High-Correlation 3D Routability Estimation for Congestion-guided Global Routing

Miaodi Su[1], Hongzhi Ding[1], Shaohong Weng[1], Changzhong Zou[1], Zhonghua Zhou[2], Yilu Chen[1], Jianli Chen[3] and Yao-Wen Chang[4]

[1]Fuzhou University

[2]The University of British Columbia

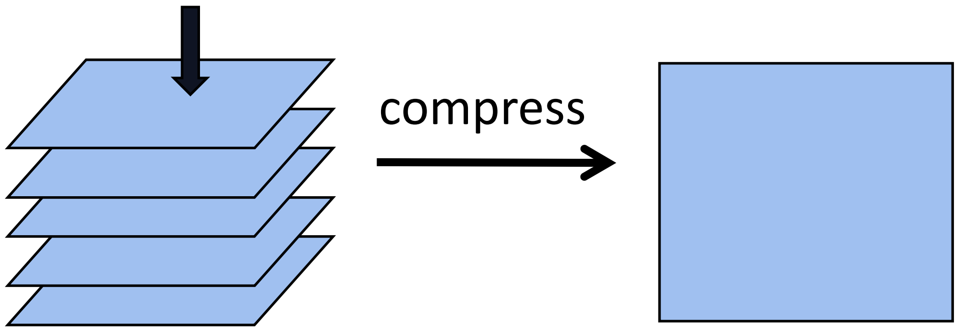
[3]Fudan University

[4]National Taiwan University

Outline

Router

2D router:

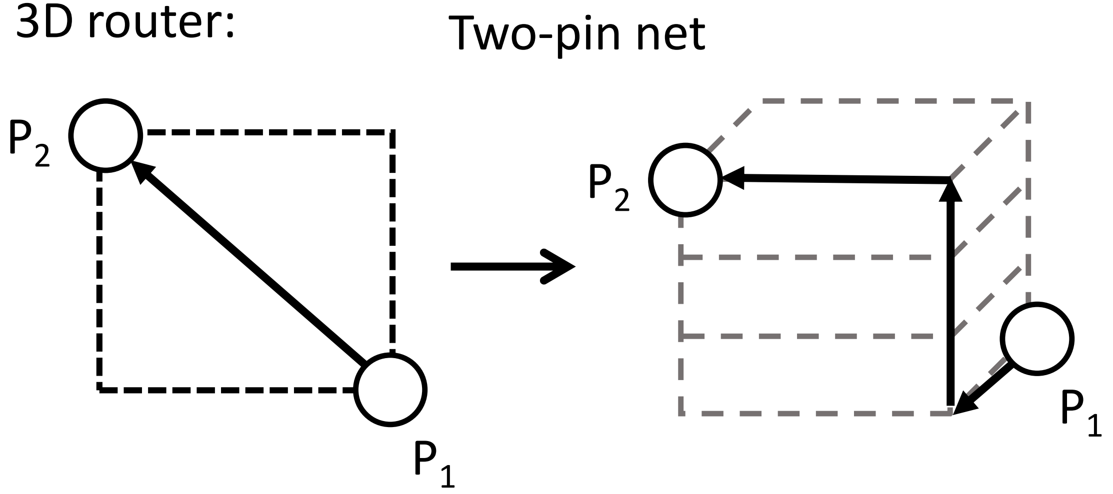


Two-dimensional routing followed by layer assignment

Advantage: Fast routing

Disadvantage: Loss of structural information and poor routing quality

3D router:



CUGR: Combine pattern routing and layer assignment

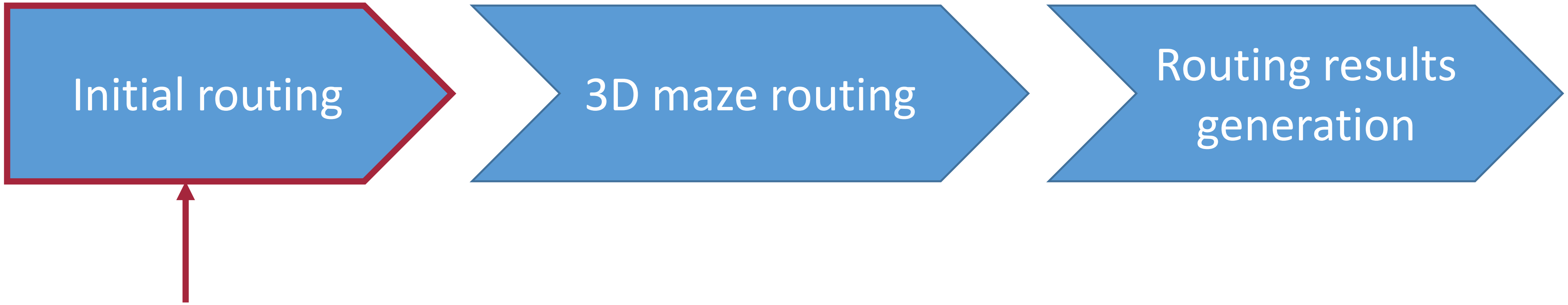
Advantage: Good routing quality

Disadvantage: **Consider current best routing solution rather than a predicted globally optimal routing one.**

Model prediction

- Using **machine learning** method, the model is used to **predict the routing solution** before global routing, and then the prediction results are used to **guide global routing**.

CUGR routing flow

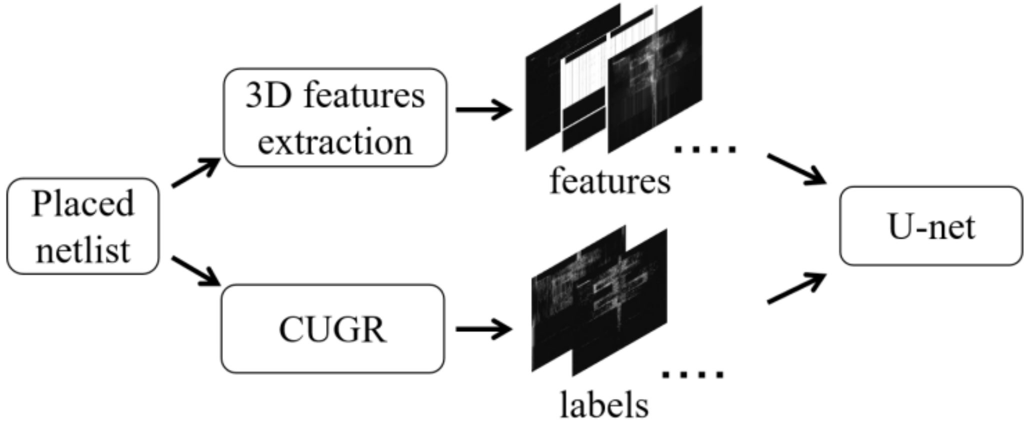


The model predicts and guides in the initial routing stage

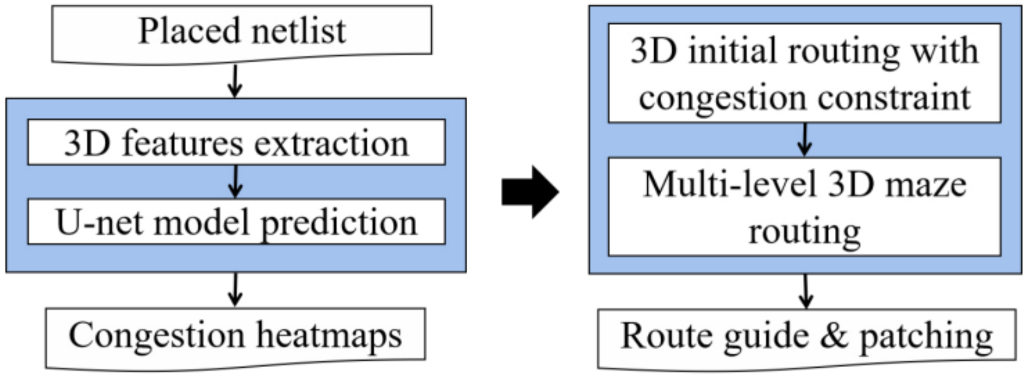
Outline

- Introduction
- **Problem Formulation**
- Routability Prediction
- Congestion-guided Global Routing
- Experimental Results
- Conclusions

Problem Definition



Model training



Model prediction and guided routing

Three main tasks:

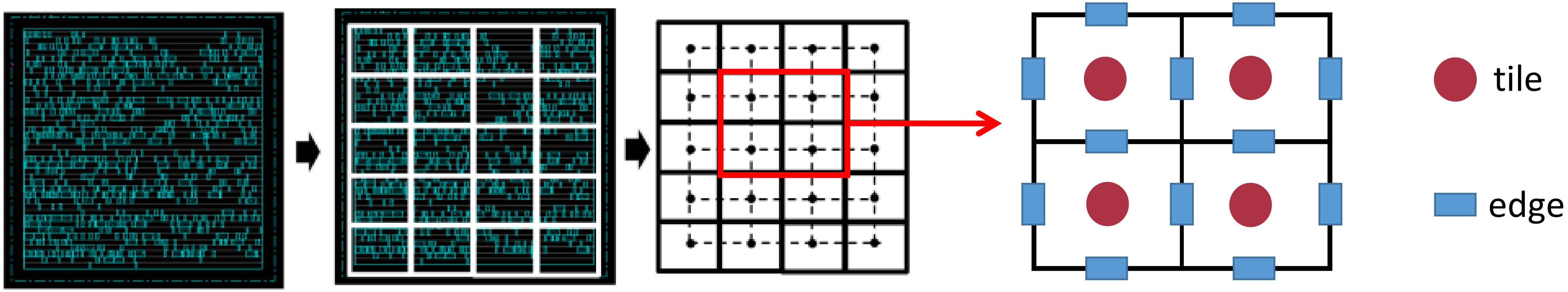
- design an effective **deep learning model** as a congestion estimator;
- extract appropriate **3D features and labels** for model training;
- develop an effective methodology for **global routing guided by the congestion estimator**.

Outline

- Introduction
- Problem Formulation
- **Routability Prediction**
- Congestion-guided Global Routing
- Experimental Results
- Conclusions

Feature Extraction

- **Pin density:** The number of pins in a tile.
- **Net capacity:** The number of wires that can be allocated within one tile (on one layer) for the nets.
- **Net density:** Improved RUDY method in a 3D space.

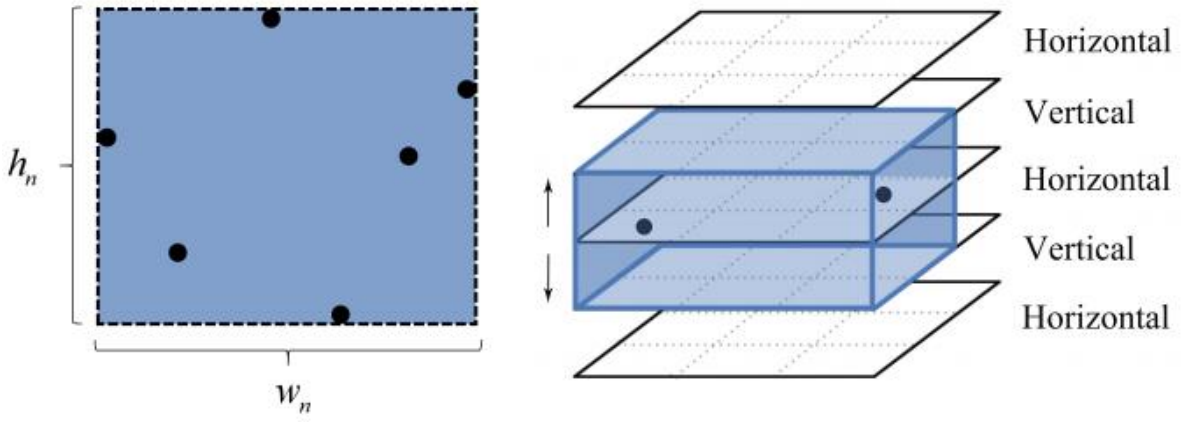


- From design to tiles to routing grid;
- A tile is a pixel of a feature map.

3D RUDY Method

2D RUDY method:

$$R(n) = \frac{HPWL(n) \times p(n)}{\boxed{w_n \times h_n}}$$



(a) 2D bounding box (b) 3D bounding box

Improved 3D RUDY method:

$$r_h(n) = \boxed{\frac{w_n}{w_n + h_n}} \times \frac{1}{y_h(n)}, r_v(n) = \boxed{\frac{h_n}{w_n + h_n}} \times \frac{1}{y_v(n)},$$

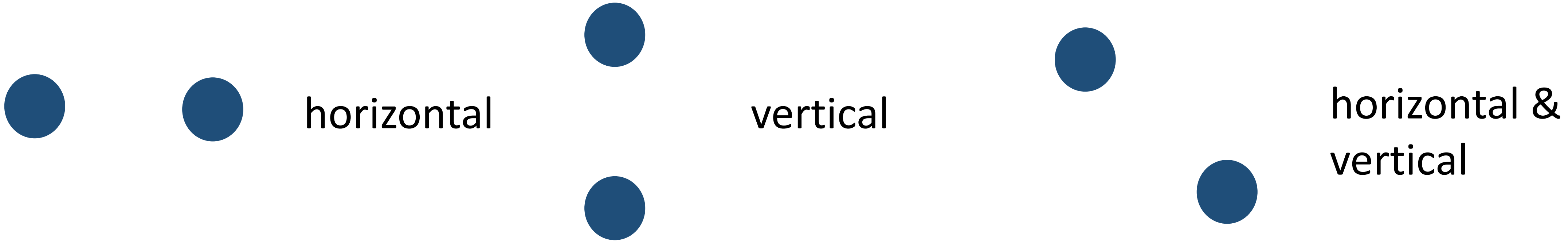
$$R^{3D}(n, i) = r_i(n) \times \frac{HPWL(n) \times p(n)}{w_n \times h_n},$$

- Determine **the number of layers (depth) of b-box** according to the relative positions of pins;
- Determine **the net proportion allocated to each layer** according to the length and width of the b-box.

3D RUDY Method

Determine the number of layers (depth) of b-box

➢ Determine routing direction according to pin position



- The pin layers and all layers between them are covered in the net b-box.
- **Up-and-down one layer expansion** will be performed when the current b-box lacks layers with the desired routing direction.

Feature map

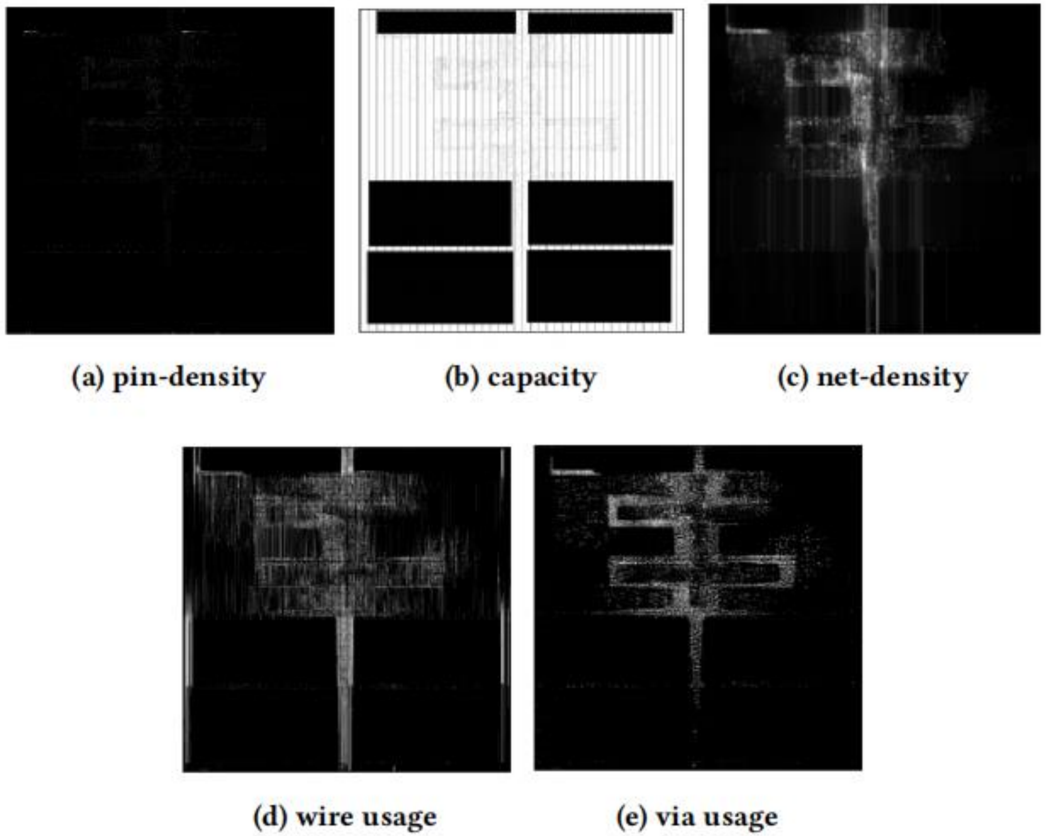


Figure 3: Gray images of features and labels in the first layer of ***pci_bridge32_b_md3*****.**

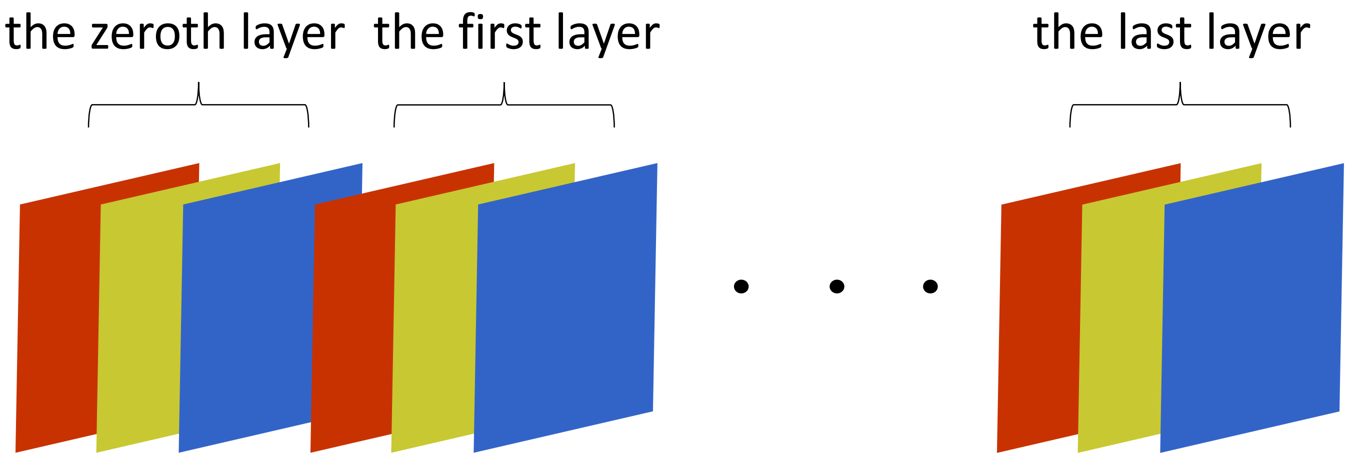
Label:

- **Wire-usage:** The number of tracks in a wire edge occupied by routed nets.
- **Via-usage:** The number of vias in a wire edge after routing.

- **Each feature or label of each layer of a design is represented by a gray image.**

Feature map

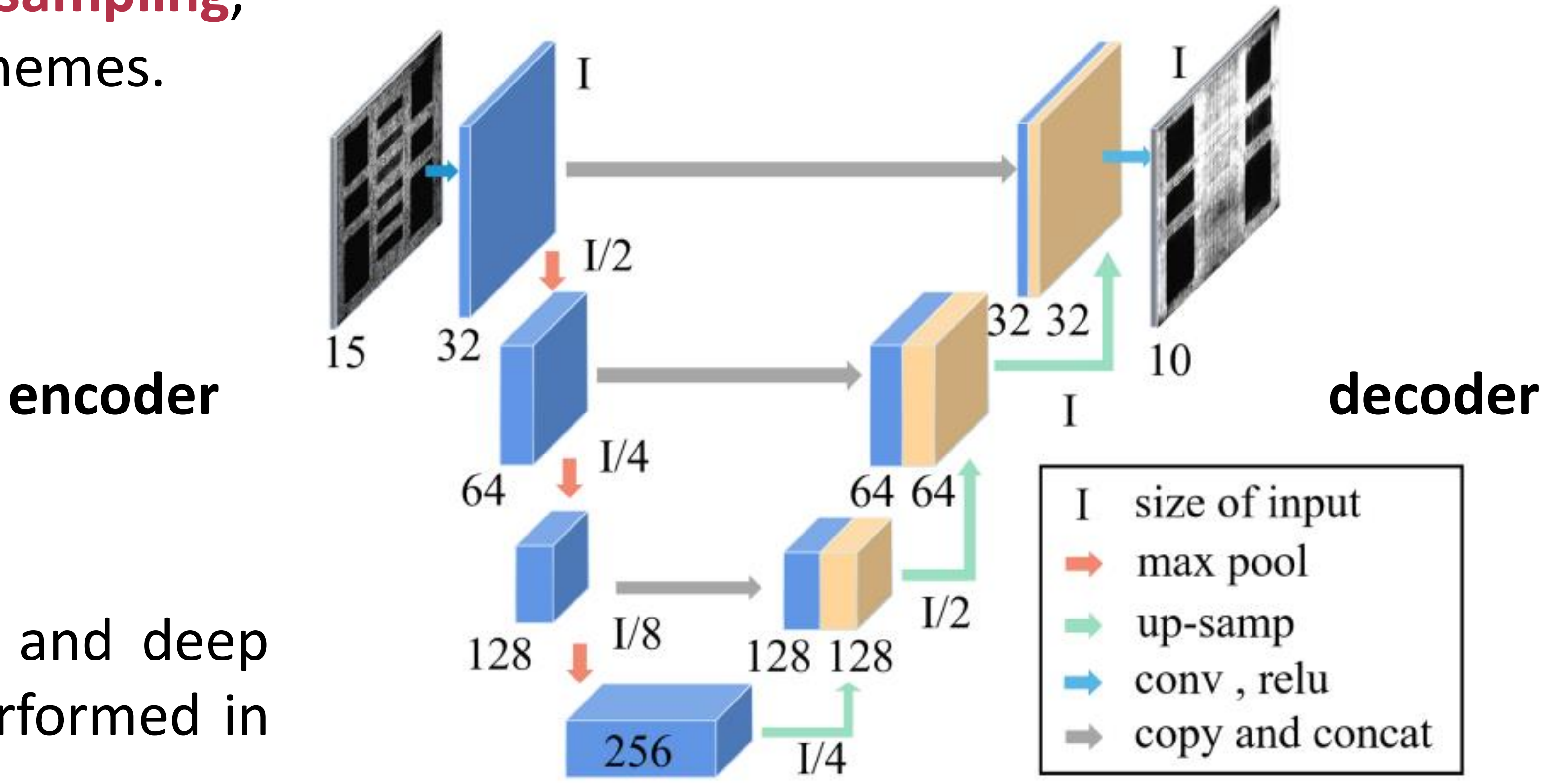
Structure of the feature map:



- A color represents a feature.
- A multi-channel feature image with ***m*n*** channels.

U-Net Model

- The core idea of the U-Net is the **down-sampling**, **up-sampling**, and **skip connection** schemes.



- Multi-scale prediction and deep supervision can be performed in the model.

Model Training

- **Features:** pin-density, net-capacity, net-density
- **Labels:** wire-usage and via-usage
- **Benchmarks:** ISPD15, ICCAD17, ICCAD19
- **Data pre-processing:**

    1. Feature extraction
    2. Channel combination
    3. Sample incision

Algorithm 1 Model Training Pre-processing

Require: Placed netlist.
Ensure: Training set.

1

pin_den, net_capacity, net_den, wire_usage, via_usage = Init_array();
while net ← nets.read() **do**
  net_den[net.position] = $\text{RUDY}^{3\text{D}}$(net) ;
  while pin ← net.getNextPin() **do**
    pin_den[pin.position]+ = 1;
  end while
end while
3DPatternRouting();
Multi-level3DMazeRouting();
net_capacity ← getCapacityMap();
wire_usage ← getWireUsageMap();
via_usage ← getViaUsageMap();

2

Features: X ← Combine(pin_den, net_capacity, net_den);
Labels: Y ← Combine(wire_usage, via_usage);
$d = \{d_1, d_2\}$ ← The number of channels of feature and label maps;

3

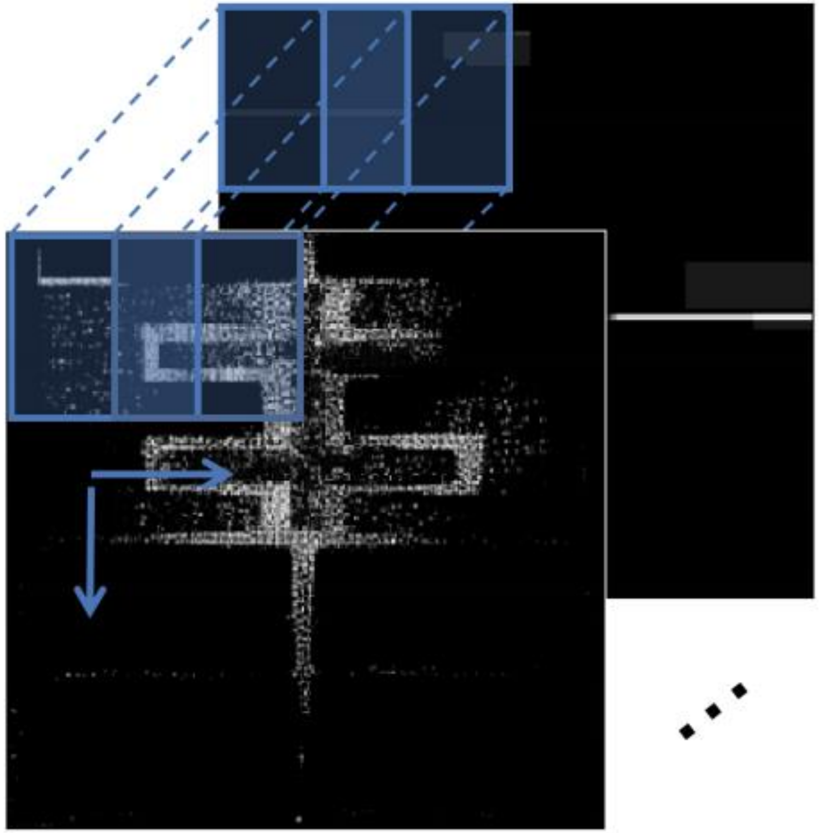
Training set ← Incise $X, Y \in R^{d \times w_n \times h_n}$ to $X_n, Y_n \in R^{d \times 64 \times 64}$;

return Training set.

Model Training

- **Training set:** Cut the design into ***n*64*64*** samples (***n*** is the number of layers). The sample number of the training set is:

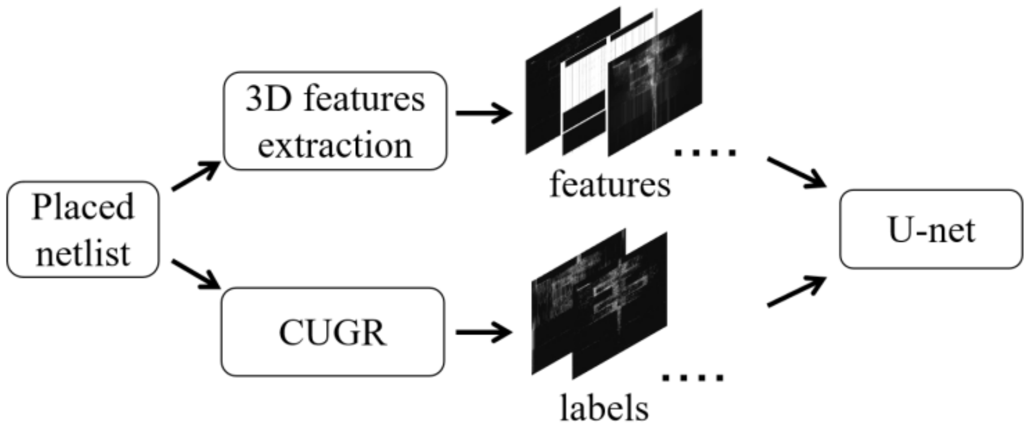
    - **ISPD15**: 9311
    - **ICCAD17**: 4058
    - **ICCAD19**: 44010



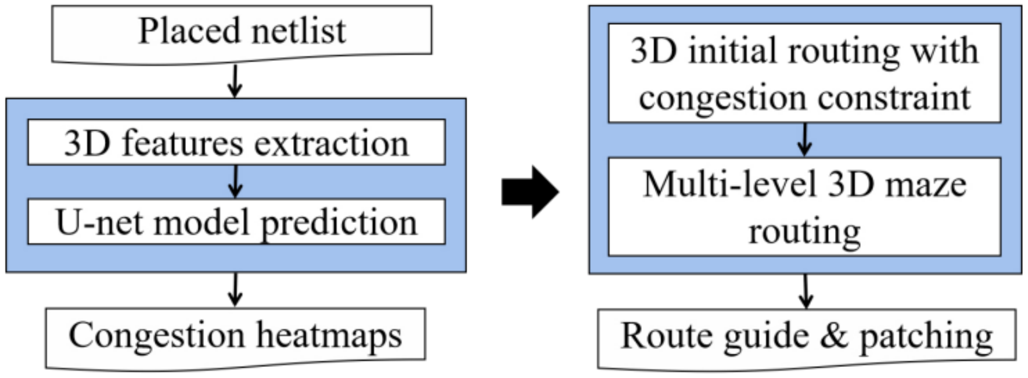
Outline

- Introduction
- Problem Formulation
- Routability Prediction
- **Congestion-guided Global Routing**
- Experimental Results
- Conclusions

Congestion-guided Global Routing



Model training



Model prediction and guided routing

Some terminologies:

- **Capacity:** c(u, v), the maximum number of tracks that can route through the edge;
- **Demand:** d(u, v), the number of nets already routed through the edge or occupied by fixed macros.
- **Utilization:** t(u, v), the proportion of the capacity occupied by routed nets and fixed macros.

$$t(u,v) = \frac{d(u,v)}{c(u,v)}.$$

Congestion-guided Global Routing

- Develop a **congestion prediction constraint** to modify the congestion cost function of the initial routing.

$$g(u,v) = wl(u,v) \times t(u,v) \times c_o, \quad (5)$$

Original congestion cost function

$$pu(u,v) = \frac{\boxed{pw(u,v) + pv(u,v)}}{c(u,v)}, \quad (6)$$

Predicted results of the model: Wire-usage & via-usage

$$\hat{g}(u,v) = \begin{cases} g(u,v) \times \frac{pu(u,v)+1}{2}, & \text{if } t(u,v) < 1-\epsilon, \\ \infty, & \text{if } t(u,v) \geq 1-\epsilon, \end{cases} \quad (7)$$

New congestion cost function

- apply the guided routing method for the first 70% of nets, while the last 30% use CUGR's initial routing.

Purpose: We make front nets **avoid the original highly congested area** when routing, and the later nets are routed in the avoided area with no routed nets, which can reduce congestion effectively.

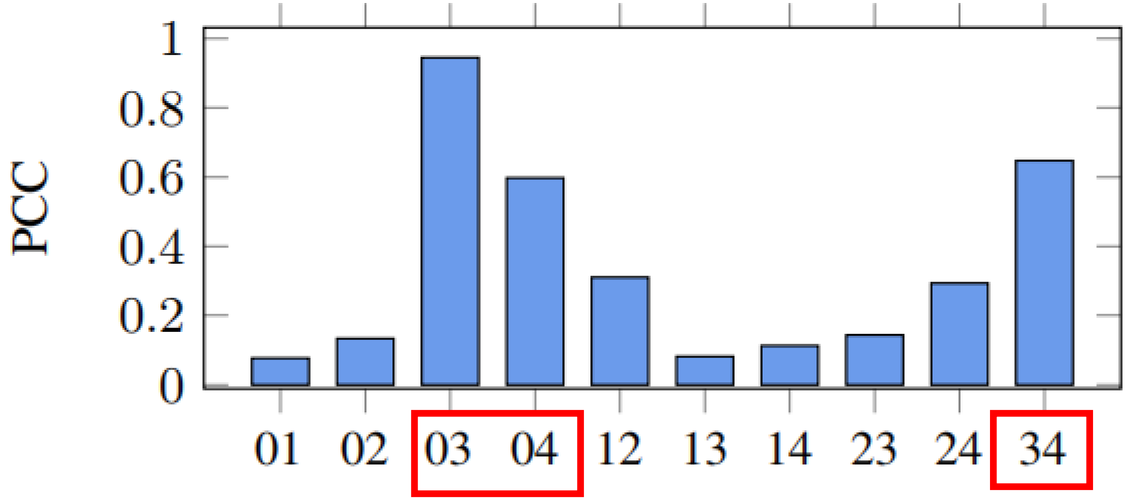
Outline

- Introduction
- Problem Formulation
- Routability Prediction
- Congestion-guided Global Routing
- **Experimental Results**
- Conclusions

Feature Selection

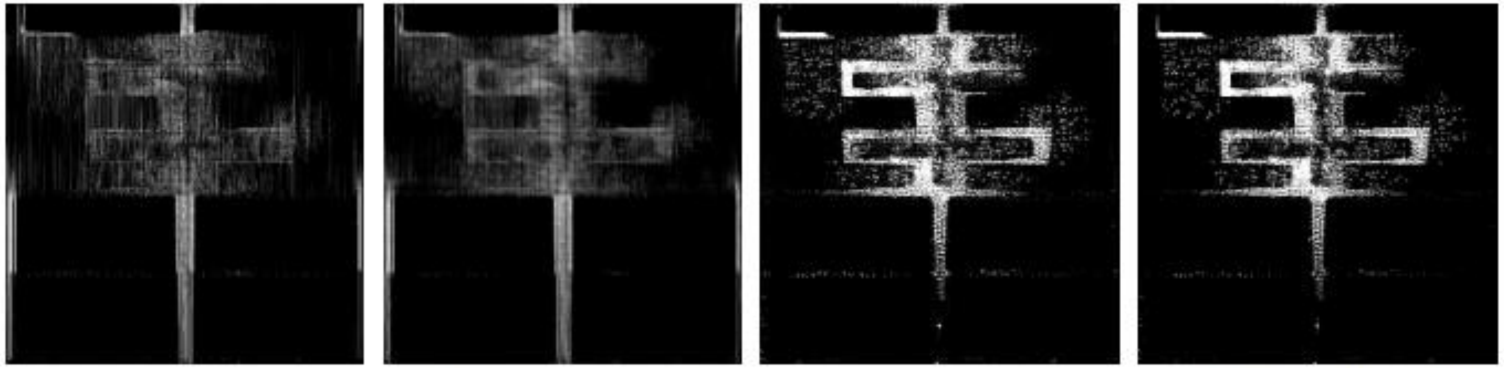
- The **PCC (Pearson correlation coefficient)** was used to calculate the correlation between features.

Numbers	Features
0	pin-density
1	net-capacity
2	net-density
3	neighbor-pins
4	NCPR



Model Estimation Quality

- **Benchmarks:** ISPD15, ICCAD17, ICCAD19. They are divided into training benchmarks and testing benchmarks.
- Model training took about 7 hours, while the prediction time is less than two seconds on the GPU.



(a) Ground wire-usage (b) Predicted wire-usage (c) Ground via-usage (d) Predicted via-usage

Ground congestion heatmaps vs. predicted congestion heatmaps.

Model Estimation Quality

TABLE II: Congestion Estimation Quality Comparison.

Designs	PCC		MANE		SDNE		Prediction time(s)	
	dc-GAN [18]	Ours	dc-GAN [18]	Ours	dc-GAN [18]	Ours	dc-GAN [18]	Ours
mgc_des_perf_1	0.757	0.857	0.256	0.139	0.188	0.156	0.585	0.435
mgc_des_perf_a	0.817	0.907	0.147	0.051	0.107	0.112	1.139	1.071
mgc_fft_2	0.605	0.860	0.234	0.112	0.240	0.141	1.163	1.061
mgc_fft_a	0.693	0.879	0.222	0.028	0.168	0.079	1.855	1.067
mgc_matrix_mult_2	0.796	0.889	0.170	0.115	0.236	0.135	1.435	1.058
mgc_matrix_mult_c	0.707	0.907	0.127	0.027	0.119	0.079	1.116	1.116
mgc_pci_bridge32_a	0.708	0.819	0.121	0.119	0.191	0.164	1.240	1.073
mgc_superblue16_a	0.693	0.828	0.133	0.095	0.193	0.173	1.560	1.071
mgc_superblue19	0.703	0.795	0.100	0.062	0.217	0.184	1.525	1.080
des_perf_1	0.690	0.828	0.105	0.182	0.295	0.222	1.104	1.043
des_perf_b_md2	0.721	0.885	0.189	0.100	0.184	0.111	1.208	1.034
edit_dist_1_md1	0.611	0.882	0.199	0.115	0.142	0.135	1.257	1.047
fft_2_md2	0.639	0.806	0.164	0.135	0.205	0.145	1.106	1.037
pci_bridge32_a_md1	0.723	0.899	0.150	0.067	0.150	0.092	1.822	1.024
pci_bridge32_b_md3	0.638	0.916	0.153	0.024	0.079	0.054	1.844	1.045
ispd18_test1	0.708	0.881	0.147	0.034	0.165	0.065	0.972	0.092
ispd18_test6	0.649	0.733	0.204	0.060	0.157	0.091	1.938	1.638
ispd18_test8	0.708	0.867	0.087	0.023	0.165	0.050	1.009	1.146
ispd19_test7	0.665	0.793	0.198	0.033	0.137	0.068	1.715	1.185
ispd19_test8	0.599	0.806	0.128	0.039	0.267	0.068	1.260	1.216
ispd19_test9	0.756	0.828	0.095	0.042	0.170	0.068	1.305	1.265
ispd18_test8_metal5	0.811	0.850	0.090	0.032	0.173	0.069	1.892	1.465
ispd19_test7_metal5	0.711	0.799	0.069	0.043	0.120	0.081	1.783	1.632
average	0.700	0.848	0.152	0.073	0.177	0.111	1.384	1.083

Three metrics:

- **PCC:** Pearson correlation coefficient
- **MANE:** the mean absolute normalized error
- **SDNE:** the standard deviation in the normalized error

PCC↑ 21.14%

MANE↓ 51.97%

SDNE↓ 37.29%

Model Estimation Quality

TABLE III: Global Routing Quality Metrics (C: CUGR routing results; G: our guided routing results)

Benchmarks	Total Overflow			Wire Length			Via Count			Run time (s)		
	C	G	ratio (%)	C (E+07)	G (E+07)	ratio (%)	C (E+05)	G (E+05)	ratio (%)	C	G	ratio
mgc_des_perf_1	496	196	-60.48	0.1325	0.1305	-1.46	4.0288	3.9365	-2.29	60.677	67.113	1.11
mgc_des_perf_a	6201	6123	-1.26	0.2113	0.2115	0.08	4.1542	4.1577	0.04	78.422	101.42	1.29
mgc_fft_2	80	39	-51.25	0.2480	0.2426	-2.17	1.4424	1.3584	-5.82	12.439	14.609	1.17
mgc_fft_a	1047	949	-9.36	0.4090	0.3944	-3.57	1.3596	1.2766	-6.11	18.592	27.948	1.50
mgc_matrix_mult_2	3181	2506	-21.22	1.4000	1.3841	-1.84	5.6231	5.3898	-4.15	70.638	76.731	1.09
mgc_matrix_mult_c	6973	6733	-3.44	0.3400	0.3398	-0.06	5.9752	5.9710	-0.07	135.42	171.50	1.27
mgc_pci_bridge32_a	1969	1909	-3.05	0.2890	0.2911	-0.72	1.0141	1.0065	-0.75	10.621	13.882	1.31
mgc_superblue16_a	28575	28502	-0.26	29.990	29.975	-0.05	23.799	23.763	-0.15	645.56	597.16	0.93
mgc_superblue19	12112	10780	-11.00	16.800	16.561	-1.43	18.600	15.984	-14.06	383.44	364.48	0.95
des_perf_1	679	360	-46.98	0.1358	0.1339	-1.44	4.1540	3.9900	-3.95	70.236	72.482	1.03
des_perf_b_md2	32	22	-31.25	0.1829	0.1830	0.00	4.4150	4.4154	0.01	37.444	44.273	1.18
edit_dist_1_md1	6259	5298	-15.35	0.4173	0.4140	-0.78	6.8732	6.6331	-3.49	123.70	125.01	1.01
fft_2_md2	11	0	-100	0.2723	0.2688	-1.27	1.5956	1.5325	-3.96	9.586	10.875	1.13
pci_bridge32_a_md1	1650	1657	0.40	0.3210	0.3217	0.20	1.2026	1.2029	0.00	10.258	14.684	1.43
pci_bridge32_b_md3	822	790	-3.89	0.0827	0.0828	0.16	1.3238	1.3243	0.04	14.994	24.286	1.62
ispd18_test1	0	0	0	0.0418	0.0404	-3.37	0.2455	0.2627	7.02	2.915	4.188	1.44
ispd18_test6	0	0	0	3.4448	3.4442	-0.02	12.634	12.584	-0.40	110.54	151.31	1.37
ispd18_test8	0	0	0	63.648	63.648	0.00	21.171	21.111	-0.28	298.36	448.90	1.50
ispd19_test7	0	0	0	11.768	11.767	-0.01	30.198	30.187	-0.04	697.10	813.46	1.17
ispd19_test8	0	0	0	18.023	18.024	0.00	55.898	55.648	0.45	584.23	890.97	1.53
ispd19_test9	0	0	0	27.146	27.145	0.00	93.271	92.838	-0.46	986.95	1420.63	1.44
ispd18_test8_metal5	731	666	-8.89	0.6684	0.6682	-0.03	19.604	20.110	2.59	354.86	469.89	1.32
ispd19_test7_metal5	582	544	-6.50	10.592	10.591	-0.06	54.552	54.049	-0.92	490.95	554.49	1.13
average	3104	2916	-6.05	8.1114	8.0983	-0.02	16.223	16.032	-1.18	226.43	281.75	1.24

- Overflow, wire length and via count are all reduced;
- The 24% runtime overhead is due to the fact that guided routing involves more procedures of feature extraction, model loading, and model prediction.

Outline

Conclusion

Innovations:

- Extract appropriate 3D features, develop an improved RUDY method;
- Identify features with low correlation to each other;
- Develop an U-net based congestion estimator;
- Incorporate our proposed congestion estimation to improve global routing.

Advantages:

- Features with low correlation are **more representative**, which can **reduce the redundant information**.
- The PCC index between actual and predicted congestion is **high at about 0.848** on average, significantly **higher** than the counterpart dc-GAN by **21.14%**.
- **Reduce** the respective routing overflows, wirelength, and via count by averagely **6.05%, 0.02%, and 1.18%**, with only 24% runtime overheads.

Thanks!