MacroPlacement and SpecPart Repositories in the TILOS AI Institute

Andrew Kahng, UC San Diego

https://tilos.ai

Closing the Gap Between Research and Practice

- **“Data, Benchmarking and Roadmapping”** is a goal of the TILOS AI institute (https://tilos.ai)
- Change **how research is conducted and *translated into practice*** in high-stakes applied optimization domains
  - **E.g., IC design automation**

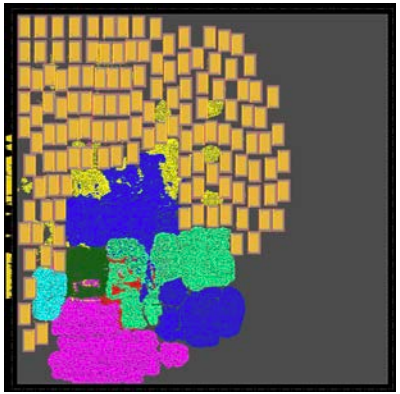
- Enable research and discovery at the leading edge
  - Less barriers
  - Culture of transparency and reproducibility
  - **Velocity of innovation and improvement**

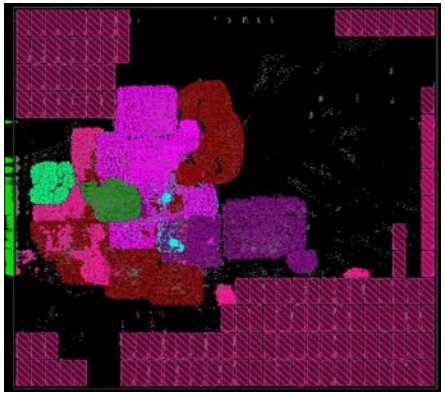
What is MacroPlacement? https://github.com/TILOS-AI-Institute/MacroPlacement

- **Open, transparent effort** to provide a public, baseline implementation of Google Brain's Circuit Training (Morpheus) deep RL-based placement
  - **With thanks to the Google Brain team for very helpful Q&A, discussions**
- **Testcases**
  - Fully-formed and relevant: Ariane, MemPool, NVDLA
- **Enablements**
  - Accessible to all: NanGate45, ASAP7, SKY130HD + "FakeRAM", "FakeStack"
- **Flows**
  - Tool setups and runscripts – **both proprietary and open-source**
  - **Includes Cadence tool scripts**: reproducible results under recent policy change
- **Code Elements**
  - Implementations of missing/binarized pieces, format translators, etc.
- **Solutions**
  - Post-routing results from three SP&R flows

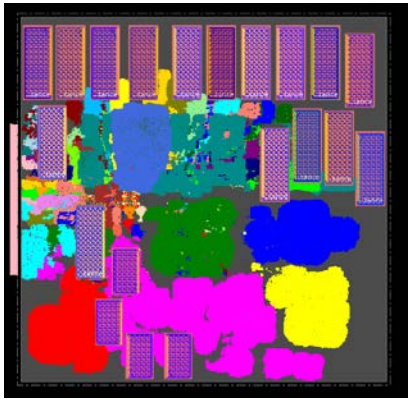
First Permitted Sharing of Commercial EDA Scripts in GitHub

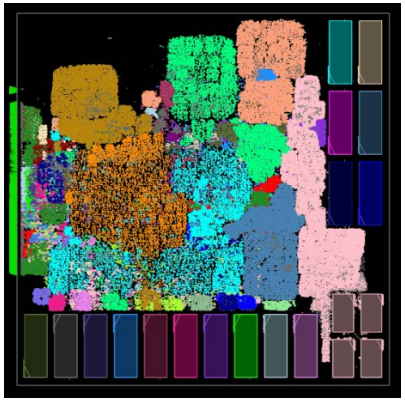
- **"We thank Cadence for granting permission to share our research to help promote and foster the next generation of innovators."**
  - ***Kudos and Thanks to David Junkin and team at Cadence !!!***
- Flow-1: Genus + Innovus
- Flow-2: Genus iSpatial + Innovus
- Flow-3: Yosys + OpenROAD
- Flow-4: Circuit Training with Genus iSpatial netlist





Ariane133 on NanGate45.
Left: Flow-1. Right: Flow-3.





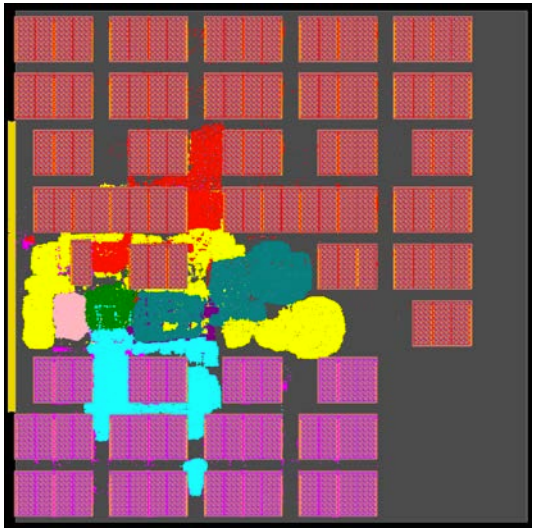
MemPool Tile on NanGate45.
Left: Flow-1. Right: Flow-3.

Observation: Human-Expert Baselines are Possible!

- **Step 1.** Choose **open** design in **open** enablement
- **Step 2.** Run **physical synthesis *(Genus iSpatial, reproducible!)***
- **Step 3.** Run gridding (+ grouping, netlist clustering) per Circuit Training methods

- **Step 4A. Gridded Human Baseline:** Human expert performs macro placement with gridding constraint
- **Step 4B,C. Circuit Training, MacroPlacement solutions:** All inputs available, apples-to-apples

- **Step 5.** Finish PD flow ***(Innovus, reproducible!)*** including Place, CTS, Route, Optimization → report final Routed WL, WNS/TNS, Power



Ariane-133, NG45 Gridded Human Baseline

Stage in Physical Design	Core Area (um^2)	Standard Cell Area (um^2)	Total Power (mW)	Wirelength (m)	WNS (ps)	TNS (ns)
Post-placement	2560080	215189	0.29	4.47	-2	-0.05
Post-CTS	2560080	216326	0.30	4.47	1	0
Post-Routing	2560080	216326	0.30	4.59	62	0

Code Elements: Fill in Undocumented* or Binarized Pieces

- **Gridding**
  - Determines dissection of layout canvas into rows and columns of gridcells
- **Grouping**
  - Ties closely-related logic to hard macros and clumps of (placed) IOs
- **Hypergraph clustering**
  - Uses hMETIS and some initial coordinate information to form soft macros
- **Force-directed placement**
  - Places the center of each soft macro onto a gridcell center
- **Simulated annealing**
  - Places the center of each hard macro onto a gridcell center
- **Format translators**
  - Converts from LEF/DEF or Bookshelf to Protocol Buffer, .plc format

- **Collaborators and Thanks:**
  - Sergio Guadarrama, Anna Goldie, Azalia Mirhoseini, Eric Johnson, Ed Chi + others **(Google)**
  - Zhiang Wang, Sayak Kundu, Ravi Varadarajan, Yucheng Wang, C.-K. Cheng **(UCSD)**

What is HypergraphPartitioning?

https://github.com/TILOS-AI-Institute/HypergraphPartitioning

- Purpose: Maintain and advance the SOTA for **hypergraph partitioning**

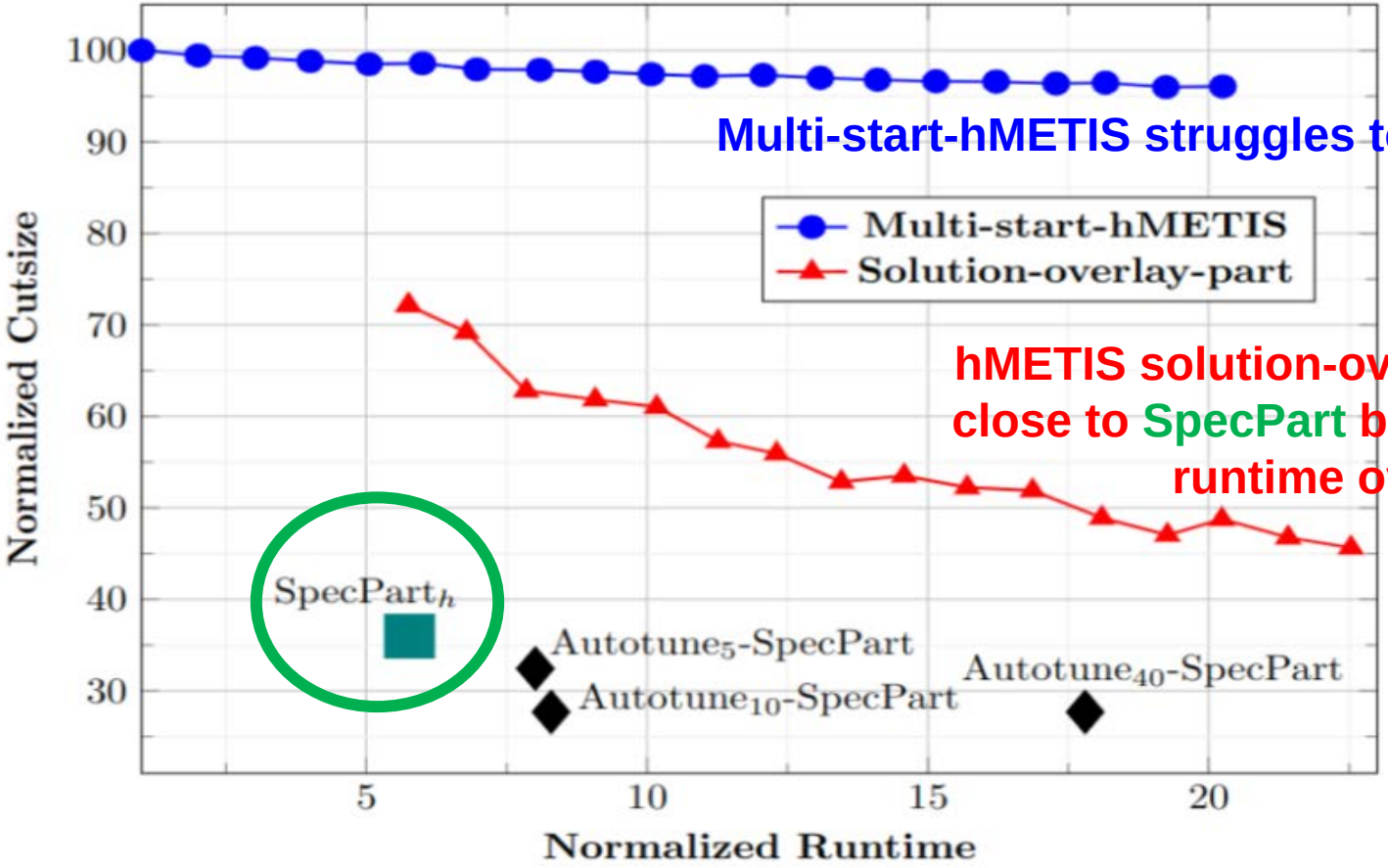
- This means:
  - Golden benchmark testcases, format converters
  - Golden solution evaluators
  - World's best-ever solutions ("leaderboard")
  - World's best-ever optimizers (in open source)

Current Leaderboard of minimum hyperedge cut values on Titan23 testcases with different imbalance factors (ε):

	Statistics		Cutsize				
Testcase	#Vertices	#Hyperedges	ε = 1	ε = 2	ε = 5	ε = 10	ε = 20
sparcT1_core	91976	92827	1088	1012	1001	1090	903
neuron	92290	125305	239	239	239	239	206
stereovision	94050	127085	186	180	180	91	91
des90	111221	139557	416	402	393	393	358
SLAM_spheric	113115	142408	1061	1061	1061	1061	1061
cholesky_mc	113250	144948	343	285	285	285	285
segmentation	138295	179051	165	126	125	125	78
bitonic_mesh	192064	235328	588	587	581	543	483

- **Collaborators and Thanks:** Ismail Bustany (AMD), Yiannis Koutis (NJIT), Bodhisatta Pramanik (Iowa State), Zhiang Wang (UCSD)

Cutsize vs. Runtime



Multi-start-hMETIS struggles to match SpecPart

hMETIS solution-overlay-part can get close to SpecPart but with significant runtime overhead

Cutsize Results

- **New best-known solutions** for almost every benchmark
- Cutsize reductions of ~70% on gsm_switch, ~25% on sparcT1_core

Testcase	ε	Best Public	Spec Part
IBM04	10	542	388
IBM06	10	885	733
IBM13	10	832	693
IBM15	2	2833	2741
IBM16	2	2059	1951
IBM17	2	2403	2354

ISPD98 Benchmarks without weight[1]

Testcase	ε	Best Public	Spec Part
IBM03.w	10	681	541
IBM07.w	10	737	634
IBM10.w	10	756	443
IBM12.w	2	1998	1965
IBM15.w	2	2099	1904
IBM17.w	2	2353	2270

ISPD98 Benchmarks with weight[1]

Testcase	ε	hMetis cutsize	Spec Part
denoise	20	478	224
gsm_switch	**20**	**5352**	**1407**
dart	2	844	807
LU230	2	3328	3273
sparcT1_chip2	**2**	**1198**	**899**
bitcoin_miner	2	1489	1297

Titan23 Benchmarks[2]

[1] C. J. Alpert, "The ISPD98 circuit benchmark suite", *Proc. ACM/IEEE International Symposium on Physical Design (ISPD)*, 1998.
[2] K. E. Murray, S. Whitty, S.Liu, J.Luu and V.Betz, "Titan: Enabling large and complex benchmarks in academic CAD", *Proc. Intl. Conf. on Field Programmable Logic and Applications*, 2013.

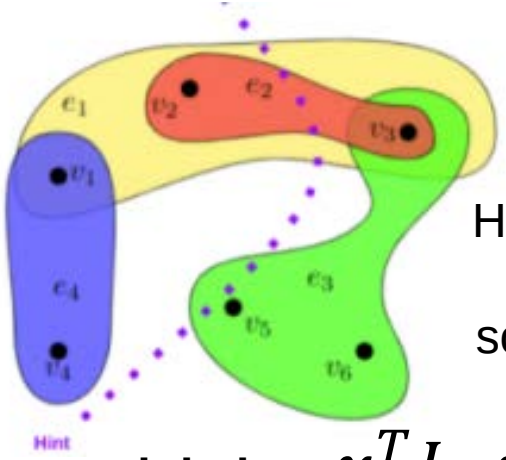
BACKUP

SpecPart: *Supervised* Spectral Partitioning

- Apply an initial good solution as a “hint” to *supervise* partitioning process
- Extend the eigenvalue problem for minimizing cutsize to a *generalized eigenvalue problem* for better balanced and supervised partitions

- Other key ideas
  - *Partition low-stretch spanning trees* based on eigenvector embedding
  - *Overlay* multiple solutions to obtain a clustering of the hypergraph
  - *Optimally partition* the clustered hypergraph
  - *Autotune* existing partitioners to obtain a strong “hint”

SpecPart: *Supervised* Spectral Partitioning

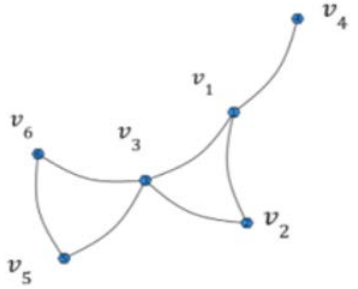


Hypergraph with initial good solution = **"hint"**

minimize $x^T L_G x$ **(cutsize)**

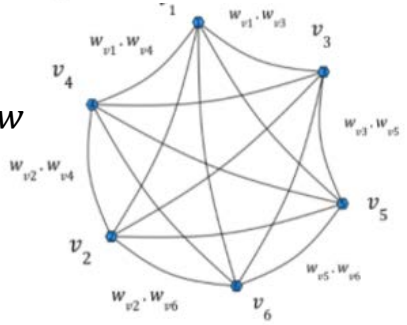
Clique expansion graph G

minimize $x^T L_G x$



Weight-balance graph G_w

maximize $x^T L_{G_w} x$

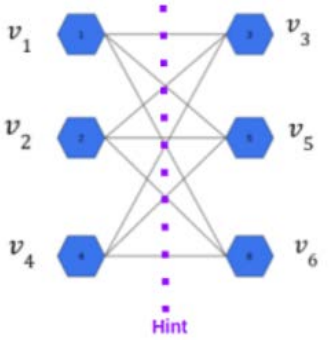


Generalized Eigenvalue Problem

minimize $\dfrac{x^T L_G x}{x^T L_{G_w} x + x^T L_{G_h} x}$

Hint graph G_h

maximize $x^T L_{G_h} x$



What is HypergraphPartitioning? https://github.com/TILOS-AI-Institute/MacroPlacement

https://github.com/TILOS-AI-Institute/HypergraphPartitioning

Benchmarks

- ISPD98 benchmarks with unit vertex weights
- ISPD98 benchmarks with actual vertex weights
- Titan23 benchmarks

Partition solution files

- Solutions to ISPD98 benchmarks with unit vertex weights for different imbalance factors
- Solutions to ISPD98 benchmarks with actual vertex weights for different imbalance factors
- Solutions to Titan23 benchmarks for different imbalance factors

Golden solution evaluator script (.py)

SpecPart codebase (.jl)

Leaderboard of mincuts

- Mincut leaderboard on ISPD98 benchmark with unit vertex weights
- Mincut leaderboard ISPD98 benchmark with actual vertex weights
- Mincut leaderboard Titan23 benchmarks