TEXAS
The University of Texas at Austin

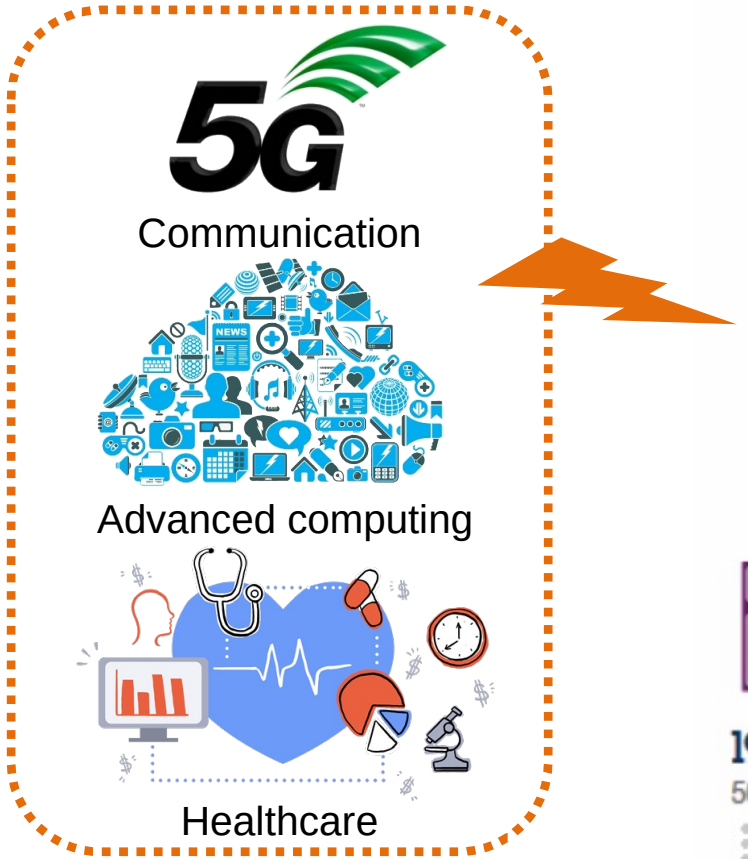
Automating Analog Constraint Extraction: From Heuristics to Learning

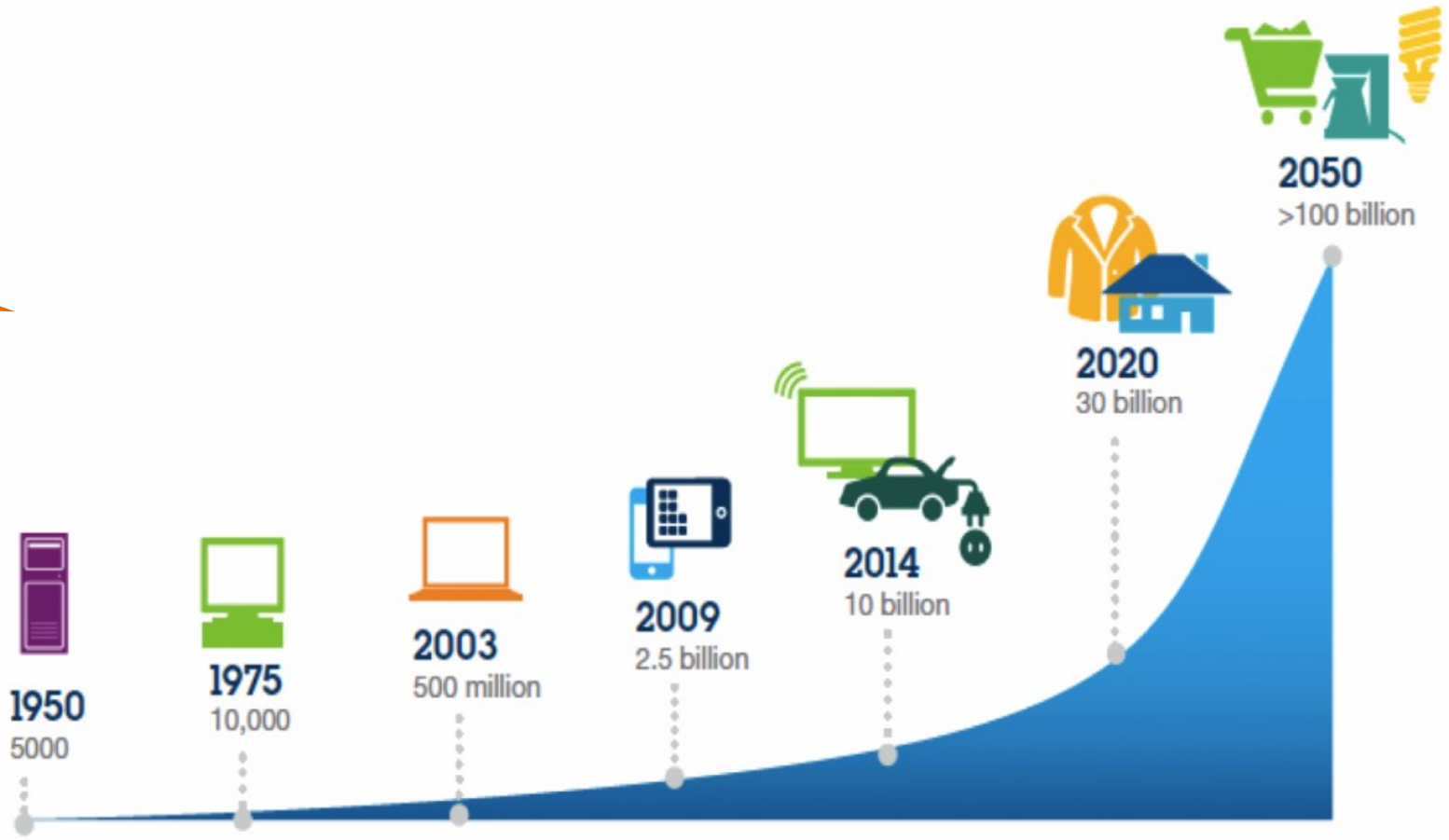
Keren Zhu, Hao Chen, Mingjie Liu and **David Z. Pan**
ECE Department
The University of Texas at Austin

This work is supported in part by DARPA and NSF

Introduction

- High demand of analog/mixed-signal IC in emerging applications
  - Internet of Things (IoT), autonomous vehicles, 5G, wearable, sensors, …





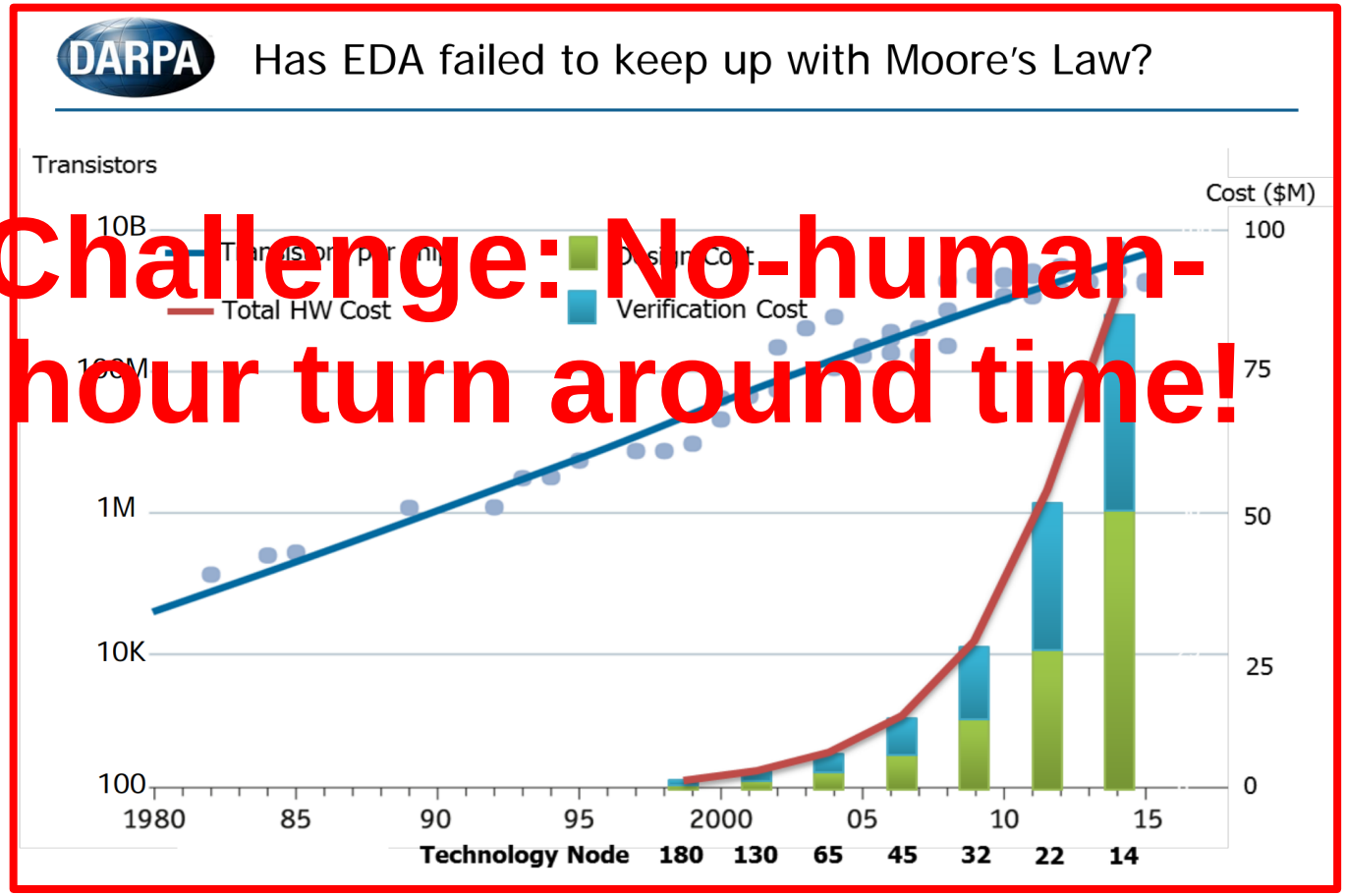
Prior Art of Analog Layout

- Analog IC layout design still heavily manual
  - Cf. digital IC layout automation
  - Very tedious and error-prone
- Modern SoCs: 20% analog, but maybe 80% design time
  - Follow complex design rules in advanced tech nodes
  - Mitigate parasitic, noise, and layout-dependent effects
  - Handle manufacturability, reliability and yield issues
- Lots of previous work on analog layout automation, **but …**
  - Placement: [Lampaert+, JSSC'95], [Strasser+, ICCAD'08], [Lin+, TCAD'09], [Ma+, TCAD'11], [Wu+, ICCAD'12], [Lin+, TCAD'16]
  - Routing: [Ozdal+, ICCAD'12], [Ou+, DAC'13], [Ou+, TCAD'16]
  - Constraint extraction: [Malavasi+, TCAD'99], [Massier+, TCAD'08]

Chip Design/Manufacturing Challenges

- Performance/Power/Area (PPA)
- Manufacturability/Yield
- Reliability
- Security
- Design cost
- …
- 2018 DARPA ERI (US$1.5B) IDEA/POSH “Silicon Compiler 2.0” (US$100M)

A. Olofsson, DARPA, ISPD-2018 Keynote

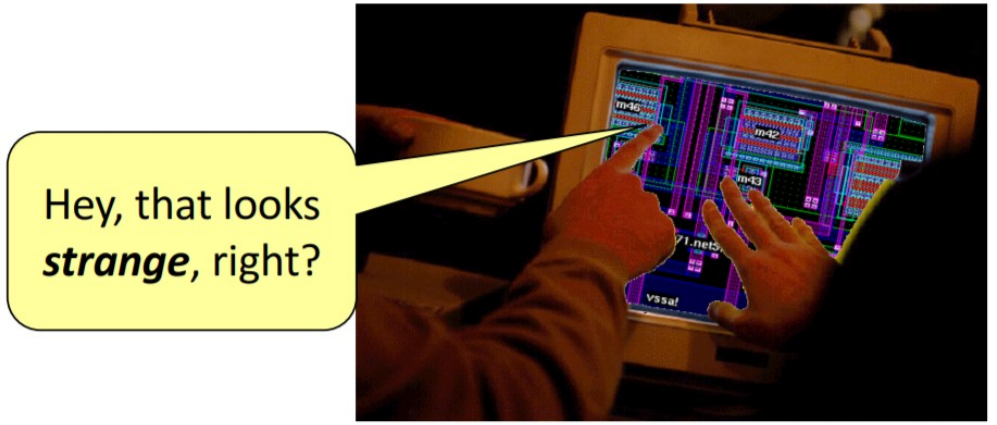


DARPA Grand Challenge: No-human-in-the-loop, 24-hour turn around time!

Challenge: No-Human-In-The-Loop

- Analog P&R uses a lot of constraints
- There are many layout techniques from designers

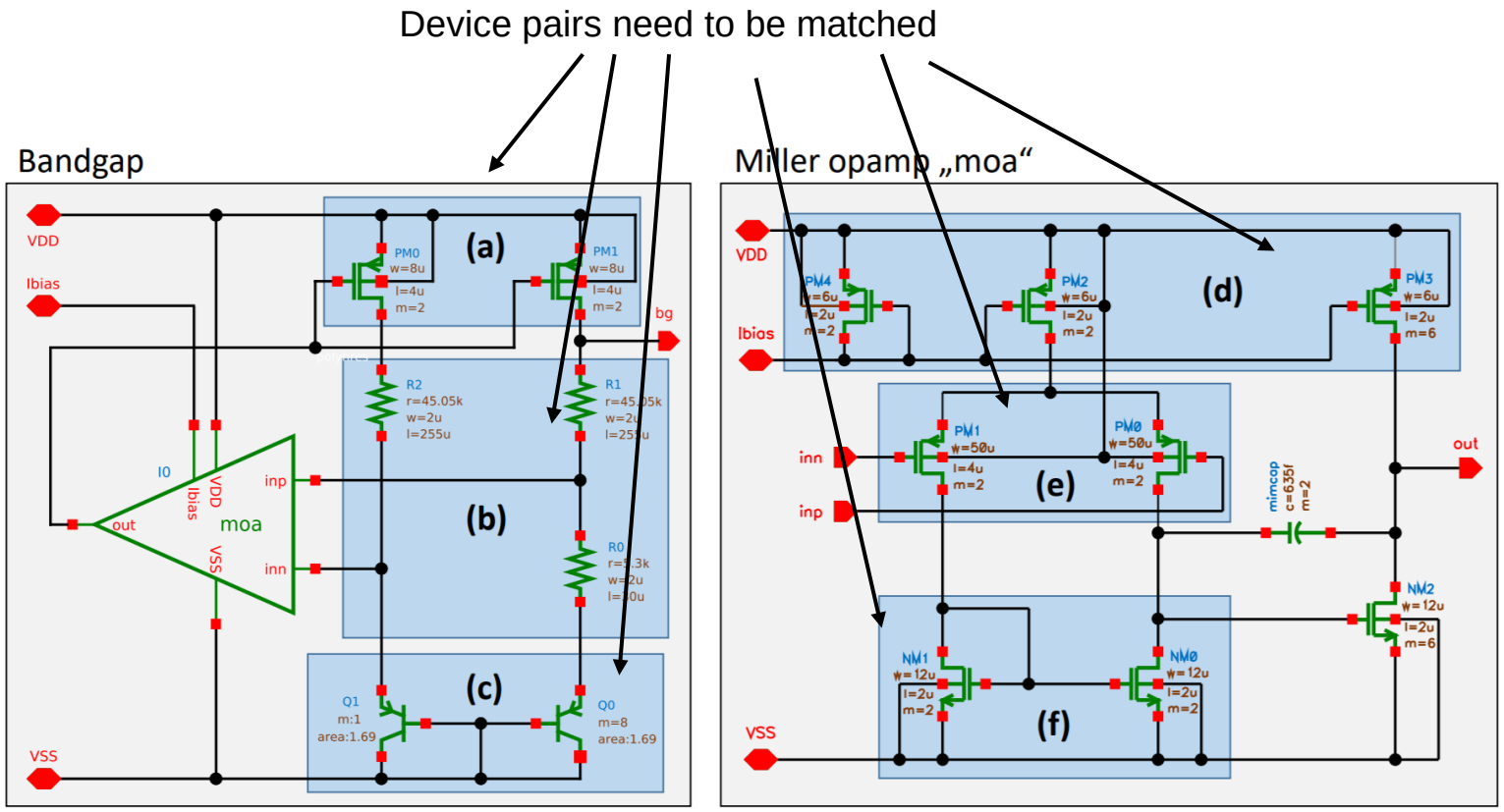
- Challenges: **automated analog constraint extraction**



[Rutenbar, *2010*]

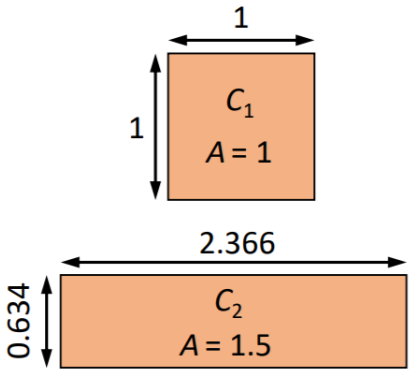
Basic Analog Constraint Formulations

- Matching is important in analog and mixed signal circuits

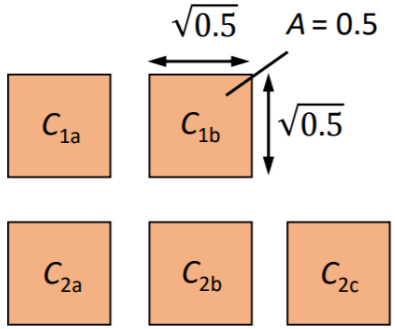


Basic Analog Constraint Formulations

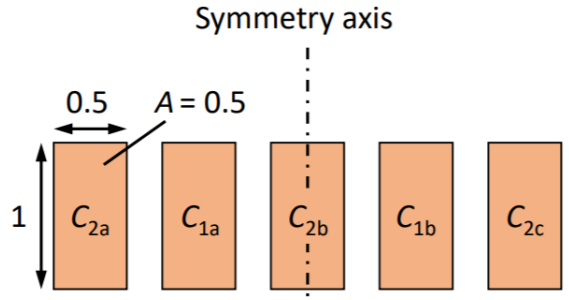
- Matched devices form certain patterns in layouts



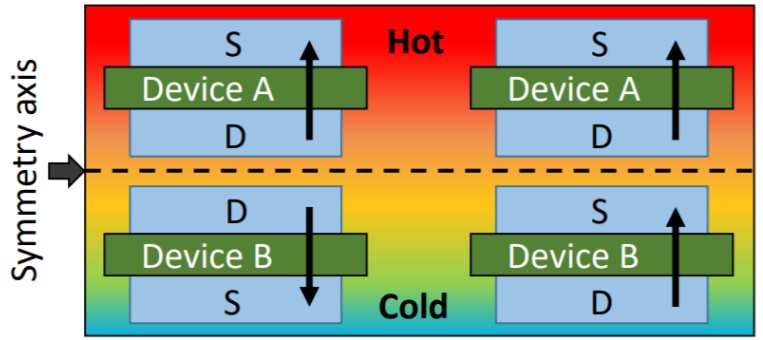
Minimum matching



Good matching



Very good matching

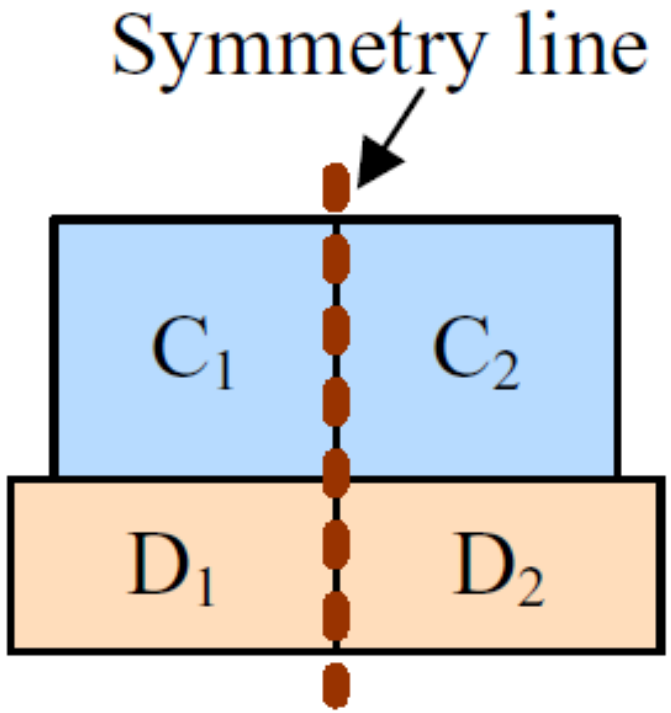


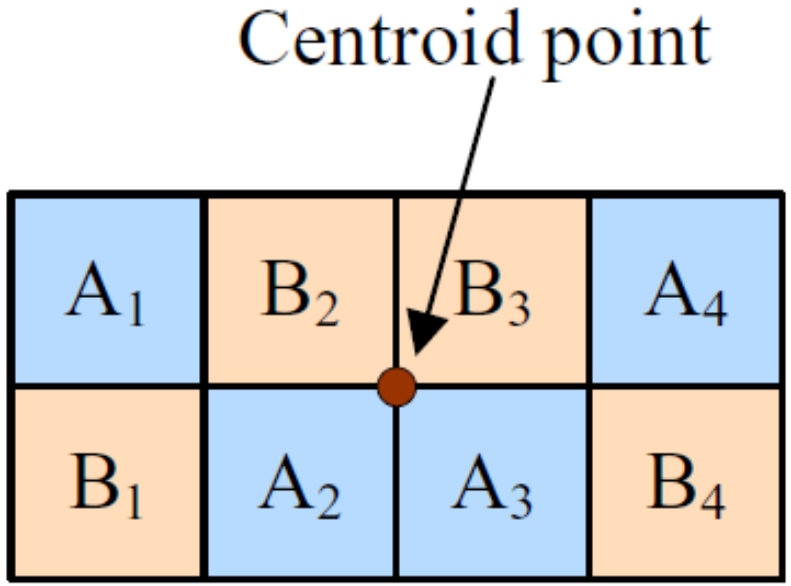
Bad matching

Good matching

Basic Analog Constraint Formulations

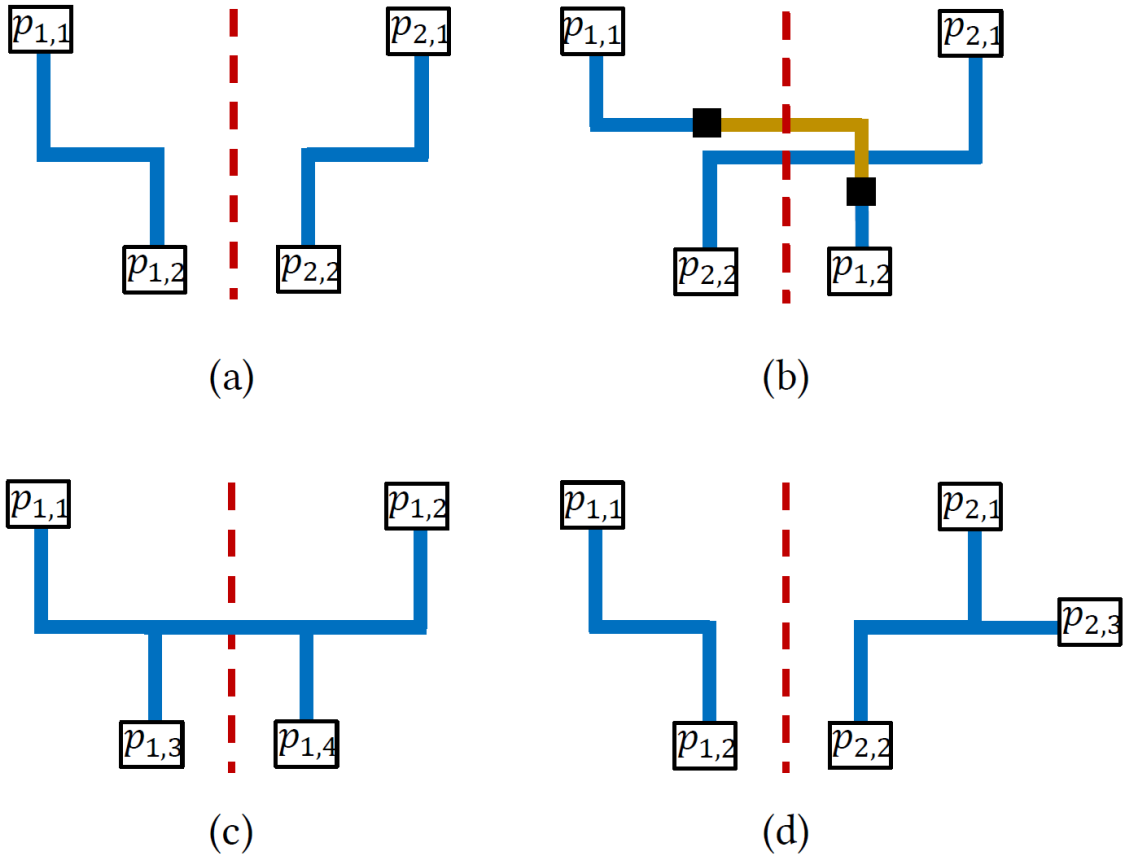
- Analog placement: **symmetry** or **common-centroid** constraints





Basic Analog Constraint Formulations

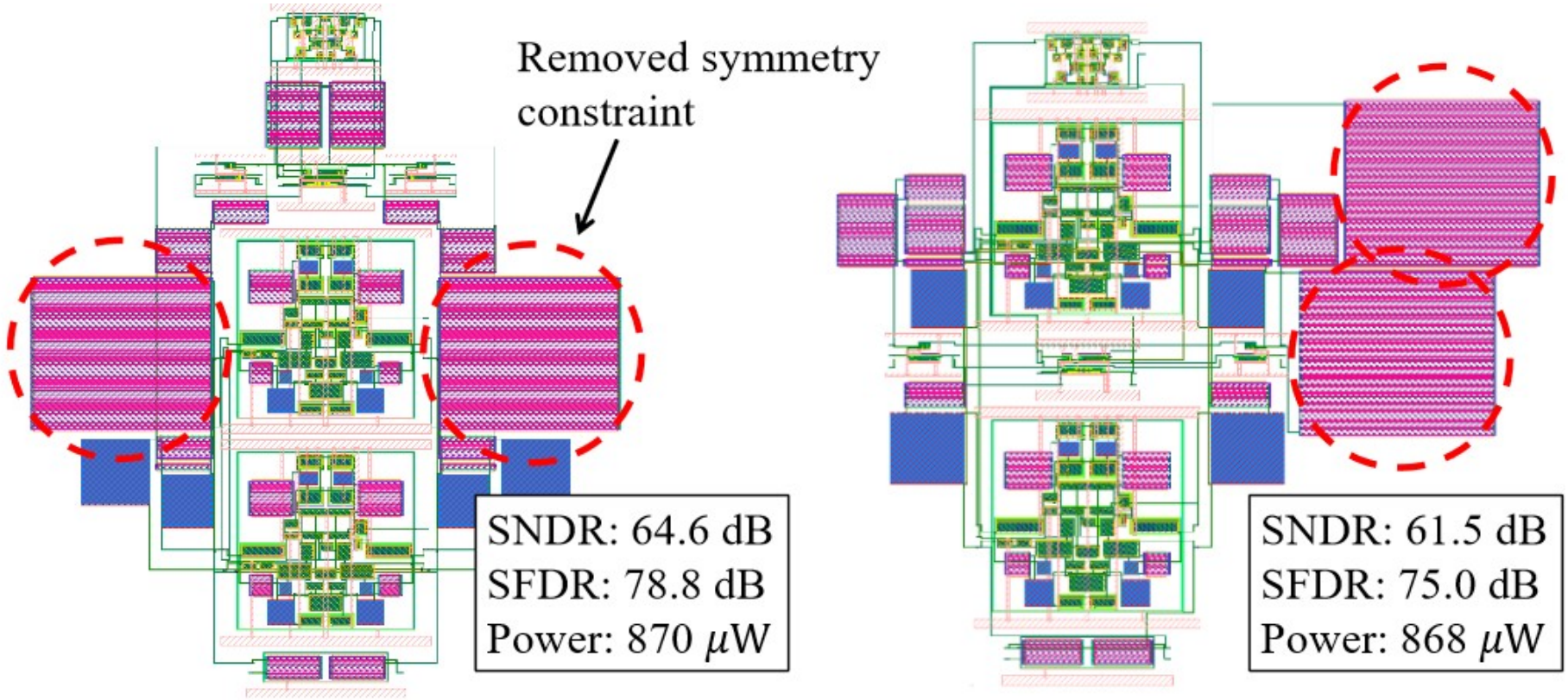
- Analog routing: symmetry constraints / objectives



[Chen+, *2021*]

Performance Degradation From Asymmetry

- Failed to honoring matching results in performance degradation
- Second order CTDSM ADC example
  - > 3dB drop in SNDR and SFDR even with smaller wirelength



Other Analog Layout Constraints

- There are a lot of more constraints
- Placement
  - Monotonic current path [Ou+, 2013] [Wu+, 2014] [Patyal+, 2018]
  - Thermal effect [Liu+, 2007]
  - System signal path [Zhu+, 2020]
- Routing
  - Avoid active region [Xiao+, 2010]
  - Power routing [Lin+, 2012]
- …

Symmetry Constraint Extraction Problem

- Problem: Given circuit netlist, detect devices that need to be matched (placed & routed symmetric) in the layout
- Convectional approaches:
  - Simulation-based methods [Choudhury+, 1990] [Charbon+, 1993]
  - Heuristic-based methods [Massier+, 2008] [Eick+, 2011] [Hao+, 2004] [Zhou+ 2005] [Wu+, 2008]

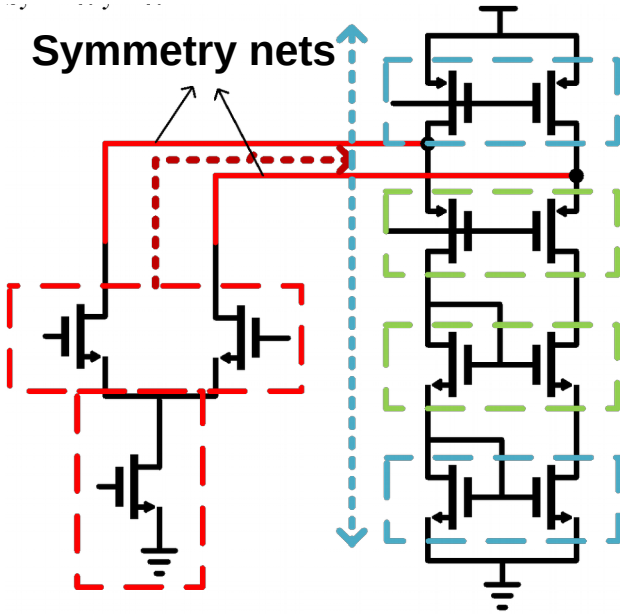
Simulation-based Methods

- Repetitive simulations with sensitivity analysis
- Correlate device parameter to performance sensitivity with matching constraints
- Pros:
  - Performance oriented
  - Recognized constraint with high confidence
- Cons:
  - Require numerous simulations
  - Difficult to detect all matching

Heuristic-based methods

- Heuristic-based methods use circuit graphs
- Pattern-matching
- Signal flow analysis
- Graph isomorphism

- Pros:
  - Flexible and customizable
  - Can apply to other constraints
- Cons:
  - Takes manual development efforts
  - Hard to generalize



Building block symmetry

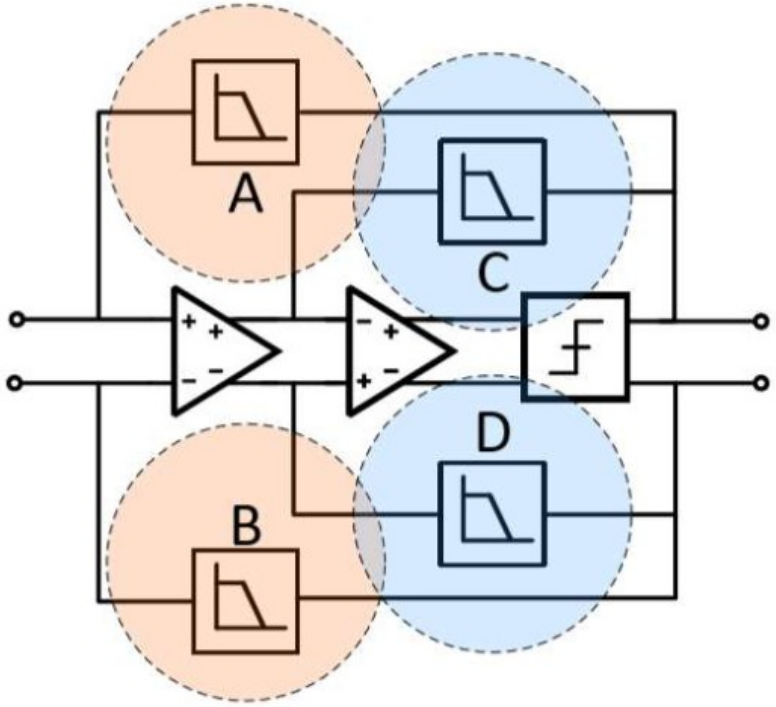
[Xu+, *2019*]

Recent Trends in Symmetry Detection

- S^{3}DET: graph symmetry for hierarchical circuit symmetry constraint [Liu+, ASP-DAC 2020]
- Accelerated Graph Similarity with Graph Neural Network [Kunal+, ICCAD 2020]
- Supervised Graph Learning [Gao+, ASP-DAC 2021]
- Graph Similarity Considering Sizing With Unsupervised Graph Learning [Chen+, DAC 2021]

[Liu+, ASP-DAC 2020]

- Symmetry ambiguity
  - Only detecting subcircuits similarities does not work well in practice
  - Designers tend to reuse building blocks if possible
  - Widely used digital standard cells create lots of issues

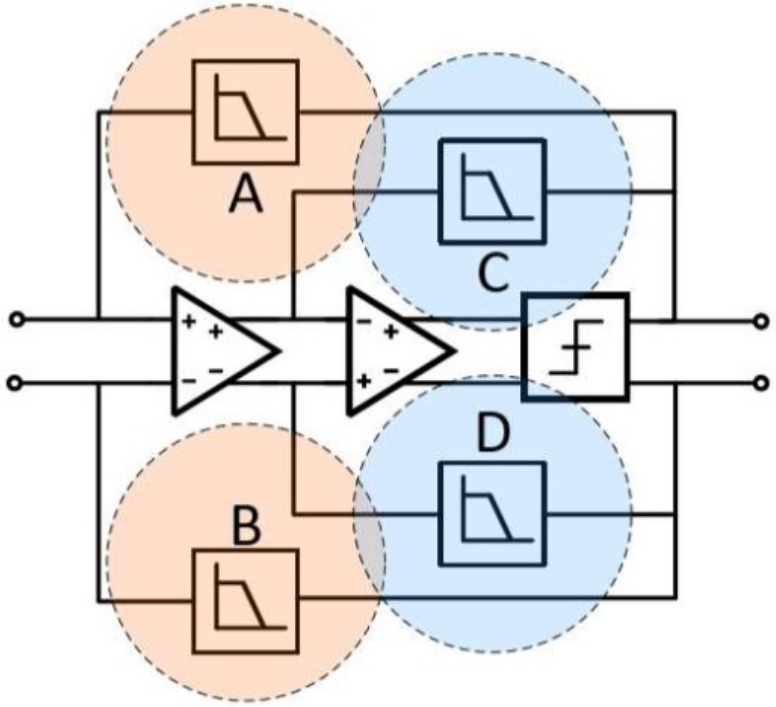


- Only (A,B) and (C,D) need matching
- Conventional methods would lead to over constraints: (A,C) and (B, D) matching create layout overhead

S³DET

[Liu+, ASP-DAC 2020]

- Resolving symmetry ambiguity
  - Extract neighboring circuit topology for each cell
  - Determine symmetry based on extracted subgraph similarity



- The neighboring circuits of A is more similar compared with B, than C
- Detect symmetry based on the “context” of the circuit system

S^{3}DET Algorithm

[Liu+, ASP-DAC 2020]

- Check each pair of subcircuits
  - 1. Extract subgraphs
  - 2. Apply K-S test on the eigenvalues of graph Laplacian matrices
  - 3. Annotate constraint if K-S test score is higher than the threshold

Algorithm 1 S^{3}DET Algorithm

Input: Hierarchical Netlist N
Output: Hierarchical Symmetry Constraint

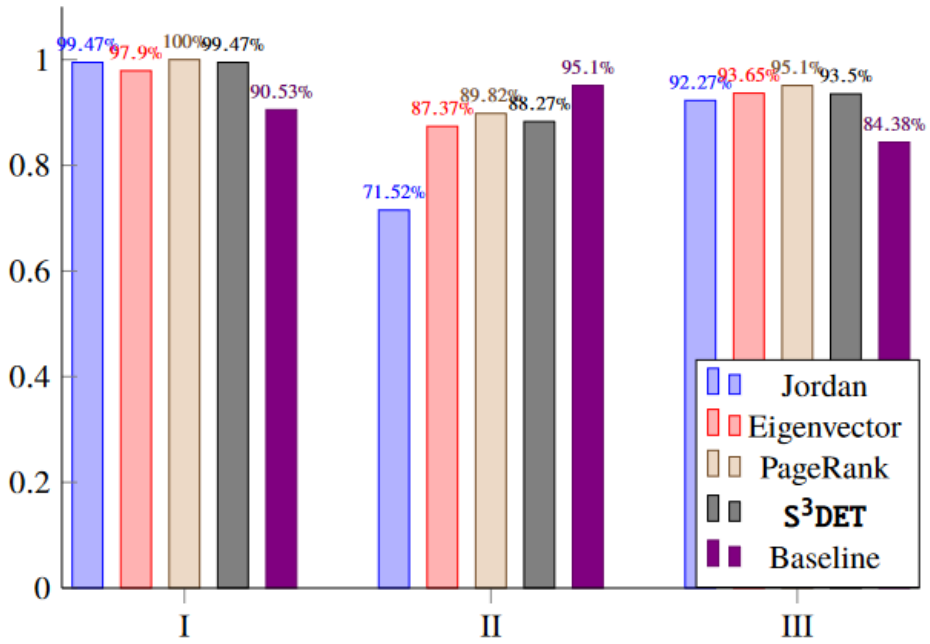
```
Generate hierarchy tree T and propagate label
Construct flattened graph representation G
Node v ← T.root
function S3DET (v)
    while v is not leaf cell do
        for valid subcircuits g1 and g2 of v do
            g1', g2' ← Extract(G, g1, g2)
            if GraphSim(g1', g2') then
                Add symmetric constraint g1, g2 to hierarchy v
        for child node c of v do
            S3DET (c)
end function
```

K-S Test: $D_n = \sup_x |F_{1,n}(x) - F_{2,m}(x)|$

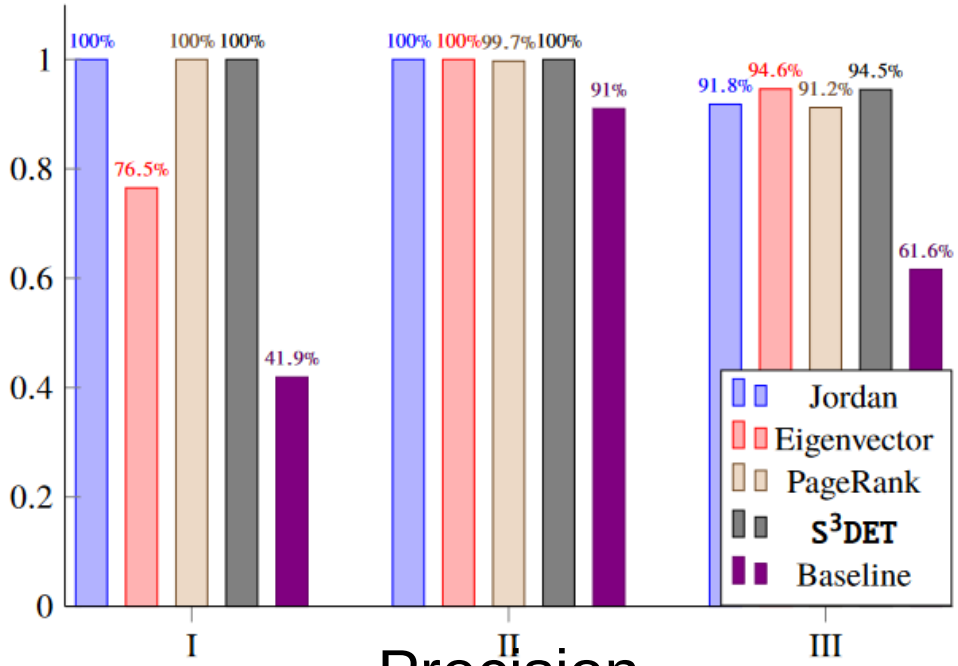
S³DET Algorithm

[Liu+, ASP-DAC 2020]

- S³DET achieve better accuracy and precision over baseline
  - Three benchmark circuits
  - Baseline is subcircuit name matching
  - Jordan, Eigenvector and PageRank are the variants of S³DET



Accuracy

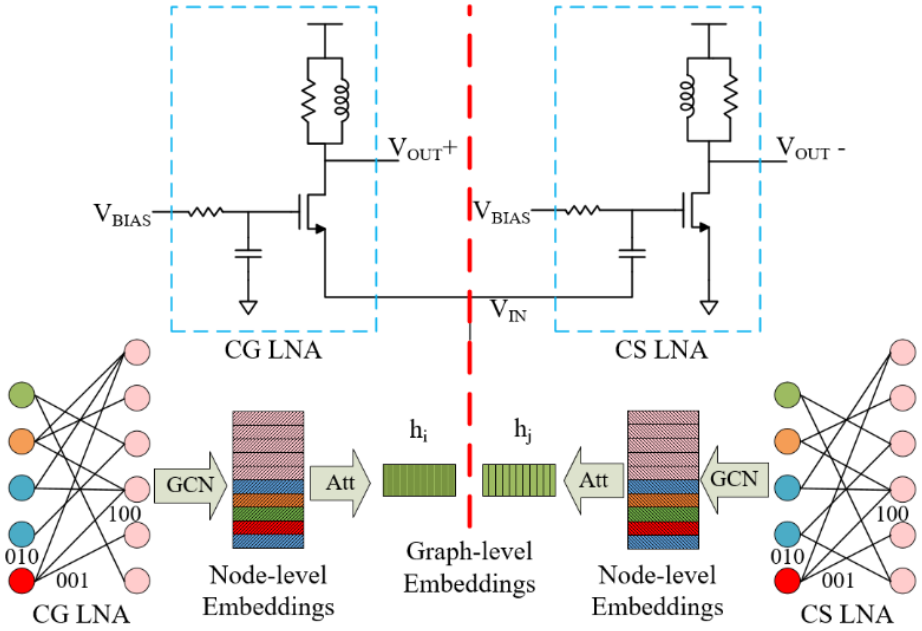


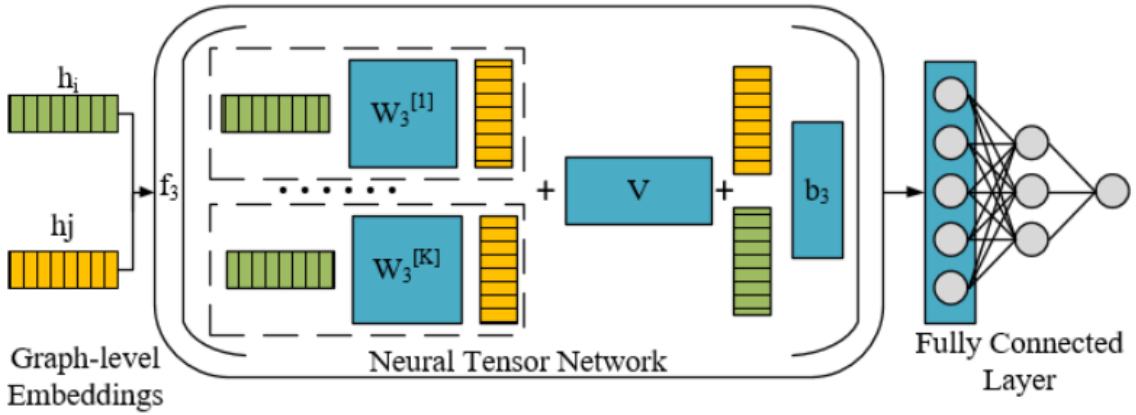
Precision

Accelerated Graph Similarity

[Kunal+, ICCAD 2020]

- Use GNN to approximate graph edit distance (GED)
- Check each pair of subcircuits
  - 1. Construct two graphs
  - 2. Predict GED between two graphs using pre-trained NN
  - 3. Annotate constraint if GED is lower than the threshold

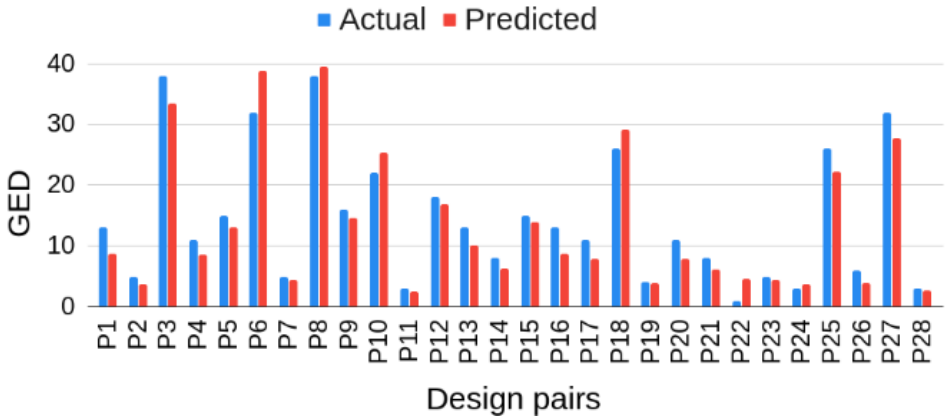




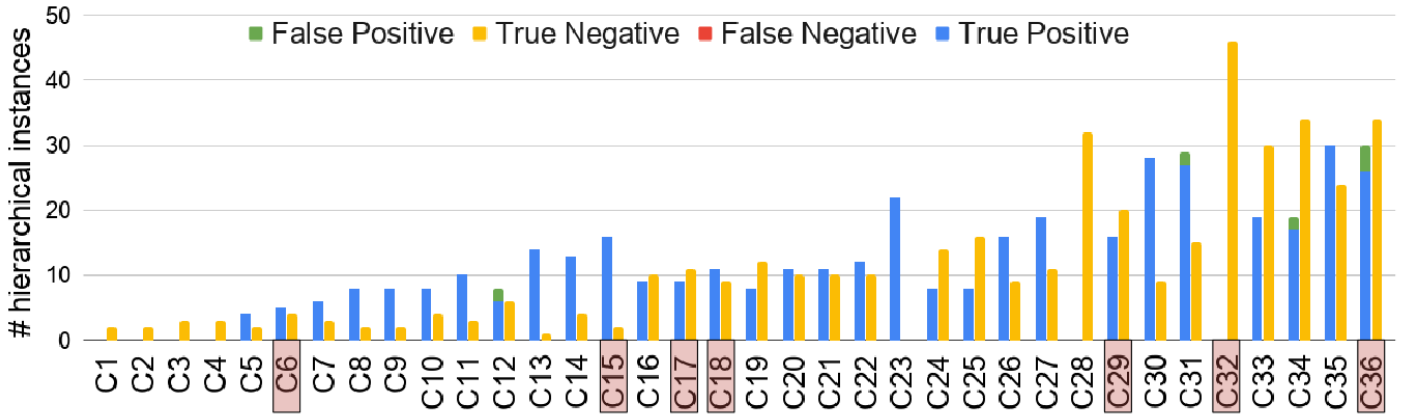
Accelerated Graph Similarity

[Kunal+, ICCAD 2020]

- Experimental results
  - Effective GED prediction
  - Effective constraint prediction



GED Prediction Accuracy

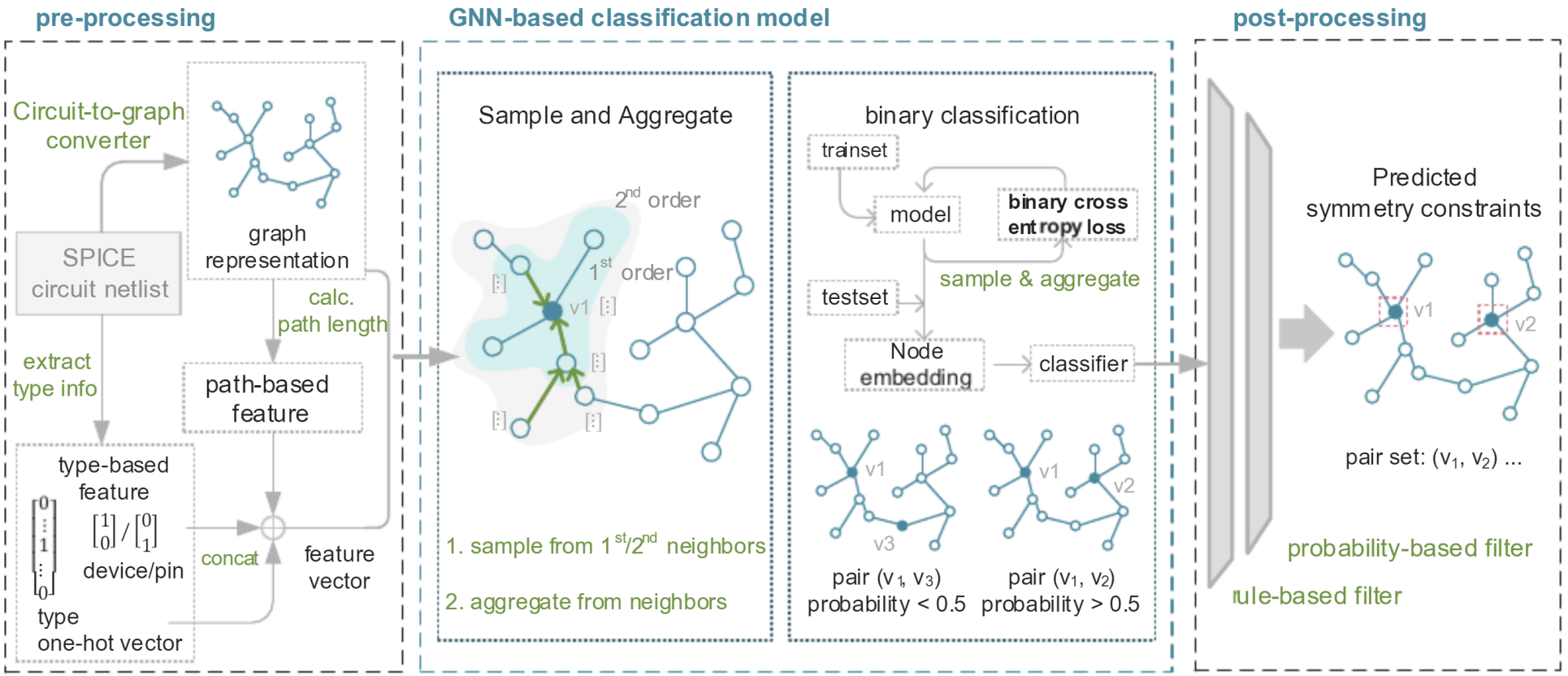


Constraint Prediction Accuracy

Supervised Graph Learning

[Gao+, ASP-DAC 2021]

- Use GNN supervised learning to detect symmetry constraints
- Applied to device-level constraints



Supervised Graph Learning

[Gao+, ASP-DAC 2021]

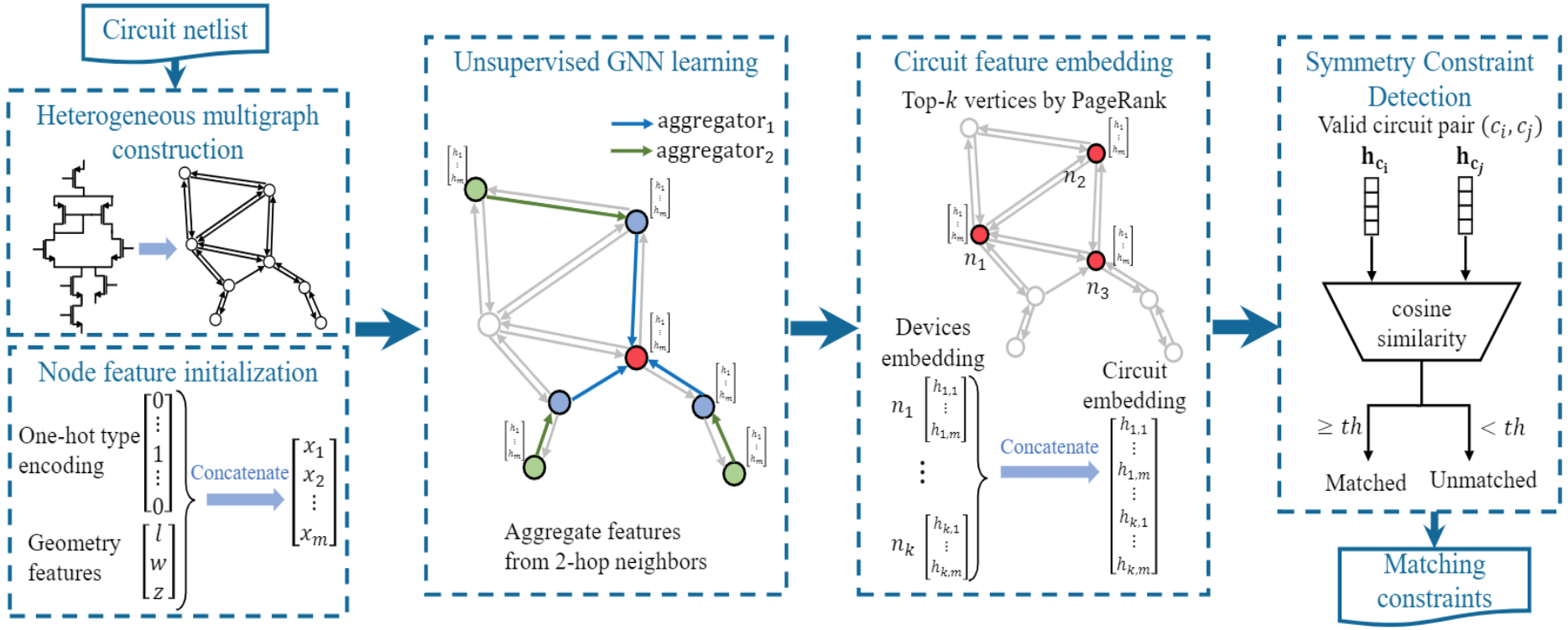
- Experimental results show better accuracy over conventional heuristic method [Xu+, ICCAD 2019]

Dataset	MIXED			OTA		
Metric	TPR	FPR	F1-score	TPR	FPR	F1-score
S^3 DET-dl	0.667	0.0722	0.485	0.556	0.0741	0.556
SFA	0.667	0.0203	0.676	0.778	**0.0185**	0.824
ours-A	**0.917**	0.0833	0.579	**1.000**	0.0556	0.857
ours-B	0.806	0.0167	0.784	0.889	**0.0185**	0.889
ours-full	**0.917**	**0.0074**	**0.904**	**1.000**	**0.0185**	**0.947**

Graph Similarity with Unsupervised GNN

[Chen+, DAC-2021]

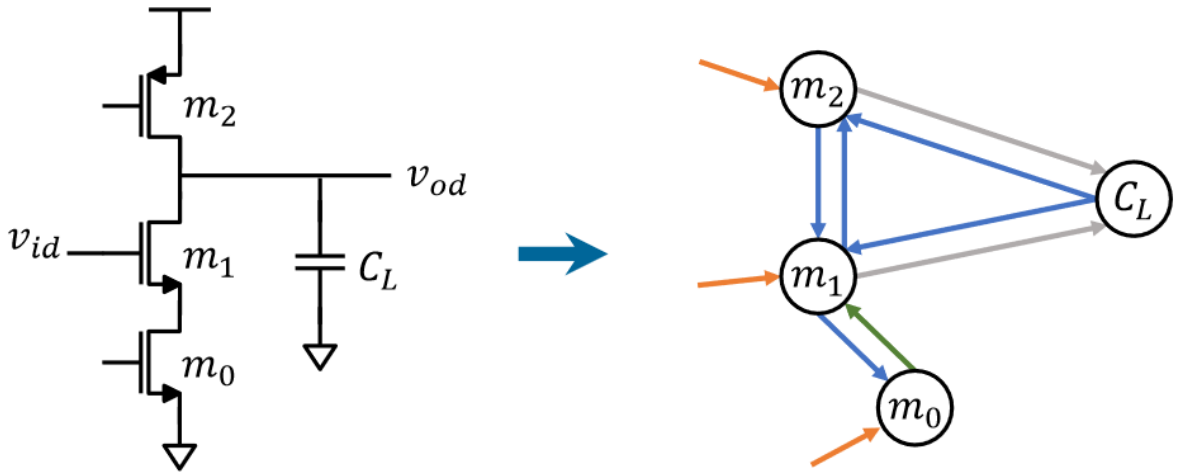
- Consider sizing parameters
- Unsupervised learning algorithm
- Universal scheme for device-level and circuit-level constraints

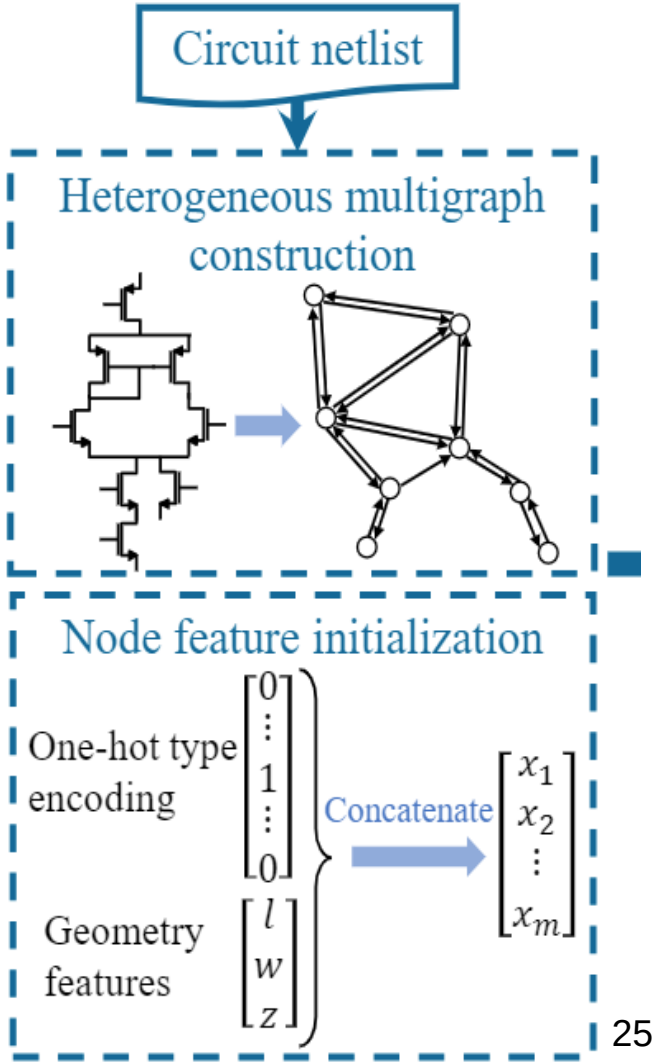


Graph Similarity with Unsupervised GNN

[Chen+, DAC-2021]

- Use transistor parameter as GNN node features
  - Device type encoding
  - The length and width of the device
  - The number of metal layers



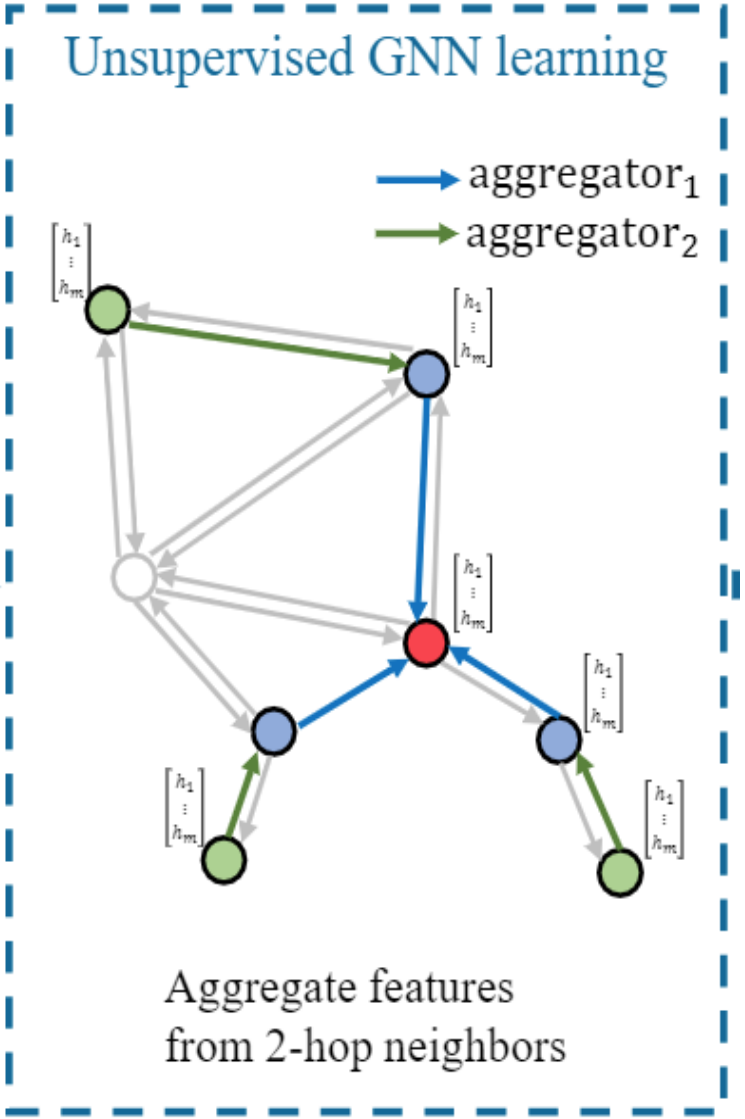


Graph Similarity with Unsupervised GNN

[Chen+, DAC-2021]

- Unsupervised learning algorithm
  - Contrastive loss to distinguish the nodes

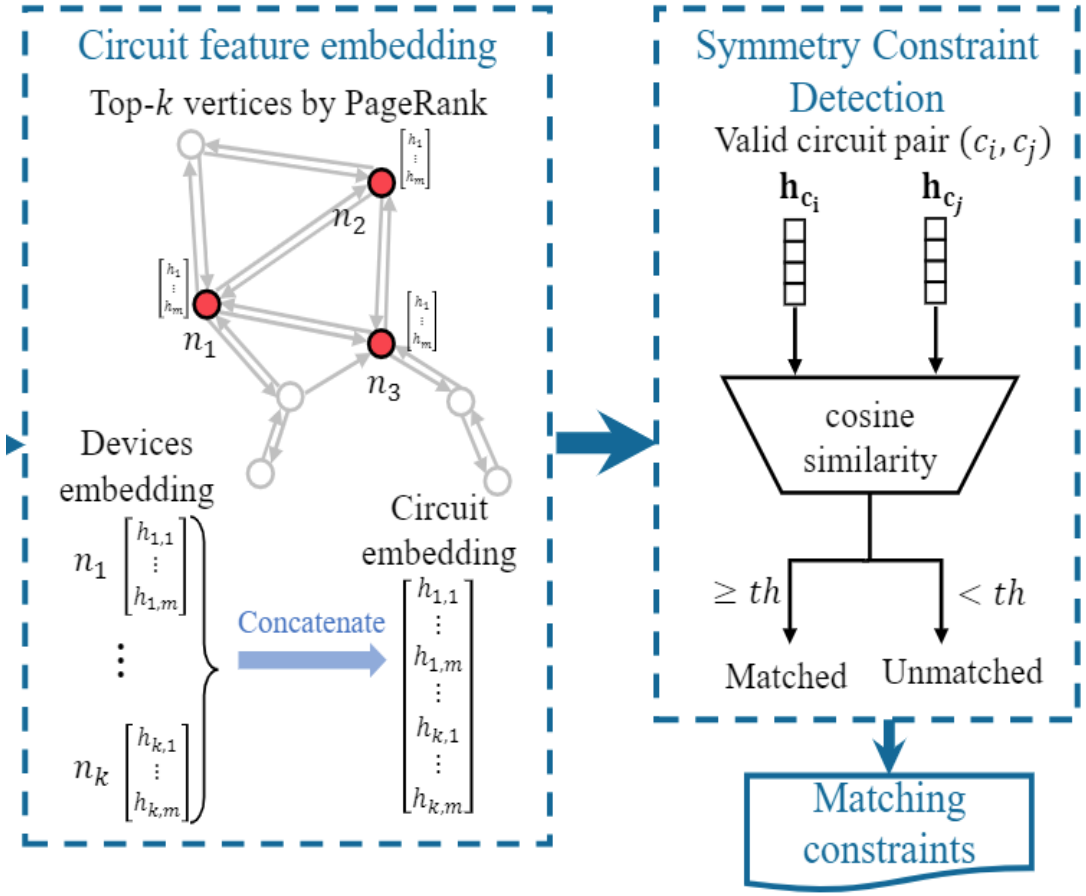
$$\mathcal{L}(z_v) = -\sum_{u \in \mathcal{N}_{in}(v)} \log(\sigma(z_u^\intercal z_v)) - \sum_{i=1}^{B} \mathbb{E}_{\tilde{u} \sim Neg(v)} \log(1 - \sigma(z_{\tilde{u}}^\intercal z_v))$$



Graph Similarity with Unsupervised GNN

[Chen+, DAC-2021]

- Universal symmetry criteria: cosine similarity
- All devices and circuits are embedded into vectors
- The vector pairs are compared by cosine similarity



Graph Similarity with Unsupervised GNN

[Chen+, DAC-2021]

- Experimental results show better accuracy
  - Over heuristic [Xu+, ICCAD 2019] on device-level detection
  - Over S^{3}DET on sub-circuit level detection

Benchmark			S^{3}DET [20]						This work					
Design	#Devices	#Nets	TPR	FPR	PPV	ACC	F_1-score	Runtime	TPR	FPR	PPV	ACC	F_1-score	Runtime†
ADC1	285	122	1.000	0.036	0.667	0.966	0.800	36.70	1.000	0.000	1.000	1.000	1.000	2.71
ADC2	345	162	1.000	0.044	0.765	0.962	0.867	30.98	1.000	0.000	1.000	1.000	1.000	2.45
ADC3	347	163	1.000	0.125	0.526	0.890	0.690	49.58	1.000	0.014	0.909	0.988	0.952	2.74
ADC4	731	372	0.619	0.000	1.000	0.812	0.765	1717.81	0.880	0.005	0.994	0.938	0.934	3.55
ADC5	1233	586	0.864	0.036	0.836	0.946	0.850	1795.52	0.835	0.015	0.920	0.958	0.875	5.14
Average	-	-	0.897	0.048	0.759	0.915	0.794	726.12	**0.943**	**0.007**	**0.965**	**0.977**	**0.952**	**3.32**

Device Level

Benchmark			SFA [6]						This work					
Design	#Devices	#Nets	TPR	FPR	PPV	ACC	F_1-score	Runtime	TPR	FPR	PPV	ACC	F_1-score	Runtime†
OTA1	12	14	0.667	0.000	1.000	0.941	0.800	<0.1	0.333	0.000	1.000	0.882	0.500	2.17
OTA2	20	20	0.875	0.171	0.333	0.833	0.483	<0.1	0.625	0.049	0.556	0.922	0.588	2.17
OTA3	12	12	0.667	0.083	0.667	0.867	0.667	<0.1	0.333	0.000	1.000	0.867	0.500	2.17
OTA4	36	36	0.667	0.131	0.170	0.861	0.271	<0.1	0.667	0.007	0.800	0.981	0.727	2.18
OTA5	38	18	0.833	0.004	0.909	0.989	0.870	<0.1	0.667	0.011	0.727	0.975	0.696	2.18
OTA6	15	9	0.571	0.000	1.000	0.870	0.727	<0.1	1.000	0.000	1.000	1.000	1.000	2.11
COMP1	47	31	1.000	0.108	0.197	0.895	0.329	<0.1	1.000	0.011	0.700	0.989	0.824	2.17
COMP2	8	16	1.000	0.000	1.000	1.000	1.000	<0.1	1.000	0.000	1.000	1.000	1.000	2.18
COMP3	34	22	0.875	0.016	0.778	0.978	0.824	<0.1	1.000	0.004	0.941	0.996	0.970	2.19
COMP4	22	16	0.625	0.057	0.455	0.921	0.526	<0.1	0.625	0.019	0.714	0.956	0.667	2.18
COMP5	17	12	1.000	0.143	0.500	0.875	0.667	<0.1	1.000	0.000	1.000	1.000	1.000	2.17
COMP6	17	12	1.000	0.000	1.000	1.000	1.000	<0.1	1.000	0.000	1.000	1.000	1.000	2.17
DAC1	10	11	1.000	0.000	1.000	1.000	1.000	<0.1	1.000	0.000	1.000	1.000	1.000	2.17
DAC2	12	19	1.000	0.000	1.000	1.000	1.000	<0.1	1.000	0.000	1.000	1.000	1.000	2.17
LATCH1	24	14	0.800	0.074	0.471	0.917	0.593	<0.1	0.600	0.000	1.000	0.970	0.750	2.18
Average	-	-	**0.839**	0.052	0.699	0.930	0.717	**<0.1**	0.790	**0.007**	**0.896**	**0.969**	**0.815**	2.17

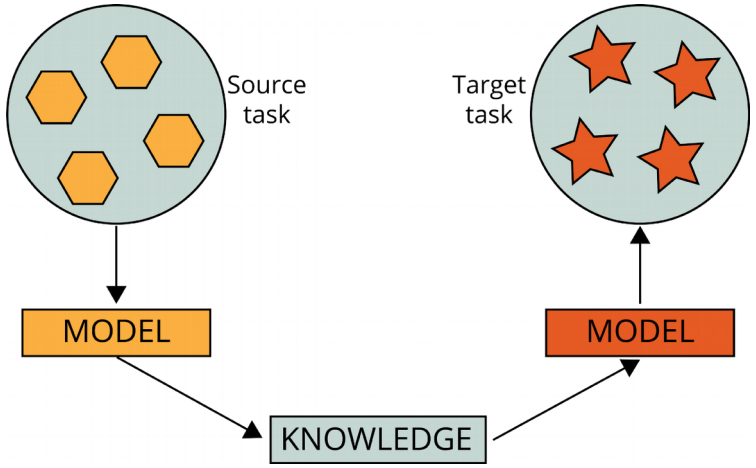
Subcircuit Level

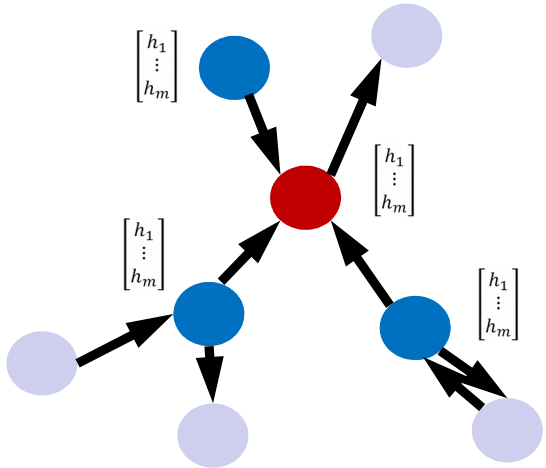
What’s Left?

- Existing work has demonstrated effectiveness on symmetry detection task
  - Algorithms specifically designed for matching constraints
  - Very few demonstration on other constraint types
- How to achieve a generalized way to other analog constraints?

Supervised Analog Constraint Extraction

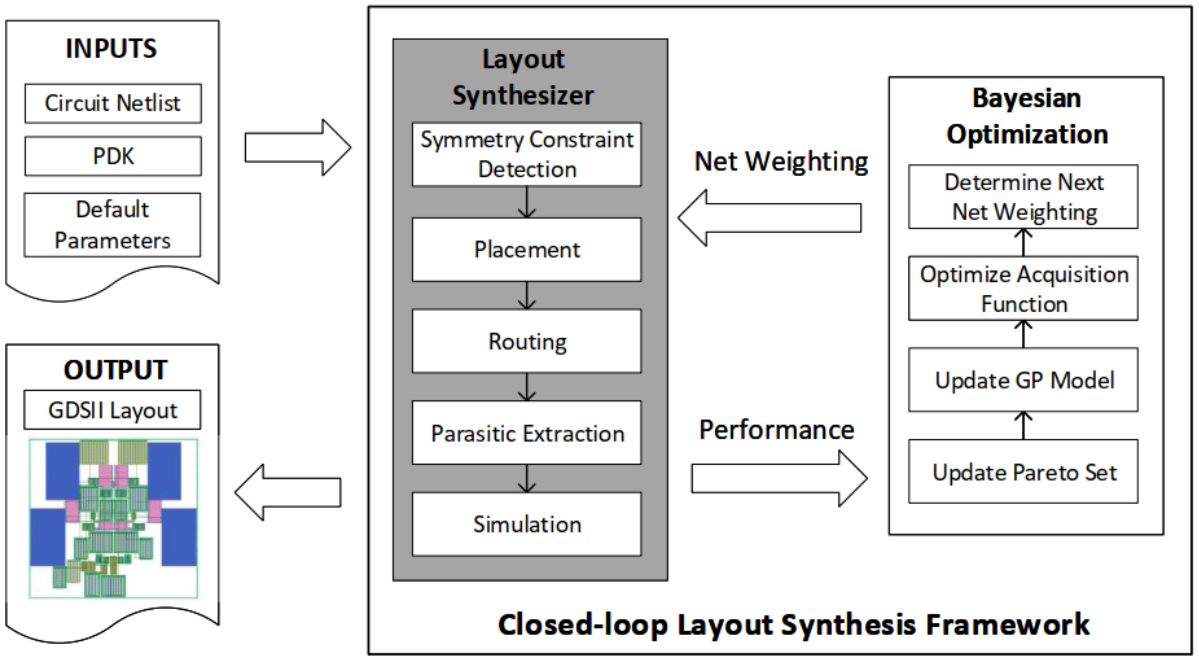
- We can use **supervised learning**
  - Apply to all types by labeling different constraints
  - Similar to [Gao+ ASP-DAC 2021] ideology
- Challenge 1: Data
  - Opportunity 1: Few-shot or zero-shot learning
  - Opportunity 2: General pre-trained circuit representation model
- Challenge 2: Limit of GNN model
  - Opportunity: More topology-aware GNN architectures





Closed-Loop Analog Constraint Extraction

- We can **integrate** analog constraint extraction with P&R tools
- Use layout results as target
  - Using feedbacks from layout to improve constraints
  - Do not require labeled data and possible to learn from the exploration



[Liu+, *2020*]

Conclusion

- Overview of the analog constraint extraction
- Overview of conventional approaches
- Presentation of recent research trends
- Discussion of challenges and opportunities