Assessment of Reinforcement Learning for Macro Placement

Chung-Kuan Cheng[1], Andrew B. Kahng[1][2], Sayak Kundu[2], Yucheng Wang[1] and Zhiang Wang[2]

[1]CSE and [2]ECE Departments, UC San Diego

GitHub: https://github.com/TILOS-AI-Institute/MacroPlacement

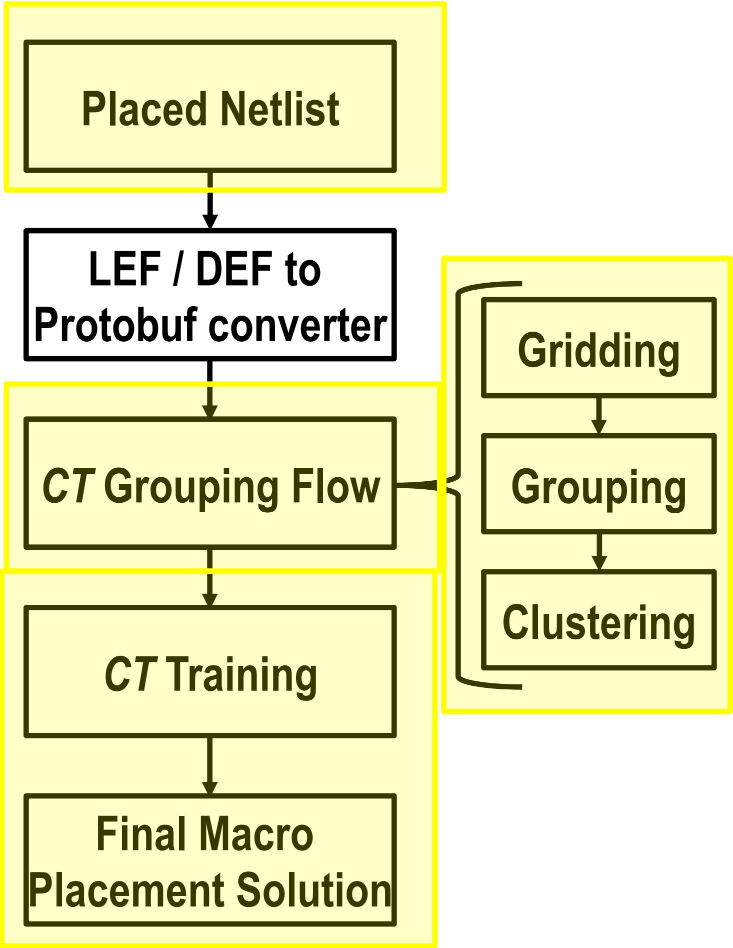
UCSD

Why *MacroPlacement* ?

- **June 2021:** Google Brain *Nature* paper proposed RL-based macro placer
  - Claimed superior or comparable macro placement solutions compared to human experts, in under six hours
  - Did not release code or data as had been committed
- **January 2022:** Google Research open-sourced Circuit Training (*CT*)
  - "reproduces the methodology published in the *Nature* 2021 paper"
  - No dataset, and insufficient code to reproduce the *Nature* results
- **March 28, 2022:** "Stronger Baselines" manuscript posted during ISPD-22
  - States shortcomings of the Nature paper
  - Also lacks code and dataset to reproduce results or confirm claims
- ***MacroPlacement*: open, transparent assessment of *Nature, CT***
  - **Goal: resolve controversy; foster calm discussion and scientific progress**
  - **June 2022:** ***MacroPlacement*** repository open-sourced

UCSD

Circuit Training (*CT*): Important Surprises and Gaps



- **Surprise: *CT* uses placement information from its input**
  - *Nature* paper does not mention this
  - Ablation: use of placement info **reduces routed wirelength by 7-10%**
- ***CT* optimizes *proxy cost* during training**
  - RL agent places hard macros on gridded canvas
  - Stdcell clusters placed using force-directed (FD) placer
  - Proxy cost (R) is then evaluated:

$$R = Wirelength + \gamma \times Density + \lambda \times Congestion$$

- **Gap: key elements hidden behind plc_client APIs**
  - FD placer, proxy cost calculation
  - We reverse-engineered these and released as open source
- **Thanks** to the **TensorFlow Agents** team for open-sourcing the grouping and training flows in the CT repository, and to **Google Brain** engineers for extensive Q&A and checking !

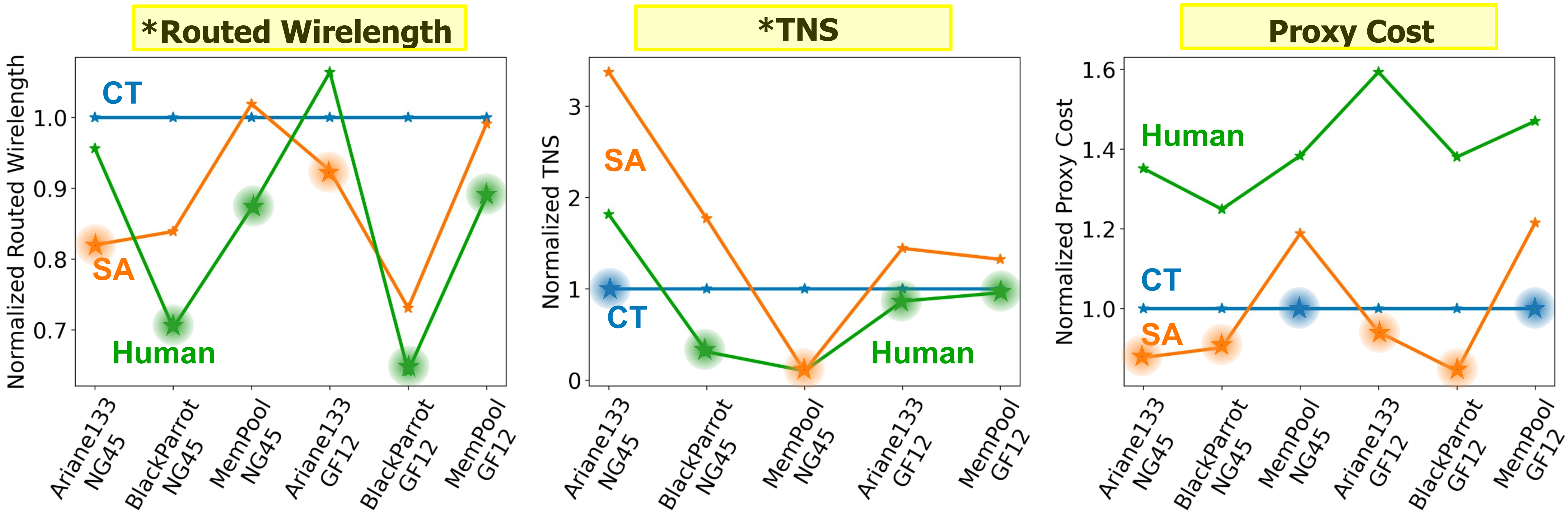
Scope of *MacroPlacement*

- **Code**: Includes open-source implementation of missing, blackbox elements of CT
  - **Missing elements**: Format translators, Baselines for comparison (simulated annealing, human expert)
  - **Blackbox elements**: force-directed placement, proxy cost components
- **Modern benchmarks**: Open testcases on open enablements
  - **Testcases (#macros, #insts): Ariane** (133, 117K), **BlackParrot** (220, 769K), **MemPool Group** (324, 2729K), **NVDLA** (128, 156K)
  - **Enablements: SKY130HD**+FakeStack, **NanGate45**, **ASAP7**+FakeRAM
- **Reproducible results with commercial synthesis, place and route evaluation flow**
  - Cadence Genus iSpatial physical synthesis flow and Cadence Innovus P&R flow
  - Synopsys Design Compiler Topographical physical synthesis flow
  - **Major changes in EDA vendor policies allow us to share our Tcl scripts in GitHub for research purposes!** ← ***Kudos and thanks to Cadence and Synopsys !!!***
- ***Extensive documentation and data: See FAQs, Docs, "Our Progress"***

*Simulated annealing is implemented following the description in the "Stronger Baselines" manuscript

UCSD

CT, SA and Human results for Modern Benchmarks



*postRouteOpt metrics

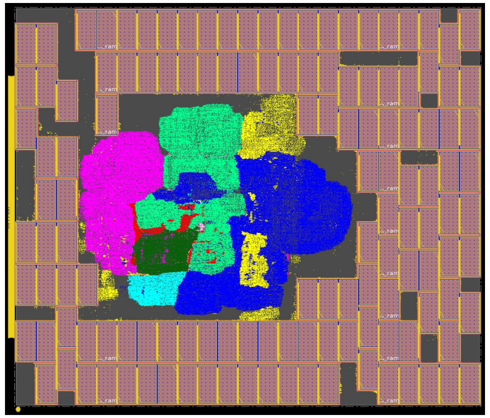
- Normalized routed wirelength, proxy cost, and TNS → **Lower values are better**
- All data is normalized to *CT* results
- More details: **Table 1** of our paper

For postRouteOpt metrics, Human outperforms CT for macro-heavy BlackParrot, MemPool Group

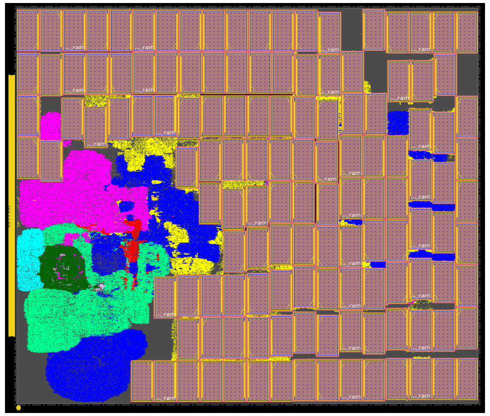
Macro Placement Solutions: Ariane133

- *__Ariane133__ with **68%** utilization, **1.3ns** target clock period on **NG45**

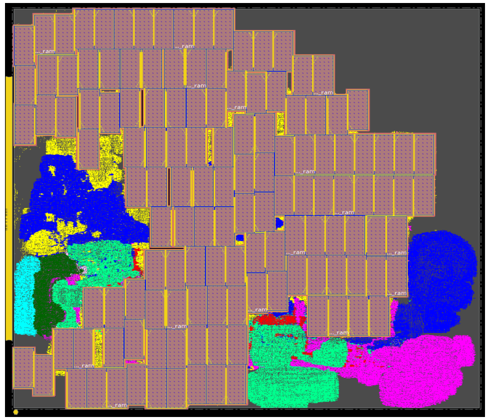
CT



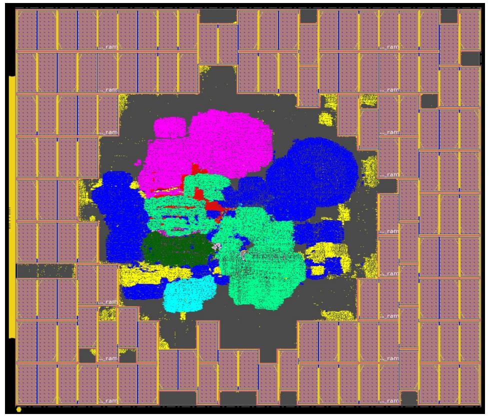
CMP



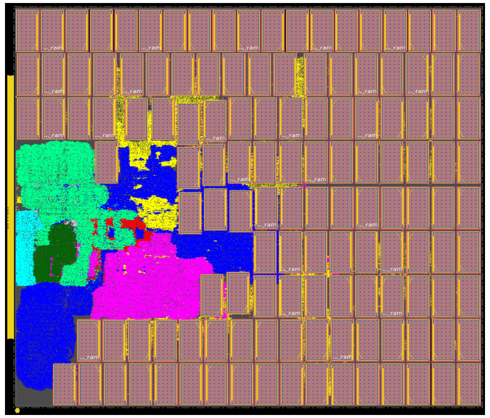
RePlAce



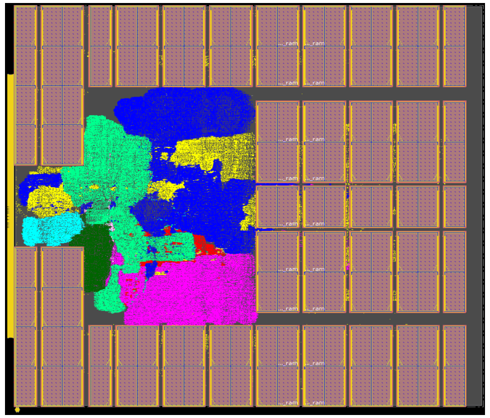
SA



AutoDMP



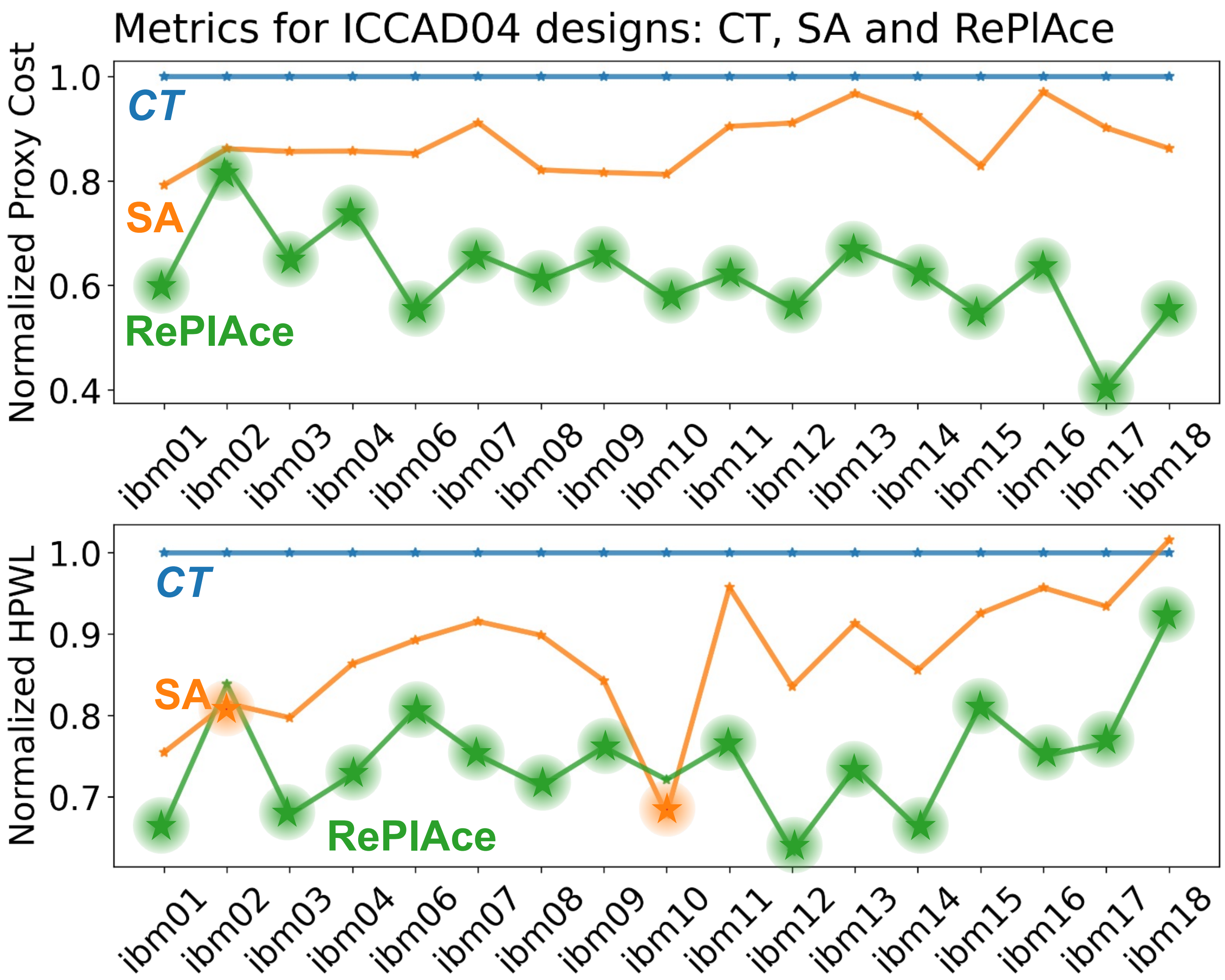
Human



- **CMP:** Innovus Concurrent Macro Placer
- **AutoDMP:** DREAMPlace based macro placer from **Nvidia Research**
- The **human** macro placement is from **Dr. Jinwook Jung** of IBM Research
- postRouteOpt metrics in Table 1 of our paper

**Nature* implements Ariane133 on a different enablement

CT, SA and RePlAce Comparison on ICCAD04 Benchmarks



Proxy Cost: RePlAce beats SA, and SA beats *CT*

- Data is **normalized** based on *CT*
- Table 6 of our paper gives raw data

HPWL: RePlAce beats SA and SA beats *CT* in most of the testcases

In terms of proxy cost, SA outperforms CT and in terms of HPWL, SA produces better result than CT in 16 out of 17 cases

Conclusions

- ***MacroPlacement* = open, transparent assessment and implementation**
  - **Source code for *CT*'s missing and blackbox elements**
  - **Modern, macro-heavy testcases** on open enablements
  - **Commercial evaluation flow** with all runscripts
  - **Baselines cited in *Nature***: human expert, simulated annealing
  - **Extensive documentation**: FAQs, "Our Progress", and more
- ***CT* benefits from placement information in incoming physical synthesis netlist**
- **Baselines (SA and Human experts) outperform *CT***
  - For 17 ICCAD04 designs and 4 out of 6 modern testcases, SA generates better proxy cost than *CT*
  - For large macro-heavy designs, human experts outperform *CT* in terms of *Nature* Table 1 metrics
- **"There is no substitute for source code (and data)"**

Please see the long video and our FAQs in GitHub for more information !!!
https://github.com/TILOS-AI-Institute/MacroPlacement

FAQ: Runtimes (Wall Times) of Different Macro Placers

Design	*CT* (Hours)	CMP (Hours)	RePlAce (Hours)	SA (Hours)	AutoDMP (Hours)
Ariane-NG45	32.31	0.05	0.06	12.50	0.29
BlackParrot-NG45	50.51	0.33	2.52	12.50	0.71
MemPool-NG45	81.23	1.97	*N.A.	12.50	1.73

- *CT*: only includes *CT* training time
- SA: stopped after 12.5 hours automatically
- CMP: only the runtime of ***place_design -concurrent_macros*** command
- Resources used:
  - *CT*: Training and evaluation jobs run on (8 NVIDIA-V100 GPU, 96 CPU thread, Memory: 354 GB) machine and 13 collector jobs on each of two (96 CPU thread, Memory: 354 GB) machines
  - SA: 320 parallel jobs where each job used 1 thread
  - RePlAce: used 1 thread
  - CMP: Innovus launched with 8 threads
  - AutoDMP: run on NVIDIA DGX-A100 machine with two GPU workers

*RePlAce run for MemPool Group did not complete

We thank David Junkin, Patrick Haspel, Angela Hwang and their colleagues at Cadence and Synopsys for policy changes that permit our methods and results to be reproducible and sharable in the open, toward advancement of research in the field. We thank **many** Google engineers (Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Eric Johnson, Roger Carpenter, Sergio Guadarrama, Guanhang Wu, Joe Jiang, Ebrahim Songhori, Young-Joon Lee and Ed Chi) and the TensorFlow Agents team for their time and discussions to clarify aspects of Circuit Training, and to run their internal flow with our data. We thank Ravi Varadarajan for early discussions and flow setup, and Mingyu Woo for guidance on RePlAce versions and setup. Support from NSF CCF-2112665 (TILOS) and DARPA HR0011-18-2-0032 (OpenROAD) is gratefully acknowledged.

THANK YOU !