The False Dawn: Reevaluating Google's Reinforcement Learning for Chip Macro Placement

arXiv:2306.09633

Igor L. Markov

Key sources and teams

Google Team 1 (Nature authors + coauthors)	Google Team 2 + external coauthors	UCSD Team
Circuit Training (CT) repo & FAQ [3] ISPD 2022 paper [4]	Stronger Baselines [5]	MacroPlacement repo & FAQ [6] ISPD 2023 paper [7] arXiv:2302.11014
4 proprietary TPU blocks ([1, Figure 3]) ariane (public) [3]	20 proprietary TPU blocks 17 public IBM circuits [21]	17 public IBM circuits [21] 2× ariane (public) [6, 7] 2× MemPool (public) [6, 7] 2× BlackParrot (public) [6, 7]

Table 1: Secondary sources published by the teams and chip designs for which they report results. The IBM circuits [21] are ICCAD 2004 benchmarks. [7] built 3 designs with 2 semiconductor technologies.

Outline

- Initial doubts about the Nature paper
- Additional evidence
- Did the Nature paper improve SOTA?
- Can the Nature work be used / improved?
- Conclusions about chip design
- Nature editors' investigation

Unsubstantiated claims in the Nature paper

- **U1**: “fast chip design” in the title was backed by the improvement of design-process time from “days or weeks” to “hours”
  - No per-design time or breakdown into stages
  - Did “days or weeks” for baseline design process include the time for functional design changes, idle time, inferior EDA tools?
- **U2:** six-hour runtimes for 5 TPU blocks do not include pre-training time on 20 blocks; RL runtimes only cover macro placement, but RePlAce and industry tools place all circuit components
- **U3**: no counts, sizes or shapes of macros in each TPU chip block used, no area utilization numbers, etc

Unsubstantiated claims in the Nature paper

- **U4**: results on only five TPU blocks, with unclear stat sig, but high-variance metrics that produce noisy results
- **U5:** “manual” design flow not described, not reproducible, results are subjective, potentially easy to improve
- **U6:** The abstract claimed improved “area”
  - Chip area and macro area did not change
  - Standard-cell area did not change during placement
  - Standard-cell area changes only after timing optimization (not done by RL)
  - The UCSD ISPD 2023 paper shows 0.0 correlation of area with RL proxy func
- **U7:** all improvements shown require much greater runtime (not fair?)

A flawed optimization proxy

- RL optimizes a simplified proxy
  - No timing analysis / optimization
- Then commercial tools perform global and detailed placement of standard cells, and full routing
- Power-grid design, clock-tree synthesis and timing closure are outsourced to an unknown third party
- Reported results include timing (TNS, WNS), power and standard-cell area, but none of these objectives are estimated within RL
  - No estimates of variance or stat sig in the Nature paper
  - As later shown in UCSD ISPD 2023 paper, correlations with the proxy are poor

Use of handicapped techniques

- **H1.** Exorbitant use of parallel computing and wall-clock time vs SOTA
- **H2**. Placing macros one by one vs. simulated annealing and analytical placers that iteratively refine locations
- **H3**. Using circuit-partitioning (clustering) methods similar to partitioning-based placement methods from 20 years ago, *without placement refinement*
- **H4.** Gridding *without placement refinement*
- **H5.** The use of force-directed placement from the 1980s

From Stronger Baselines

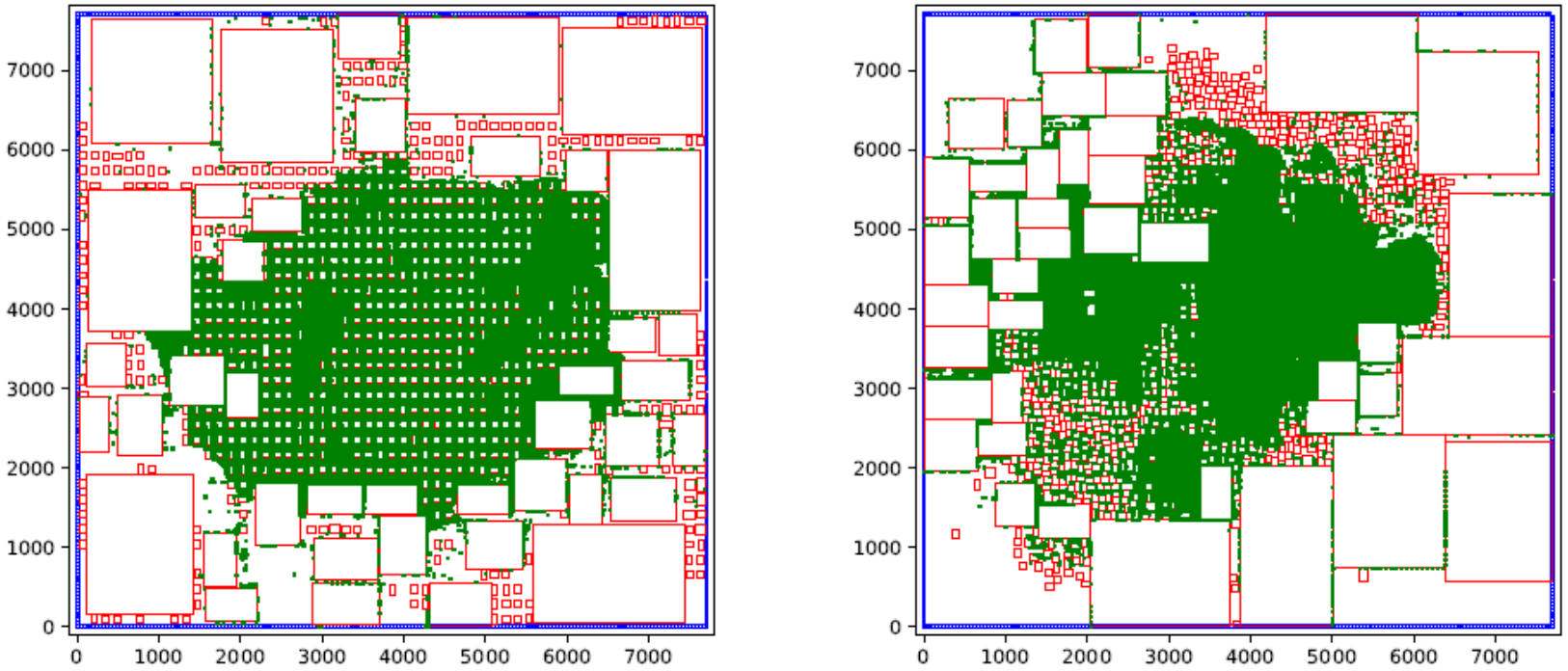

Figure 1: Layouts from [5, Fgure 2] with macros in red and standard cells in green, locations produced by RL (left) and RePlAce (right) for the **ibm10** benchmark from [21]. Limiting macro locations to a coarse grid (left) leads to spreading of small macros (red squares on a grid) and elongates connecting wires: from 27.5 (right) to 44.1 (left) for **ibm10** [5, Table 1]). High area utilization and many macros of different sizes make the ICCAD 2004 benchmarks [21] challenging compared to [1] and [2, page 43].

Questionable baselines

- **B1**. Comparisons to Simulated Annealing do not key chip metrics from Table 1, only proxy function values
- **B2.** For production use on TPUs, Simulated Annealing was used to postprocess the results of RL, but no ablation studies were given to evaluate the impact of SA on chip metrics
- **B3.** The RePlAce baseline was in a way inconsistent with its intended use: analytical placers assume large numbers of small components, not chunky clusters
- **B4.** Did not specify how macro locations were initialized for SA, suggesting a naive approach that could be improved.

Google source code: Circuit Training

- Google released RL code in the open source, 7 months after Nature publication, but the code was missing critical pieces and did not run
- Under pressure, some components were released and have shown that the clustering/grouping flow for RL required initial (x,y) placements, which were produced at Google by Synopsys tools
- Eventually, the UCSD team reimplemented all missing components and were able to run experiments

Additional evidence

- In responses to peer reviews, authors stated that macro locations were not modified after placement by RL, confirming coarse-grid placement of macros
- The UCSD ISPD 2023 paper has shown that human designers produce
  - better placements than those produced by Google RL
  - Better placements when not limited to a grid
- The Stronger Baselines paper and the SPD 2023 paper implemented request of Reviewer #3 to perform comparisons on ICCAD 04 benchmarks
  - Google RL decisively lost by >20% in HPWL

Correlation studies from ISPD 2023

Chip metrics →	area	routed wirelength	power	WNS	TNS
Rank correlation to RL proxy cost	**0.00**	**0.28**	**0.05**	**0.2**	**0.05**
Mean μ	247.1K	834.8	4,978	-100	-65
Standard deviation σ	1.652K	4.1	272	28	36.9
$\sigma/\lvert\mu\rvert$	0.01	0.00	0.05	**0.28**	**0.57**

Table 2: Evaluating the soundness of the proxy cost used with RL in [1] and the noisiness of reported chip metrics after RL-based optimization. We summarize data from [7, Table 2] on the Kendall rank correlation of chip metrics to the RL proxy cost and from [7, Tables 3 and 4] on statistics for chip metrics (only Ariane-NG45 design dats is shown, but data for BlackParrot-NG45 shows similar trends). As expected, purely-additive metrics (standard-cell area, routed wirelength and chip power) exhibit low variance, but the TNS and WNS metrics, that measure timing-constraint violations, have high variance.

Did the Nature article improve SOTA?

- The Stronger Baselines and the UCSD ISPD 2023 papers performed similar experiments and also some unique experiments (➔ same conclusions)
  - Stronger Baselines has access to Google TPU blocks and original RL code, but ISPD 2023 didn't
  - ISPD 2023 created several large modern benchmarks and open-sourced them
  - **Stronger Baselines did not use initial (x,y) placements, but ISPD 2023 did**
  - **ISPD 2023 did not use pre-training, but Stronger Baselines (SB) tried placing macros with and w/o (the impact of pre-training was small at best)**
  - **Stronger Baselines did not perform routing and timing optimization, nor evaluated chip metrics, but ISPD 2023 did**
  - Both papers used RePlAce as it was intended and obtained better results
  - **The UCSD team had to reimplement Sim Annealing, but replicated SB results**
  - **The ISPD 2023 paper compared everything to Cadence CMP, which was far ahead**

Simulated Annealing implementation

- Stronger Baselines explained that the SA implementation in the Nature paper was handicapped
  - Adding two additional actions “move” and “shuffle” to “swap”, “mirror” and “shift” improved results (this was replicated by ISPD 2023 paper)
  - Initializing macro locations with a simple heuristic improved results
- SA consistently outperformed RL by the proxy function and runtime
  - Can replace RL in the proposed flow

Runtimes

↓ Designs / Tools →	Google CT	Cadence CMP	UCSD SA
Ariane-NG45	32.31	0.05	12.50
BlackParrot-NG45	50.51	0.33	12.50
MemPool-NG45	81.23	1.27	12.50

Table 3: Runtimes in hours for three mixed-size placement tools and methodologies on three large chip modern designs reported in the arXiv version of [7]. Google CT - Circuit Training code supporting RL in the Nature paper, used without pre-training. Cadence CMP - Concurrent Macro Placer (commercial EDA tool). SA - parallel Simulated Annealing implemented at UCSD following [5] given 12.5 hours of runtime in each case. CT and SA are used only to place macros, the remaining components are placed by a commercial EDA tool whose runtime is not included. Cadence CMP places all circuit components.

Can the Nature work be used / improved?

- High variance of TNS and WNS
- Dogfooding
- Postprocessing by SA
- Pre-training on similar designs

Obstacles to improvement

- The proxy function
- Gridding and clustering
- Long runtimes and use of large compute resources

Conclusions for chip design

- Limited training data for ML
- Gridding and clustering – a handicap
- Exorbitant CPU resources don’t seem to help in placement
- An incremental placement improvement flow – worth exploring
- Modern open-source infrastructure developed at UCSD for macro placement experiments

Nature investigation

Article | Published: 09 June 2021

A graph placement methodology for fast chip design

Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang, Ebrahim Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nazi, Jiwoo Pak, Andy Tong, Kavya Srinivasa, William Hang, Emre Tuncer, Quoc V. Le, James Laudon, Richard Ho, Roger Carpenter & Jeff Dean

Nature **594**, 207–212 (2021) | Cite this article

20 September 2023 Editor's Note: Readers are alerted that the performance claims in this article have been called into question. The Editors are investigating these concerns, and, if appropriate, editorial action will be taken once this investigation is complete.