattice graph routability model.

3.2.2 Lattice Graph Routability Model Architecture. Figure 4 shows the overview of lattice graph routability model. The input of lattice graph routability model is a graph which includes the transistor placement, external pin placement, and circuit net connection. After feeding the lattice graph structure input to the model, the graph neural net extracts the lattice graph embeddings. Then, the graph embeddings are fed into shared multilayer perception (MLP) network. The output of shared MLP is fed into routing congestion and routability probability MLP networks to predict $\hat{y}_{reg}$ and $\hat{y}_{rout}$ of each column in the transistor placement.

Lattice graph. The lattice graph of each transistor placement consists of front-end-of-line (FEOL) lattice grid, external pin layer lattice grid, and circuit net. Here, the external pin placement lattice grid is optional and depends on the output of the placer. For example, the placer in [1] determines the external pin locations. As a result,

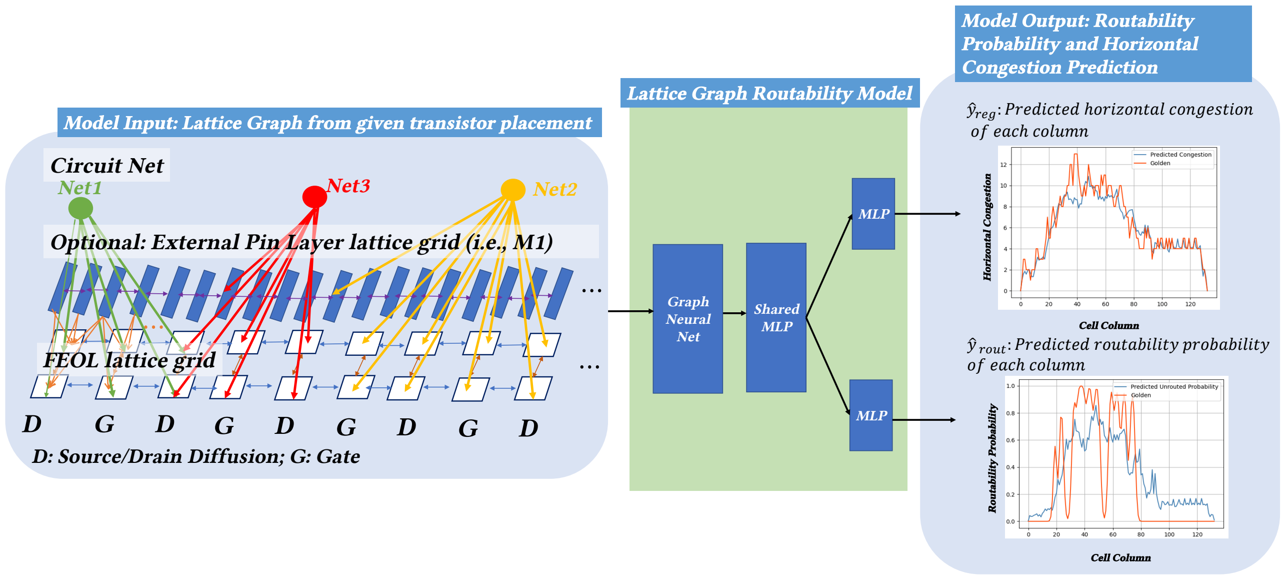

Model Input: Lattice Graph from given transistor placement
Circuit Net
Net1
Net3
Net2
Optional: External Pin Layer lattice grid (i.e., M1)
FEOL lattice grid
D G D G D G D G D
D: Source/Drain Diffusion; G: Gate
Lattice Graph Routability Model
Graph Neural Net
Shared MLP
MLP
MLP
Model Output: Routability Probability and Horizontal Congestion Prediction
$\hat{y}_{reg}$: Predicted horizontal congestion of each column
Horizontal Congestion
Cell Column
Predicted Congestion
Golden
$\hat{y}_{rout}$: Predicted routability probability of each column
Routability Probability
Predicted Unrouted Probability
Golden

Figure 4: Overview of Lattice Graph Routability Model.

the lattice graph routability model includes the external pin lattice grid to support more accurate routability prediction.

Front-of-line lattice grid: The FEOL lattice grids are on the device layer and consist of gate, source, and drain terminals. The net information of gate, source, and drain terminals on the FEOL lattice grid are extracted from the transistor placement. The FEOL lattice grid node feature can include the current type of terminal (i.e., Diffusion or Gate), number of contact access points, number of required contacts, average PDA metric at the column, etc. The horizontal edge type between FEOL lattice grid nodes is gate to diffusion. The vertical edge type is dynamically determined by the nets of PMOS and NMOS region. If the nets of PMOS and NMOS region are the same, the vertical edge type is common gate or diffusion. Otherwise, the vertical edge type is split gate or diffusion.

External pin layer lattice grid: For the placement with standard cell external pins information, the placed external pins can be extracted on the external pin layer lattice grid. Here, we use M1 (i.e., vertical routing layer) for the standard cell external pin layer as an example. The node features of the external pin layer lattice grid node includes the usage for standard cell pins, number of via points to connect to lower metal layers, etc. There are two types of edges of external pin layer lattice grid nodes. The first type is external layer to external layer, and the second type is external layer to FEOL grid.

Circuit net: The circuit nets are the logic nets in the standard cell netlist. The circuit net node features include the number of terminals that need to be connected (#terms), vertical span, horizontal span, etc. Here, the number of terminals is dynamically decided based on the diffusion sharing and diffusion break of transistors. For example, in Figure 5(a), the #terms of net32 is 0 if transistor XM_I5 and XM_I6 share the diffusion. However, the #terms of net32 is 2 if transistor XM_I5 and XM_I6 are not sharing the diffusion as shown in Figure 5(b). There are two types of edges of circuit net nodes. The first type is direct terminal link which connect the corresponding net terminals on FEOL and external pin layer lattice grids. The second type is span link which connect the FEOL and external pin layer lattice grids within the bounding box of the net.

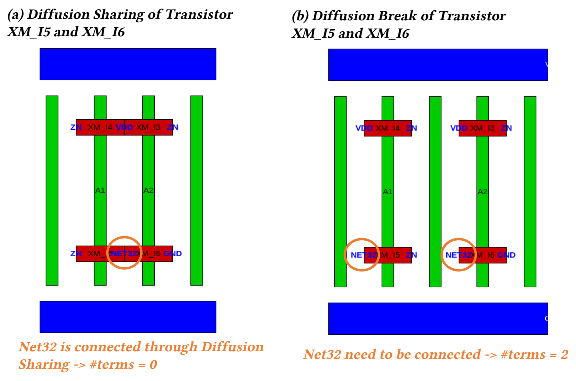

(a) Diffusion Sharing of Transistor XM_I5 and XM_I6
(b) Diffusion Break of Transistor XM_I5 and XM_I6
A1
A2
Net32 is connected through Diffusion Sharing -> #terms = 0
Net32 need to be connected -> #terms = 2

Figure 5: Example of the number of terminals need to be connected (#terms) with diffusion sharing and diffusion break.

Training model. Predicting routability inside a standard cell is a challenging problem, since it is related to not only routing congestion, but also the transistor pin accessibility, which is highly dependent on the number of metal tracks (i.e., M0) in the cell, contact rules, the number of contact points, and the design rules of metals (i.e., M0). Here, we use a jointing supervision approach to achieve the routability prediction in the cell. Let the $\hat{y}_{reg}/\ y_{reg}$ be the vector of predicted/golden horizontal congestion of each column in the cell. The $\hat{y}_{rout}/\ y_{rout}$ is the vector of predicted/golden routability probability distribution of cell columns. We use mean square error loss function for the horizontal congestion prediction as shown in Equation (4).

$$L_{reg} = -\frac{1}{N_c}\sum(y_{reg} - \hat{y}_{reg})^2 \tag{4}$$

Where N_c is the number of columns in the cell. To calculate the routability probability loss (i.e., L_{rout}), we firstly calculate the log SoftMax of y_{rout} and $\hat{y}_{rout}$ (i.e., Y_{rout}, and $\hat{Y}_{rout}$), respectively.

Then, we apply KL divergence loss for routability prediction with routability probability distribution as shown in the following equations.

$$Y_{rout,i} = log(\frac{e^{y_{rout,i}}}{\sum_j e^{y_{rout,j}}}), \hat{Y}_{rout,i} = log(\frac{e^{\hat{y}_{rout,i}}}{\sum_j e^{\hat{y}_{rout,j}}}), i = 1, ..., N_c$$
$$L_{rout} = D_{KL}(Y_{rout} \| \| \hat{Y}_{rout}) = Y_{rout} log(\frac{Y_{rout}}{\hat{Y}_{rout}}) \quad (5)$$

Equation (6) shows the final loss, and it is used for backward propagation.

$$L = L_{reg} + L_{rout} \quad (6)$$

Cell-Level Routability Metric Extraction. We develop a cell-level routability metric to consider the device pin accessibility and global routing net together using the predicted horizontal congestion ($\hat{y}_{reg}$) and routability probability ($\hat{y}_{rout}$). Given the device placement, we can obtain the transistor pin density map (d_{pin}) of cell columns. Here, by applying element-wise multiplication of $\hat{y}_{reg}$ and $\hat{y}_{rout}$, the product vector indicates difficulty in crossing the net across each cell column. Similarly, by applying element-wise multiplication of d_{pin} and $\hat{y}_{rout}$, the product vector indicates the difficulty of the transistor pin being accessed in each cell column. Then, we can calculate the *PinAccessScore* and *CongestionScore* with operations (i.e., max, sum, average, top-k, etc.) of cell columns for cell-level routability metric. Here, we use top-k operation since the routability issues happen in certain of local regions in the cell. They are written in the Equation (7).

$$\begin{aligned} PinAccessScore &= \frac{\sum TopK(d_{pin} * \hat{y}_{rout})}{K} \\ CongestionScore &= \frac{\sum TopK(\hat{y}_{reg} * \hat{y}_{rout})}{K} \end{aligned} \quad (7)$$

Finally, the cell-level routability metric, R_{cell}, can be written as the sum of *PinAccessScore* and *CongestionScore* to indicate the difficulty of the given device placement for the router to route

3.3 Routability-Driven Placement with Multi-Objective Optimization

We introduce our routability-driven placement approach through multi-objective optimization scheme.

Routability-driven placement objectives. In the placer, we use the cell-level PDA metric (P_{cell}) and cell-level routability metric (R_{cell}) of the lattice graph routability model to guide the placer engine [1] for routability-driven optimization in our framework. Equation (8) shows the multi-objectives of the implemented simulated annealing algorithm [1] to optimize the routability through the P_{cell} and R_{cell}.

$$\textbf{Minimize } w_a \times CW + w_m \times \overline{WL} + w_{pda} \times P_{cell} + w_{pred} \times R_{cell} \quad (8)$$

Where the w_a, w_m, w_{pda}, and w_{pred} are Weightings of the cell width (i.e., CW), estimated wirelength (i.e., $\overline{WL}$), cell-level PDA metric (P_{cell}), and cell-level routability metric (R_{cell}), respectively.

Multi-objective optimization. Optimizing multiple objectives (i.e., cell area, total wirelength, and routability) simultaneously can be computationally expensive. In Equation (8), larger w_{pda} and w_{pred} improves the routability, but it may lead to larger cell area. However, larger w_a could potentially lead to poor routability.

We use multi-objective BOHB [3, 4] method for multi-objective optimization, which includes multi-objective tree-structured parzen Estimator (MOTPE) [4], and HyperBand [13]. MOTPE is a multi-objective Bayesian optimization algorithm for the hyperparameter optimization with categorical hyper-parameters in a tree-structured [4]. Hyperband searches the best time allocation for each of the hyper-parameter configurations [13]. Here, we perform multi-objective BOHB with w_a, w_{pda}, and w_{pred} parameters to optimize the cell area and routability together.

3.4 Dynamic External Pin Allocation for Routing

We introduce our novel dynamic standard cell external pin allocation methodology for various routing algorithms (i.e., Maze routing).

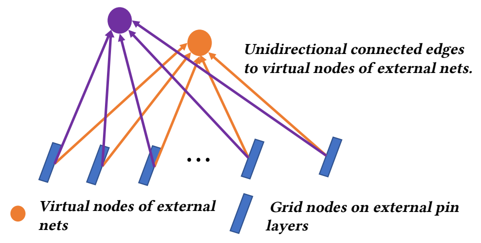

Unidirectional connected edges to virtual nodes of external nets.
...
Virtual nodes of external nets
Grid nodes on external pin layers

Figure 6: The illustration of dynamic standard cell external pin allocation.

The dynamic standard cell external pin allocation enables the router to construct the standard cell external pin shapes dynamically considering the routability and DRCs. Figure 6 shows the illustration of the dynamic standard cell external pin allocation methodology. Firstly, the virtual node of each external net is created. Then, we construct the edges with the direction from the candidate grid nodes on external pin layers to the virtual nodes of all the external nets. Thirdly, we add the virtual node of each external net to the routing terminals. Lastly, we start the routing procedure as described in [12]. The router needs to connect the terminals of external net to the virtual node terminal through the external pin layers. As a result, the pin shape of external net is automatically constructed in the routing phase. Notice that our dynamic standard cell external pin allocation can be easily adopted for various well known routing algorithms, such as Maze routing, A^* search, etc.

4 EXPERIMENTAL RESULTS

Our work is implemented with Python and runs on a server with multiple Intel Xeon CPUs where a maximum of 20 threads are used in the implementation. It generates a simplified grid-based cell layout, which is given to a separate Perl program called Sticks to handle DRC checking and conversion to tapeout quality Cadence Virtuoso layout.

We select 94 complex and hard to route standard cell designs in an advanced node from an industrial technology. The number of transistors is from 14 to 114. There are 57 Flip-flops, 21 Combinational cells, and 15 Latchs as shown in Figure 7. The experiments are organized as follows:

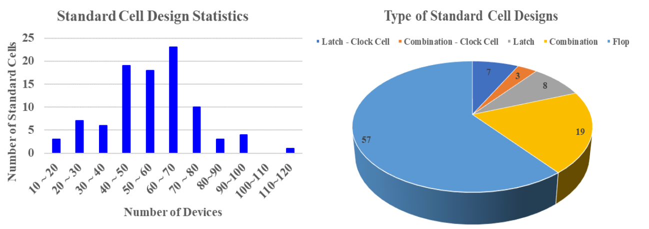


Figure 7: The number of devices and design types of selected 94 standard cells.

- Exp. 4.1 (Routability Metric Accuracy Study): We validate the proposed routability metric, R_{cell}, with commonly used routability metrics (i.e., congestion, PDA, etc.) and the area of golden unrouted probability distribution.
- Exp. 4.2 (Routability Experiment): We study the routability improvement of PDA metric, lattice graph routability model, and dynamic pin allocation methodology using the selected 94 complex cells.
- Exp. 4.3 (Multi-Objective Optimization): We perform Multi-Objective BOHB [3, 4] on cell area, routability, and DRC to generate optimized cell layouts.

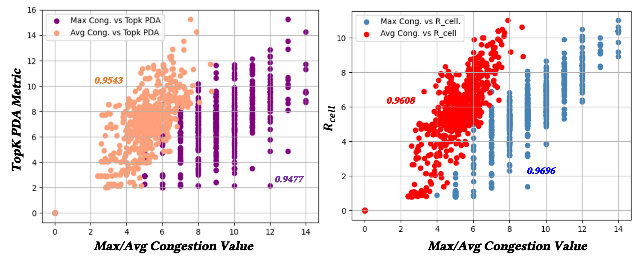


Figure 8: The correlations between proposed cell-level routability metric (i.e., P_{cell} and R_{cell}) and max and average congestion values after routing.

4.1 Cell-Level Routability Metric Accuracy

To verify that the proposed cell-level PDA and routability metrics can truly guide the placer to find routable placements, we generate 2240 different transistor placements and collect the max and average congestion values after routing the 2240 transistor placements. Then, we calculate the correlations of the proposed cell-level PDA and routability metrics (i.e., P_{cell} and R_{cell}) to the max and average congestion values. Figure 8 shows that the proposed P_{cell} and R_{cell} are highly correlated to the max and average congestion values.

Furthermore, we calculate the correlations between the area of golden unrouted probability distribution, which is used to train the model as introduced in Section 3.2.1, and max congestion, average congestion, TopK PDA metric, and the proposed cell-level routability metric (R_{cell}), respectively. The proposed cell-level routability metric (R_{cell}) has the strongest correlation (i.e., 0.8536) to the area of golden unrouted probability distribution among these routability related metrics as shown in Table 1. From the above studies, we show that the proposed cell-level routability metric (R_{cell}) is more efficient and accurate to guide the placer to find routable cell placement than max congestion, average congestion, and TopK PDA metric.

Table 1: Correlation value Table. Correlation between the area of golden unrouted probability distribution and max congestion, average congestion, TopK PDA metric, and the proposed cell-level routability metric (R_{cell}), respectively.

Metric Name	Correlation with the area of golden unrouted probability distribution
Max Congestion	0.7922
Average congestion	0.8030
TopK PDA metric	0.8364
R_{cell}	0.8536

4.2 Routability Experiment

We compare the routability improvement of the PDA metric, lattice graph routability model, and dynamic pin allocation to the previous work [1] using the selected complex 94 standard cells of over 1000 single-row standard cells in an industrial standard cell library. For other cells in the library, the R_{cell} is relatively small compared to CW and $\overline{WL}$ in Equation (8) since there is no routability issue. The placer mainly optimizes cell width and estimated wirelength as the same as [1].

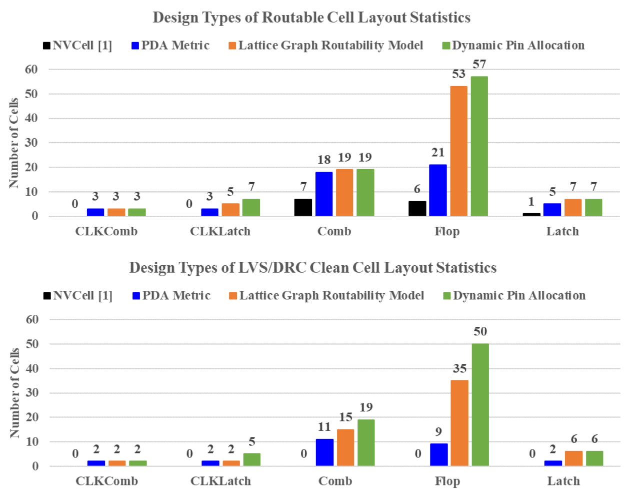
Figure 9 shows the statistics of routable and LVS/DRC clean layouts of [1], PDA metric, lattice graph routability model, and dynamic pin allocation. Compared to [1], the proposed PDA metric (i.e., P_{cell}) improves the number of routable cells by 38.3%, and the number of LVS/DRC clean cells by 27.7%. After leveraging the lattice graph routability model, compared to the proposed PDA metric, the number of routable and LVS/DRC clean cells are further improved by 39.3% and 36.1%. Lastly, enabling the dynamic external pin allocation in the routing phase can achieves 98.9% and 87.2% routable and LVS/DRC clean cells. In summary, compared to [1], using PDA metric, lattice graph routability model, and dynamic external pin allocation together improves the number of routable and LVS/DRC clean cells by 84.0% and 87.2%.

For the performance of lattice graph routability model, the inference time per lattice graph is around 0.5 seconds, which is more than 1000× faster than performing the routing. However, the placement runtime is increased by 4.33× on average than without using lattice graph routability model because the model inference is performed for every actions in the simulated annealing based placement algorithm [1]. In summary, NVCell 2 standard cell design automation framework can produce 98.9% LVS/DRC clean cell layouts using the novel PDA metric, lattice graph routability model, and dynamic pin allocation approach over 1000 single-row standard cells in an industrial standard cell library.

4.3 Multi-Objective Optimization

We apply multi-objective BOHB [3, 4] method to explore the weights (i.e., w_a, w_{pda}, and w_{pred}) in Equation (8) for cell area, routability, drcs, and total wirelength. Firstly, we extract the pareto to get the optimal weight candidates of the given metric axes (i.e., CW, $drcs$, TWL, etc.). If there is no dominant vector in the given metric axes, we pick the optimal weight candidates based on the priority of the give metric axes to obtain the optimal weight candidate according to the priority of metric axes.

Figure 10 shows the extracted pareto points of each metric axe (i.e., cell width, drcs, and total wirelength) and selected candidate point. Here, we select the weight configurations without unrouted nets for pareto extraction. Here, the baseline cell width is from



	Routable Cells	LVS/DRC Clean Cells
NVCell [1]	14.9%	0.0%
PDA Metric	53.2%	27.7%
Lattice Graph Rout. Model	92.5%	63.8%
Dynamic Pin Allocation	98.9%	87.2%

Figure 9: The statistics of the number of routable and LVS/DRC clean cell layouts generated by [1], PDA metric, lattice graph routability model, and dynamic pin allocation. The number of routable cell layouts of [1], PDA metric, lattice graph routability model, and dynamic pin allocation are 14, 50, 87, and 93, respectively. The number of LVS/DRC clean cell layouts of [1], PDA metric, lattice graph routability model, and dynamic pin allocation are 0, 26, 60, and 82, respectively.

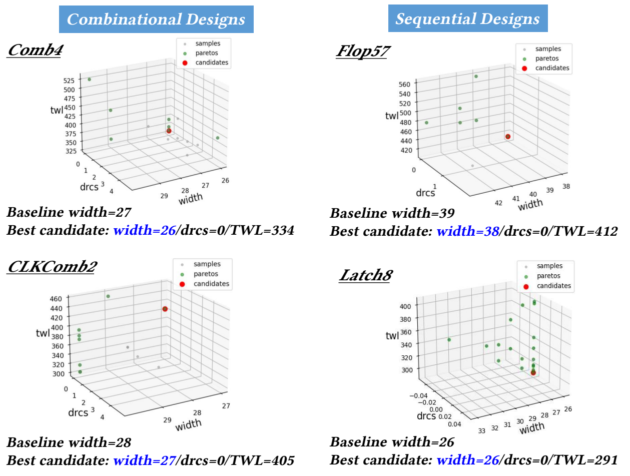


Figure 10: The results of Multi-Objective BOHB for combinational and sequential cells. The grey point is the sample runs. The green point is the pareto of each metric axe. The red point is the selected candidate point. The proposed routability-driven standard cell design automation framework can not only improves the routability but also achieves better cell level metric (i.e., cell width). The cell names are renamed based on the design type and number due to intellectual property consideration.

the manual designed industrial standard cell library. The multi-objective optimization approach improves the number of smaller cell layouts and reduces the number of larger cell layouts by 18.05% and 54.30%, respectively, on the benchmark of 94 hard-to-route standard cells using the baseline cell width from the manual designed industrial library. The runtime of multi-objective optimiaztion depends on the number of samples, iterations, and workers. All the runs of a cell in multi-objective BOHB optimization process include placement, routing, and drc fixing.

In summary, the proposed framework achieves 13.9% smaller cell layouts than the existing industrial standard cell library over 1000 cells as shown in Figure 11.

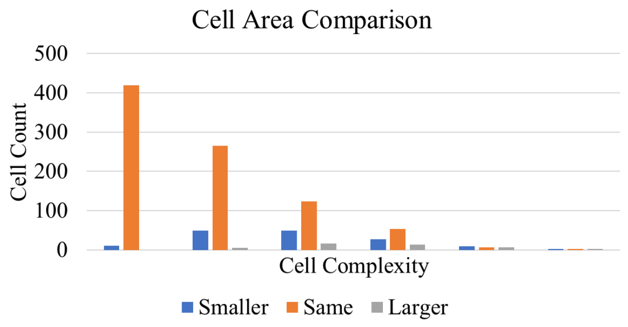


Figure 11: Cell area comparison with existing industrial library. NVCell 2 generates smaller layouts for 13.9% of cells and larger layouts for 4.3% of cells.

5 CONCLUSION

We propose a routability-driven standard cell synthesis framework using a novel pin density aware congestion metric, lattice graph routability modelling approach, and dynamic external pin allocation methodology to generate optimized layouts to improve the routability of standard cell designs beyond *5nm*. We firstly demonstrate that the proposed framework improves the routable and LVS/DRC clean cell layouts by 84.0% routable and 87.2%, respectively, compared to [1] using the 94 complex and hard to route standard cells. In total, the proposed framework can generate 98.9% LVS/DRC clean cell layouts over 1000 standard cells in an industrial standard cell library. Then, we demonstrate that the proposed routability-driven standard cell design automation framework can generate smaller cell layouts for 13.9% of cells compared to an existing industrial standard cell library of over 1000 cells through the multi-objective BOHB [3, 4] process.

We believe that the important directions for future research include: extending the standard cell design automation framework for multi-heights architecture and utilizing reinforcement learning technique to improve the performance of standard cell design automation.

REFERENCES

[1] Haoxing Ren and Matthew Fojtik. Nvcell: Standard cell layout in advanced technology nodes with reinforcement learning. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 1291–1294. IEEE, 2021.

[2] Pascal Van Cleeff, Stefan Hougardy, Jannik Silvanus, and Tobias Werner. Bonncell: Automatic cell layout in the 7-nm era. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(10):2872–2885, 2019.

[3] Stefan Falkner, Aaron Klein, and Frank Hutter. Bohb: Robust and efficient hyperparameter optimization at scale. In *International Conference on Machine Learning*, pages 1437–1446. PMLR, 2018.

[4] Yoshihiko Ozaki, Yuki Tanigaki, Shuhei Watanabe, and Masaki Onishi. Multiobjective tree-structured parzen estimator for computationally expensive optimization problems. In *Proceedings of the 2020 genetic and evolutionary computation conference*, pages 533–541, 2020.

[5] Ang Lu, Hsueh-Ju Lu, En-Jang Jang, Yu-Po Lin, Chun-Hsiang Hung, Chun-Chih Chuang, and Rung-Bin Lin. Simultaneous transistor pairing and placement for cmos standard cells. In *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1647–1652. IEEE, 2015.

[6] Yih-Lang Li, Shih-Ting Lin, Shinichi Nishizawa, Hong-Yan Su, Ming-Jie Fong, Oscar Chen, and Hidetoshi Onodera. Nctucell: A dda-aware cell library generator

for finfet structure with implicitly adjustable grid map. In *Proceedings of the 56th Annual Design Automation Conference 2019*, pages 1–6, 2019.

[7] Daeyeal Lee, Dongwon Park, Chia-Tung Ho, Ilgweon Kang, Hayoung Kim, Sicun Gao, Bill Lin, and Chung-Kuan Cheng. Sp&r: Smt-based simultaneous place-and-route for standard cell synthesis of advanced nodes. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 40(10):2142–2155, 2020.

[8] Chung-Kuan Cheng, Chia-Tung Ho, Daeyeal Lee, and Dongwon Park. A routability-driven complimentary-fet (cfet) standard cell synthesis framework using smt. In *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–8. IEEE, 2020.

[9] Chung-Kuan Cheng, Chia-Tung Ho, Daeyeal Lee, Bill Lin, and Dongwon Park. Complementary-fet (cfet) standard cell synthesis framework for design and system technology co-optimization using smt. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 29(6):1178–1191, 2021.

[10] Chung-Kuan Cheng, Chia-Tung Ho, Daeyeal Lee, and Bill Lin. Multirow complementary-fet (cfet) standard cell synthesis framework using satisfiability modulo theories (smts). *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, 7(1):43–51, 2021.

[11] Vincent A Cicirello. On the design of an adaptive simulated annealing algorithm. In *Proceedings of the international conference on principles and practice of constraint programming first workshop on autonomous search*, 2007.

[12] Haoxing Ren and Matthew Fojtik. Standard cell routing with reinforcement learning and genetic algorithm in advanced technology nodes. In *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, pages 684–689, 2021.

[13] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017.