Multiplicative vs. Additive State Coupling in Learned Optimizer Schedules

M. Harper Langston Pierre-David Letourneau Dalton Jones Roland Memisevic Peter Scott Hong Cai
Mingu Lee Richard A. Lethin Christopher Lott Fatih Porikli Harris Teague
Qualcomm AI Research

OPT Workshop 2026

Key finding: *how* solver state enters a learned schedule decides out-of-distribution transfer

+0.0084
Bilinear: Biq Mac ratio gain over its own frozen baseline (paired $t = 4.1$, 15 seeds)

–0.0522
Additive: same inputs and heads, *worse* than its baseline ($t = -11.7$)

5 of 5
weight-decay values: bilinear stays positive, additive never recovers

Same in distribution; opposite on instances far larger than any seen in training.

Learn the schedule, keep the solver

Spins $s = \tanh(u)$, adjacency W. Penalty-continuation solver:

$$\Phi(u; W, \rho) = -\widehat{\text{cut}}(u) + \rho\,\text{bviol}(u), \quad \text{bviol} = \tfrac{1}{n}\textstyle\sum_i(\tanh(u_i)^2 - 1)^2.$$

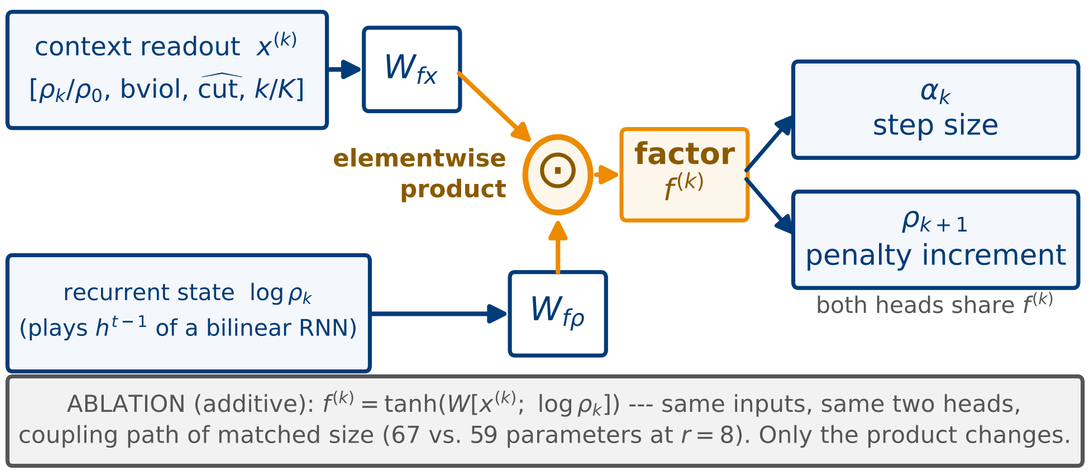
K stages, each with T gradient steps at step size α_k and penalty ρ_k.

- We **learn** (α_k, ρ_k) by unrolling all KT inner updates and backpropagating through them.
- Unlike learned solvers that predict the *output* (OptGNN, neural SDP), the classical solver is untouched.
- **Question:** does the schedule's architecture generalize beyond its training distribution?

Three schedules, one varied factor

State-blind: one (α_k, ρ_k) per stage; knows the stage, not the trajectory.
State-adaptive: sees $x^{(k)} = [\rho_k/\rho_0,\ \text{bviol},\ \widehat{\text{cut}},\ k/K]$ and $\log \rho_k$.

$$f_{\text{bil}} = (W_{fx}\, x^{(k)}) \odot (w_{f\rho} \log \rho_k) \qquad f_{\text{add}} = \tanh\big(W_{\text{add}}[x^{(k)}; \log \rho_k] + b\big)$$



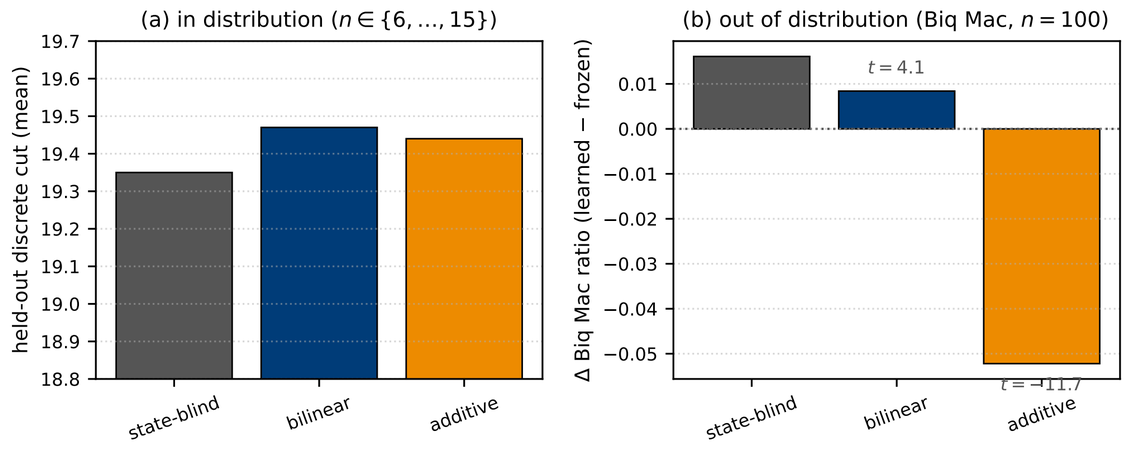
Same inputs, heads and capacity: only the **coupling** differs.

Protocol and in-distribution results

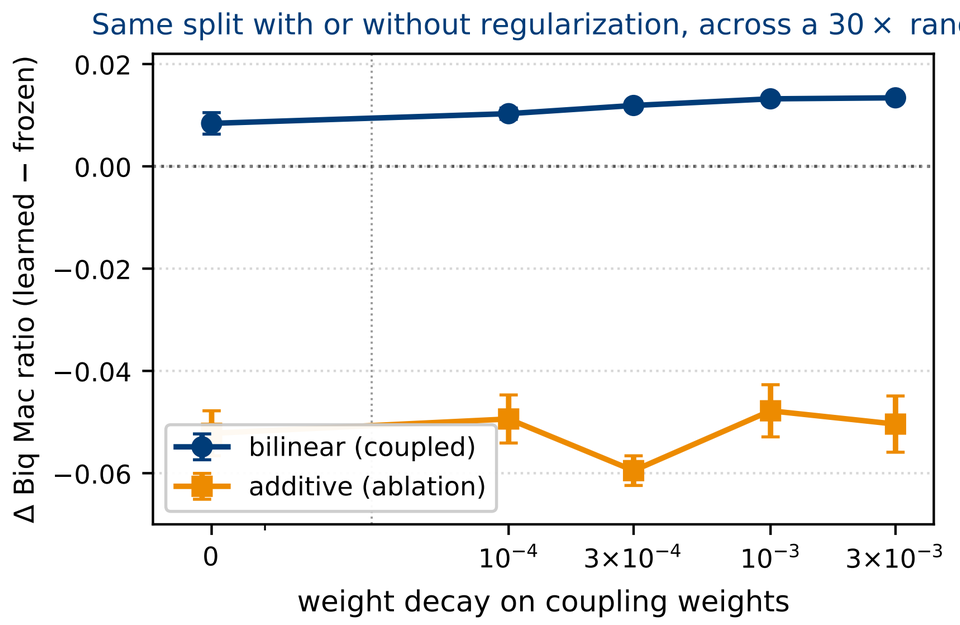
- **Train:** $G(n, 0.5)$, $n \in \{6, \ldots, 15\}$, $K = 7$, 200 epochs. **Test:** $n \le 100$ and 80 Biq Mac instances ($n = 100$). 15 seeds; each model vs. its own frozen baseline.
- **In distribution:** mean cut 19.35–19.47 for all three. Hard $n = 15$ instance solved in additive 13/15, bilinear 12/15, state-blind 7/15 runs. **Bilinear and additive are indistinguishable here.**

Out of distribution ($n = 100$): opposite signs

	synthetic cut		Biq Mac ratio	
Model	frozen	learned	frozen	learned
State-blind	1393.1	1406.9	0.9259	0.9420
Bilinear	1410.9	1415.4	0.9543	**0.9627**
Additive	1411.0	1397.9	0.9545	0.9023
Tuned fixed schedule			0.9624	



Not a regularization artifact



Five weight-decay values, 0 to 3×10^{-3}: bilinear $\Delta \in [+0.0084, +0.0134]$, additive $\Delta \in [-0.0595, -0.0478]$.

What this does and does not show

- A **tuned fixed schedule** (0.9624) is statistically *tied* with bilinear (0.9627): an architectural result, *not* "learning beats tuning".
- **Mechanism is a hypothesis** (unsaturated product vs. self-limiting tanh), unverified for MAX-CUT. Next: saturating-bilinear vs. unsaturated-additive ablation.
- **Scope:** one relaxation, solver, training set and suite. A knapsack extension shows the same coupling asymmetry but *no* OOD separation.

Takeaways

1. Reading solver state *multiplicatively* generalizes to instances $\approx 7\times$ larger than the training range; a capacity-matched *additive* schedule does not.
2. Invisible in distribution, unchanged by weight decay: **how state enters** is the variable, not state-adaptivity alone.
3. Open: mechanism, other solvers, and optimizer families (Adam vs. Shampoo / K-FAC / Muon). No transfer claim.

Paper

arXiv:2610.32801