Risk-Averse Thermal-Aware 2D Macro Placement under Workload Uncertainty

Georgia Tech College of Engineering
H. Milton Stewart School of Industrial and Systems Engineering

Arsen Kozhabek[1] and Johannes Milz[2]

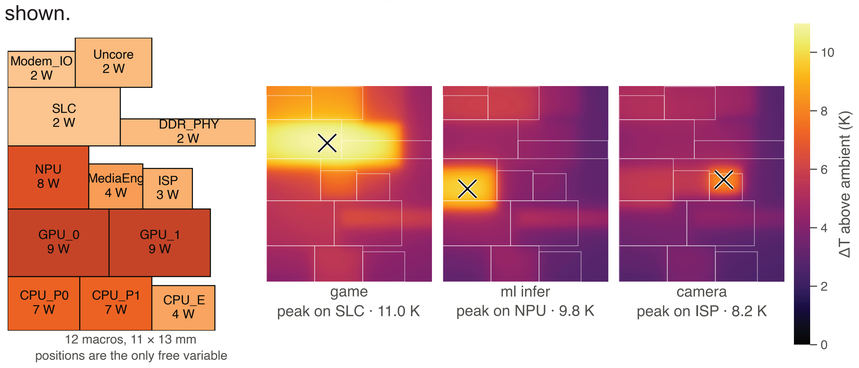
1. School of Electrical and Computer Engineering, 2. H. Milton Stewart School of Industrial and Systems Engineering

The problem

The workload is uncertain. The placement is fixed.

Placing the macros (a pre-designed functional block of circuitry) of a computer chip is a trade-off between wirelength, timing, and heat.

The thermal component of macros depends on software: different applications and execution phases activate different functional units, producing different power maps across the die, and therefore different hotspots.; The macro placement is chosen once, during the chip-design stage, and cannot change after fabrication. The workload, however, keeps changing, so a single, irreversible placement is judged against an entire ensemble of workloads it was never shown.



Uncertain workloads create a great challenge

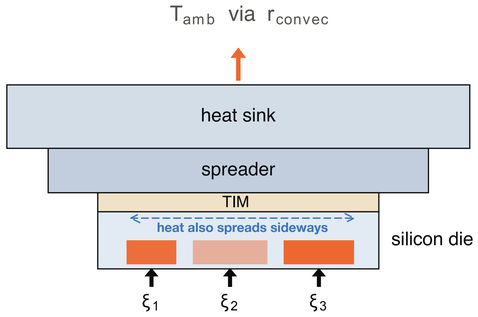
The chip carries ***B*** rectangular macros of fixed size. Position is the only choice, so a placement is the vector ***p*** of macro centers:

$$p = \left((c_x^1, c_y^1),\ \ldots,\ (c_x^B, c_y^B)\right) \in \mathcal{X} \subset \mathbb{R}^{2B}$$

where ***X*** is the die rectangle intersected with pairwise non-overlap. A placement alone has no temperature; that requires a workload ***ξ***, the activity level of each macro.

Ranking placements needs one scalar, and the binding one is peak temperature. A placement meets a whole ensemble of workloads, so it faces a distribution of peaks; ranking means choosing which statistic of that distribution to minimize.

From placement to temperature



2D Four-layer chip package.
Heat leaves the silicon upward to ambient, spreading laterally on the way.

Discretizing the package on a uniform grid turns the physics into one sparse linear system:

$$G\,T(p,\xi) = Q(p,\xi)$$

with ***G*** from geometry and materials by Kirchhoff's current law. Because the layers are laterally uniform, ***G*** does not depend on where the macros sit — placement changes where heat is injected, not how it propagates, and enters only the right-hand side:

$$Q(p,\xi) = Q_0 + A(p)\,\xi$$

The hottest silicon cell of $T = G^{-1}Q$ is the scalar we asked for:

$$M(p,\xi) = \max_{i \in \mathcal{I}_{\mathrm{Si}}} T_i(p,\xi) - T_{\mathrm{amb}}$$

Fixed ***G***, placement-dependent ***Q*** is what makes this possible. One factorization serves every scenario; the derivative of ***M*** is one back-substitution against it. Thousands of workloads per step become affordable — so an objective over the whole distribution is finally in reach.

The machine and measurement

Mean, or tail?

The mean smooths over precisely the workloads a thermal designer worries about. So alternative (or addition) to it is ***CVaRα***, the mean of the worst ***(1−α)*** fraction of outcomes:

$$\min_p \mathbb{E}_{\mathbb{P}}[M] \qquad \text{versus} \qquad \min_{p \in \mathcal{X}} \mathrm{CVaR}_\alpha[M(p,\xi)]$$

$$\mathrm{CVaR}_\alpha[M] = \min_{t \in \mathbb{R}} \left\{ t + \frac{1}{1-\alpha}\,\mathbb{E}[(M-t)_+] \right\}$$

At the inner minimizer, $t^* = \mathrm{VaR}_\alpha$ and the t-term drops out, leaving the mean's per-scenario gradients reweighted onto the tail:

$$\frac{1}{1-\alpha}\,\mathbb{E}\left[\mathbf{1}\{M \geq t^*\}\,\nabla_p M\right] \ \in\ \partial_p\,\mathrm{CVaR}_\alpha[M(p,\xi)]$$

And the tail costs information. That subgradient is supported on only ***[N(1 − α)]*** scenarios at ***α = 0.9***, ten times fewer samples than the mean gradient. Noisier, and sharper.

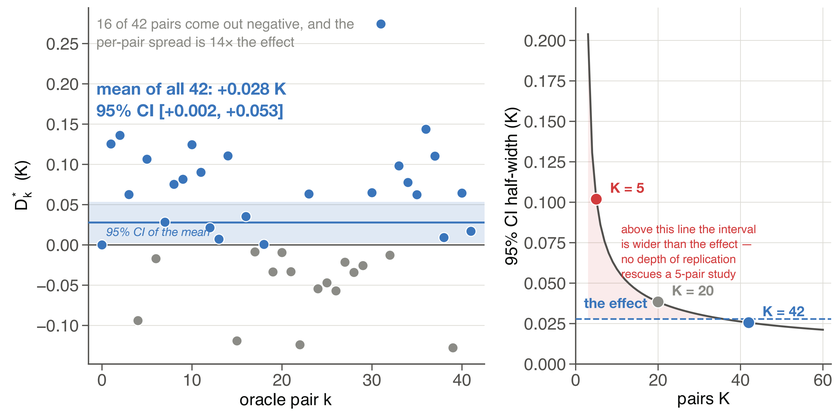
Which makes this a measurement problem

Whether the tail objective actually lowers tail temperature cannot be settled by analysis, only measured. The comparison is not between two placements but between two procedures: the training run that minimizes CVaR against the one that minimizes the mean. A procedure is a random map $\omega \mapsto p(\omega)$ from seed to placement, so the quantity of interest is a property of that map — the expected gap:

$$D_k^* = \mathrm{CVaR}_\alpha(p_{\mathrm{mean}}) - \mathrm{CVaR}_\alpha(p_{\mathrm{cvar}}), \qquad \bar{D}^* = \frac{1}{K}\sum_{k=1}^{K} D_k^*$$

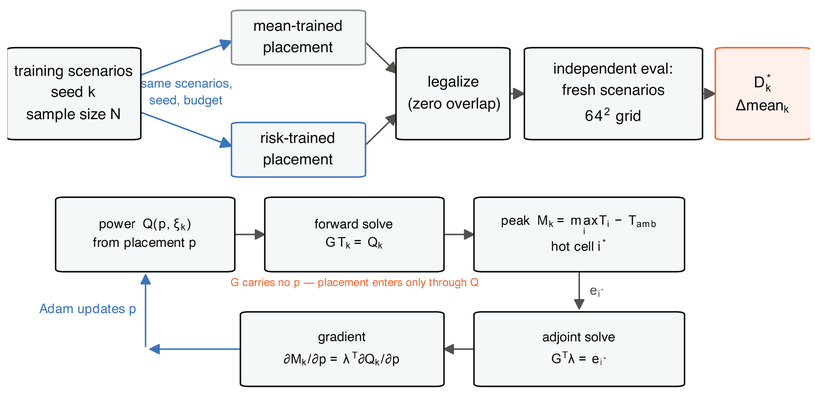
Each oracle pair ***k*** gives one measured gap ***Dk***; we report their mean ***D*** over ***K = 42 pairs.*** Neither arm returns an argmin — ***X*** is disjunctive, ***M(·, ξ)*** is nonconvex, the descent is a subgradient method — and each minimizes an empirical ***CVaR*** on finitely many scenarios. So ***D**** compares procedures, not optima, and every property of the

With exact minimizers the true gap is ***≥ 0*** by construction — the ***CVaR-optimal*** placement cannot be beaten on ***CVaR***. That does not transfer to ***D****, so a significantly negative ***D**** is diagnostic — of finite samples, or one arm stuck in a worse local optimum, never of the theory. Our first study measured exactly that.



Per-pair gap D, 42 pairs. Mean +0.028 K, but 16 negative and the spread is 14× the effect.*

CI width vs pairs K. Under ~5 pairs the interval exceeds the effect; ~42 are needed to clear it.



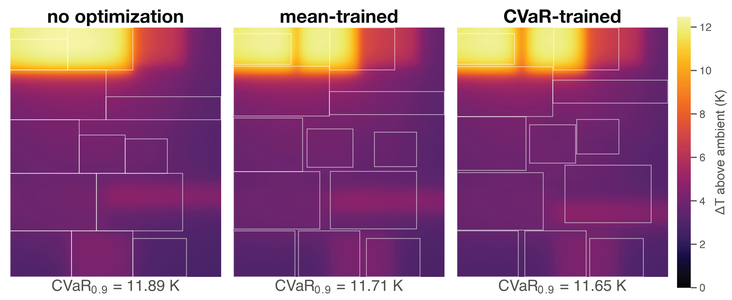
one factorization of G serves every scenario and every step; the adjoint reuses it

What survives measurement?

The result

A trained placement is not a measurement — it is an optimizer's output: finitely many Adam steps, a finite sample, a chosen mesh, constraints enforced only approximately. Each is a knob that can favor one arm, and the sharper tail objective exploits any slack. So the knobs are equalized before anything is measured — common random numbers, matched budget, identical legalization — and the mesh is escaped rather than shared: jittered in training, then scored on a finer one neither arm saw.

+0.066 K [+0.059, +0.074] engineered SoC, trade-shaped (Δmean < 0)
+0.028 K [+0.002, +0.053] guaranteed-legal placements under i.i.d. power

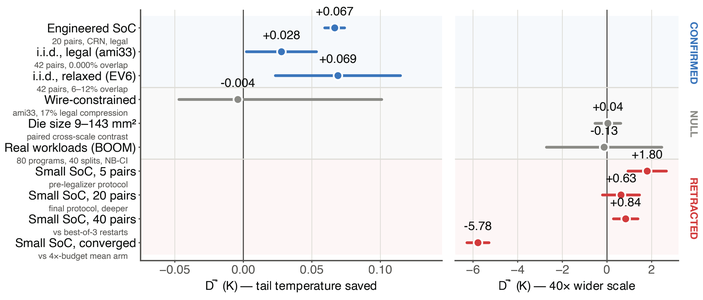


When should a chip designer bother?

CHECK ALL THREE BEFORE OPTIMIZING

1. **Multimodal workloads** — distinct phases exciting distinct macros
2. **A mobile hotspot** — migrating between anti-correlated macros that carry substantial power
3. **Placement freedom** — whitespace on the die to move into

Pass all three and expect ~0.06 K optimization



D across testbeds and protocols, grouped by verdict. Left: sound protocols — the effect is real but under 0.1 K. Right (40× wider scale): null results, and small-sample or illegal-overlap runs that inflate D* and were retracted, not reported.*

Acknowledgements

Supported by ISyE 2026 Summer Scholars Program at Georgia Tech. This research was supported in part through research cyberinfrastructure resources and services provided by the Partnership for an Advanced Computing Environment (PACE) at the Georgia Institute of Technology.

Workload traces from the McPAT-Calib release; the ami33 floorplan from the GSRC benchmark distribution. Neither is redistributed with our code.

References:
1. Rockafellar RT, Uryasev S (2000) Optimization of conditional value-at-risk. J Risk, 2(3):21–41.
2. Huang W, Ghosh S, Velusamy S, Sankaranarayanan K, Skadron K, Stan MR (2006) HotSpot: a compact thermal modeling methodology for early-stage VLSI design. IEEE Trans VLSI Syst, 14(5):501–513.
3. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. ICLR.
4. Lin Y, Dhar S, Li W, Ren H, Khailany B, Pan DZ (2019) DREAMPlace: deep-learning-toolkit-enabled GPU acceleration for VLSI placement. DAC.
5. Zhai J, et al. (2022) McPAT-Calib: a RISC-V BOOM microarchitecture power modeling framework calibrated against RTL sign-off power. IEEE TCAD.
6. Di Mauro A, et al. (2022) Kraken: a direct event/frame-based multi-sensor fusion SoC for ultra-efficient visual processing in nano-UAVs. arXiv:2209.01065.
7. Holm S (1979) A simple sequentially rejective multiple test procedure. Scand J Statist, 6(2):65–70.
8. Nadeau C, Bengio Y (2003) Inference for the generalization error. Machine Learning, 52(3):239–281.
9. MCNC/GSRC floorplanning benchmark suite (ami33): 33 macros, 78 nets.
10. DiffChip: differentiable thermal-aware chip placement. arXiv:2502.16633 (2025).
11. ATPlace2.5D: analytical thermal-aware placement for 2.5D systems. ICCAD (2024).