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-Al-Institute/MacroPlacement](https://github.com/TILOS-Al-Institute/MacroPlacement)

- **Open, transparent effort** to provide a public, baseline implementation of Google Brain’s [Circuit Training](https://github.com/TILOS-Al-Institute/MacroPlacement) (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
• “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
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

<table>
<thead>
<tr>
<th>Stage in Physical Design</th>
<th>Core Area (um²)</th>
<th>Standard Cell Area (um²)</th>
<th>Total Power (mW)</th>
<th>Wirelength (m)</th>
<th>WNS (ps)</th>
<th>TNS (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Post-placement</td>
<td>25600080</td>
<td>215189</td>
<td>0.29</td>
<td>4.47</td>
<td>-2</td>
<td>-0.05</td>
</tr>
<tr>
<td>Post-CTS</td>
<td>25600080</td>
<td>216326</td>
<td>0.30</td>
<td>4.47</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Post-Routing</td>
<td>25600080</td>
<td>216326</td>
<td>0.30</td>
<td>4.59</td>
<td>62</td>
<td>0</td>
</tr>
</tbody>
</table>
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 Hypergraph Partitioning?

- **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)

- **Collaborators and Thanks:** Ismail Bustany (AMD), Yiannis Koutis (NJIT), Bodhisatta Pramanik (Iowa State), Zhiang Wang (UCSD)
Multi-start-hMETIS struggles to match SpecPart

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

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

<table>
<thead>
<tr>
<th>Testcase</th>
<th>$\varepsilon$</th>
<th>Best Public</th>
<th>Spec Part</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM04</td>
<td>10</td>
<td>542</td>
<td>388</td>
</tr>
<tr>
<td>IBM06</td>
<td>10</td>
<td>885</td>
<td>733</td>
</tr>
<tr>
<td>IBM13</td>
<td>10</td>
<td>832</td>
<td>693</td>
</tr>
<tr>
<td>IBM15</td>
<td>2</td>
<td>2833</td>
<td>2741</td>
</tr>
<tr>
<td>IBM16</td>
<td>2</td>
<td>2059</td>
<td>1951</td>
</tr>
<tr>
<td>IBM17</td>
<td>2</td>
<td>2403</td>
<td>2354</td>
</tr>
</tbody>
</table>

**ISPD98 Benchmarks without weight[1]**

<table>
<thead>
<tr>
<th>Testcase</th>
<th>$\varepsilon$</th>
<th>Best Public</th>
<th>Spec Part</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM03.w</td>
<td>10</td>
<td>681</td>
<td>541</td>
</tr>
<tr>
<td>IBM07.w</td>
<td>10</td>
<td>737</td>
<td>634</td>
</tr>
<tr>
<td>IBM10.w</td>
<td>10</td>
<td>756</td>
<td>443</td>
</tr>
<tr>
<td>IBM12.w</td>
<td>2</td>
<td>1998</td>
<td>1965</td>
</tr>
<tr>
<td>IBM15.w</td>
<td>2</td>
<td>2099</td>
<td>1904</td>
</tr>
<tr>
<td>IBM17.w</td>
<td>2</td>
<td>2353</td>
<td>2270</td>
</tr>
</tbody>
</table>

**ISPD98 Benchmarks with weight[1]**

<table>
<thead>
<tr>
<th>Testcase</th>
<th>$\varepsilon$</th>
<th>hMetis cutsize</th>
<th>Spec Part</th>
</tr>
</thead>
<tbody>
<tr>
<td>denoise</td>
<td>20</td>
<td>478</td>
<td>224</td>
</tr>
<tr>
<td>gsm_switch</td>
<td>20</td>
<td>5352</td>
<td>1407</td>
</tr>
<tr>
<td>dart</td>
<td>2</td>
<td>844</td>
<td>807</td>
</tr>
<tr>
<td>LU230</td>
<td>2</td>
<td>3328</td>
<td>3273</td>
</tr>
<tr>
<td>sparcT1_chip2</td>
<td>2</td>
<td>1198</td>
<td>899</td>
</tr>
<tr>
<td>bitcoin_miner</td>
<td>2</td>
<td>1489</td>
<td>1297</td>
</tr>
</tbody>
</table>

**Titan23 Benchmarks[2]**

---


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”

\[
\text{minimize } x^T L_G x \quad (\text{cutsize})
\]

Clique expansion graph \( G \)

\[
\text{minimize } x^T L_G x
\]

Weight-balance graph \( G_w \)

\[
\text{maximize } x^T L_{G_w} x
\]

Hint graph \( G_h \)

\[
\text{maximize } x^T L_{G_h} x
\]
What is HypergraphPartitioning? [https://github.com/TILOS-AI-Institute/MacroPlacement]

**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

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