Multi-Objective Reinforcement Learning and Planning for Large Language Models



Yining Lu

University of Notre Dame

November 13, 2026

MOTIVATION

RL for reasoning usually optimizes accuracy

$$\max_{\theta} \quad \mathbb{E}_{y \sim \pi_\theta} \left[\text{accuracy}(y) \right]$$

MOTIVATION

RL for reasoning usually optimizes accuracy

$$\max_{\theta} \quad \mathbb{E}_{y \sim \pi_\theta} \left[\text{accuracy}(y) \right]$$

How many positive factors does 360 have?

MOTIVATION

RL for reasoning usually optimizes accuracy

$$\max_{\theta} \quad \mathbb{E}_{y \sim \pi_\theta} \left[\text{accuracy}(y) \right]$$

How many positive factors does 360 have?

ANSWER A

360 = $2^3 \times 3^2 \times 5$, so it has (3+1)(2+1)(1+1) = 24 factors.
24 factors.

Correct **Cost:** about 35 tokens **Latency:** about 1 s

ANSWER B

First, factor into primes: 360 = $2^3 \times 3^2 \times 5$.
Second, a factor can use 0 to 3 twos, 0 to 2 threes, and 0 or 1 five.
Third, count the choices: 4 × 3 × 2 = 24.
So 360 has 24 factors.

Correct **Cost:** about 80 tokens **Latency:** about 2 s

Both get reward 1. A student checking an answer wants A. The same student stuck on the problem wants B.

MOTIVATION

Real systems care about more than accuracy

Accuracy	Cost	Latency	Robustness	Consistency
is the answer right?	tokens spent	time to answer	stable when the questions are noisy	same answer when asked twice

A high-volume API cares most about cost and latency.

A scientific pipeline cares most about robustness and consistency.

MOTIVATION

Real systems care about more than accuracy

Accuracy	Cost	Latency	Robustness	Consistency
is the answer right?	tokens spent	time to answer	stable when the questions are noisy	same answer when asked twice

A high-volume API cares most about cost and latency.

A scientific pipeline cares most about robustness and consistency.

MOTIVATION

The usual fix is a weighted sum

$$r = w_1 \,\text{accuracy} + w_2 \,\text{conciseness} + w_3 \,\text{clarity}$$

accuracy-focused	balanced	efficiency-focused
(0.50, 0.25, 0.25)	(⅓, ⅓, ⅓)	(0.25, 0.375, 0.375)

Each weight vector gives one trade-off. A different trade-off needs a new training run.

MOTIVATION

The usual fix is a weighted sum

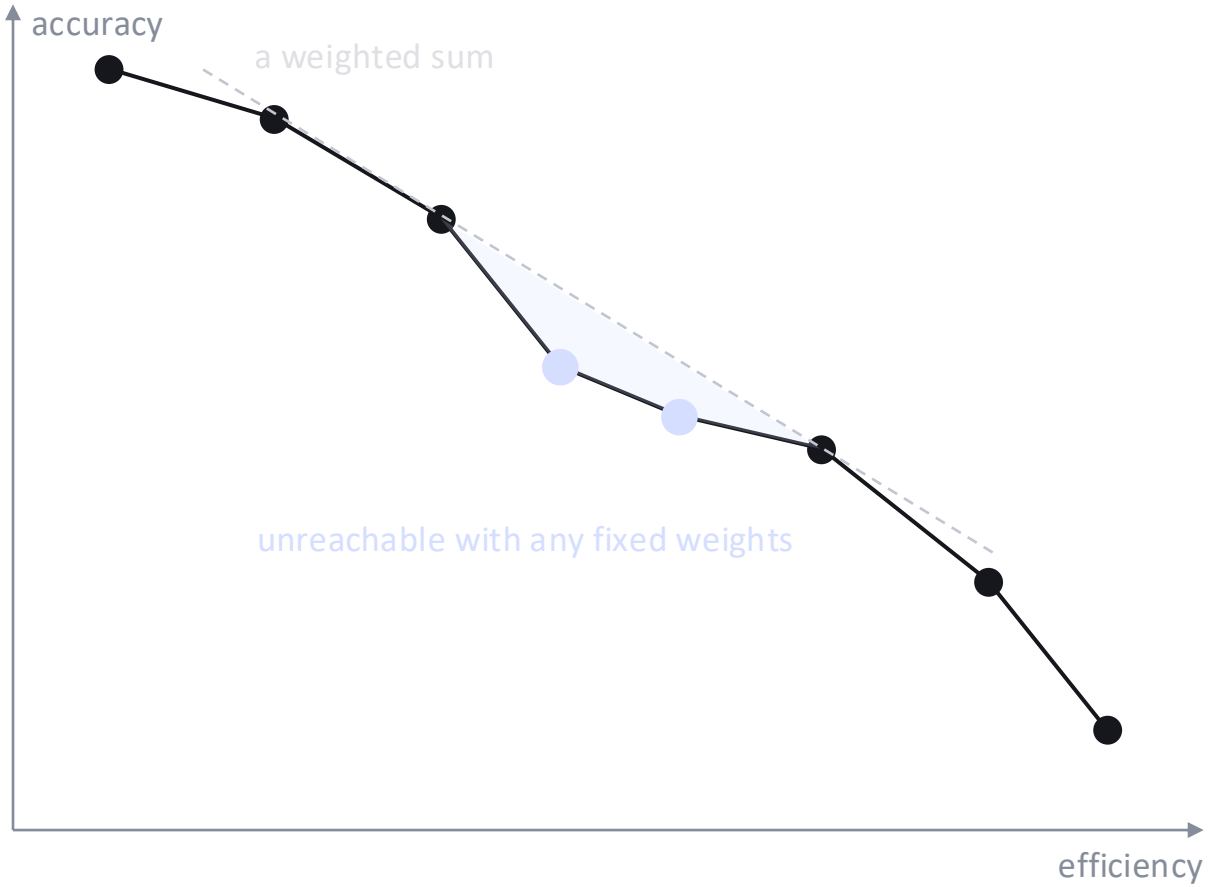
$$r = w_1 \,\text{accuracy} + w_2 \,\text{conciseness} + w_3 \,\text{clarity}$$

accuracy-focused	balanced	efficiency-focused
(0.50, 0.25, 0.25)	**(⅓, ⅓, ⅓)**	**(0.25, 0.375, 0.375)**

Each weight vector gives one trade-off. A different trade-off needs a new training run.

MOTIVATION

Fixed weights only reach the convex hull



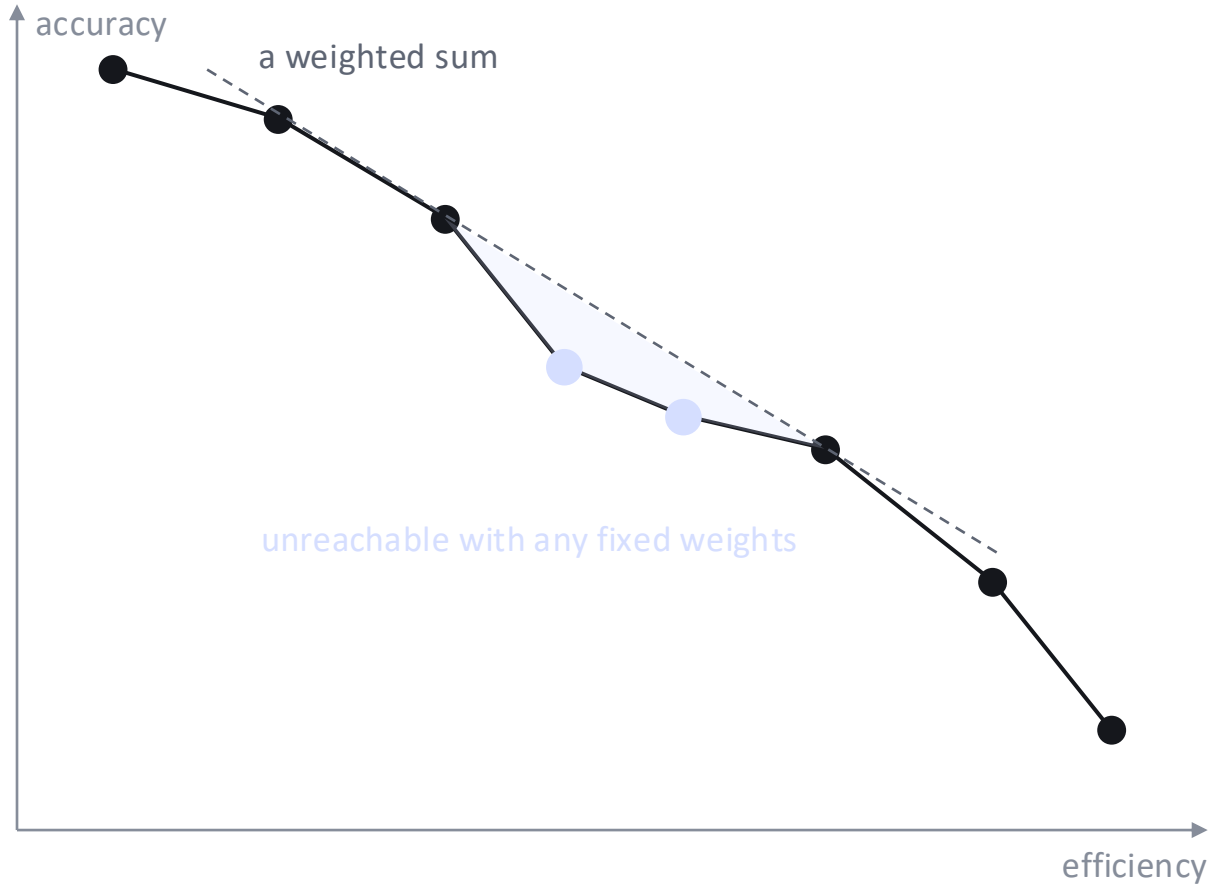
The Pareto front is the set of trade-offs where no objective can improve without another getting worse.

Maximizing a weighted sum is like pressing pressing a ruler against the front. It only only touches the convex hull.

Points in a concave region are never optimal for any fixed weights.

MOTIVATION

Fixed weights only reach the convex hull



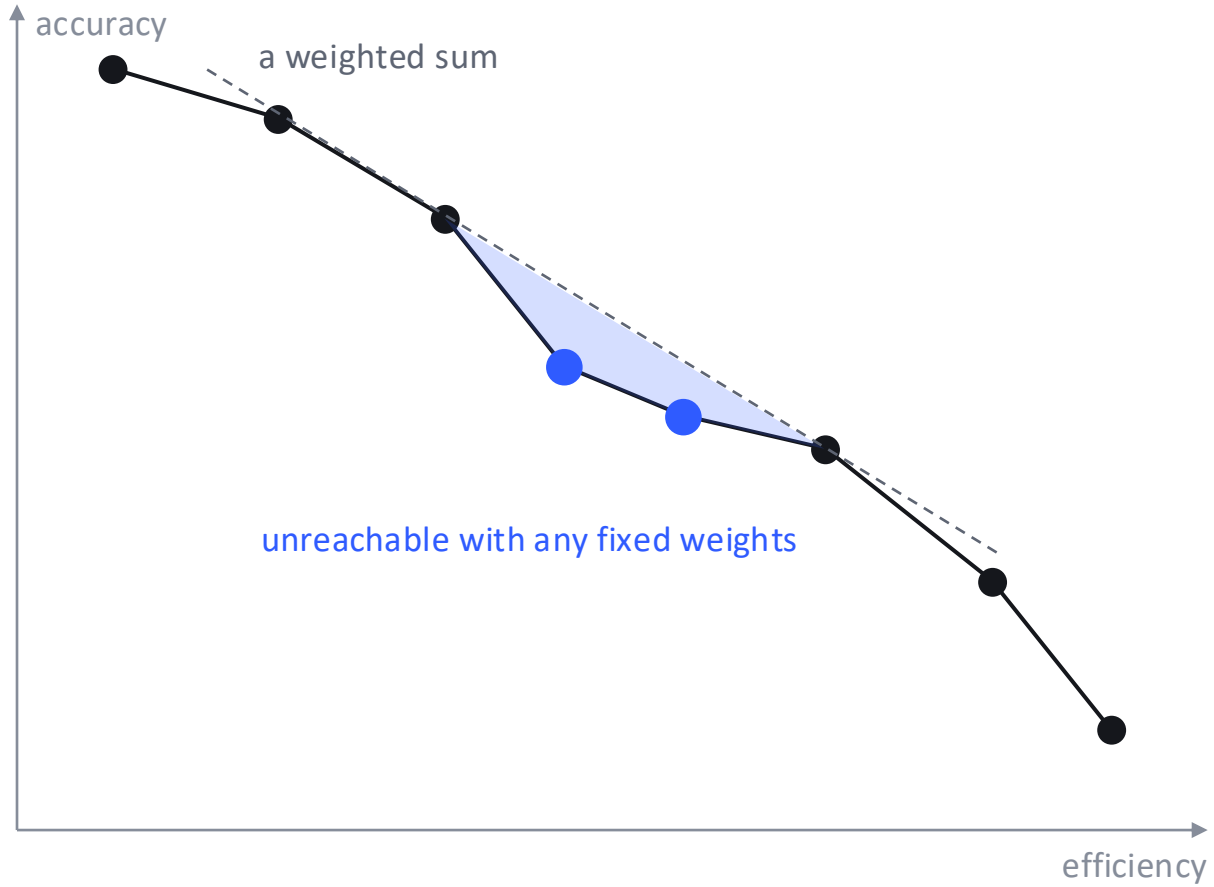
The Pareto front is the set of trade-offs where no objective can improve without another getting worse.

Maximizing a weighted sum is like pressing a ruler against the front. It only touches the convex hull.

Points in a concave region are never optimal optimal for any fixed weights.

MOTIVATION

Fixed weights only reach the convex hull



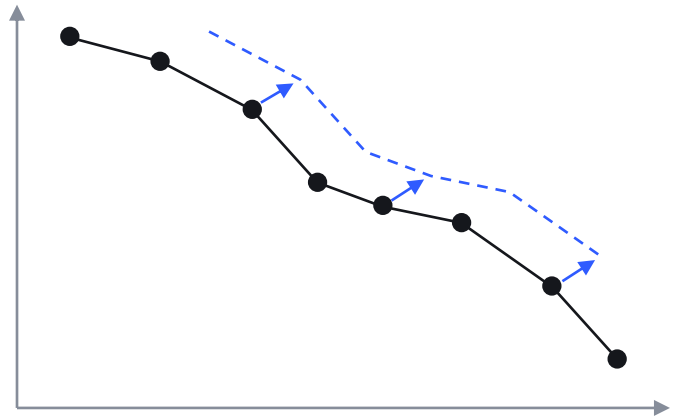
The Pareto front is the set of trade-offs where no objective can improve without another getting worse.

Maximizing a weighted sum is like pressing a ruler against the front. It only touches the convex hull.

Points in a concave region are never optimal optimal for any fixed weights.

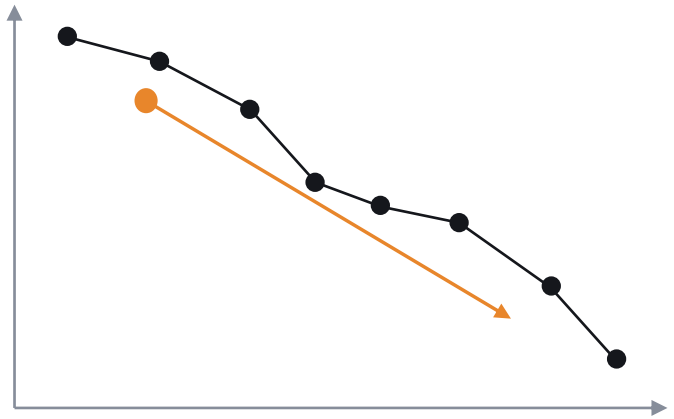
OUTLINE

Three questions



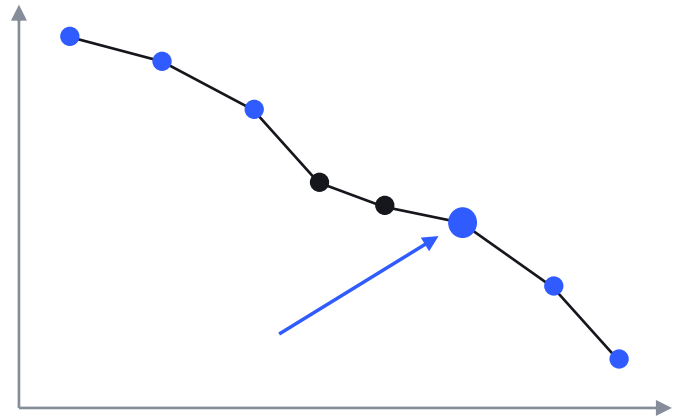
1 Training

Can training reach better trade-offs than a fixed weighted sum?



2 Interference

Why does training sometimes sacrifice one objective for another?



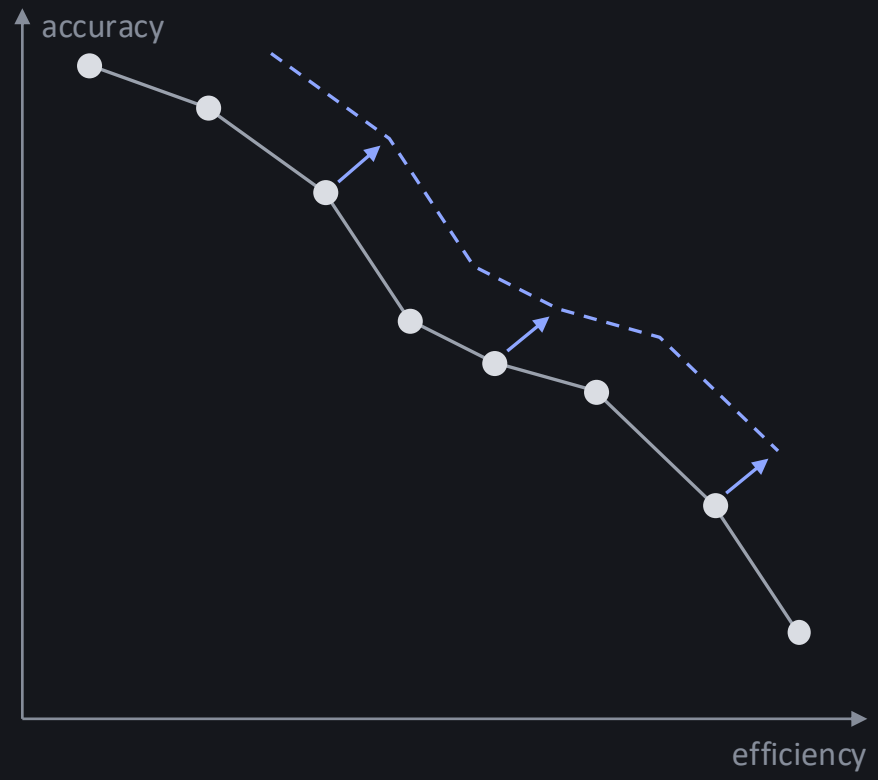
3 Planning

Can one system serve any trade-off without retraining?

1

Training

Can training reach better trade-offs than a fixed weighted sum?



Learning to Optimize Multi-Objective Alignment Through Dynamic Reward Weighting

Yining Lu, Zilong Wang, Shiyang Li, Xin Liu, Changlong Yu, Qingyu Yin, Zhan Shi, Zixuan Zhang, Meng Jiang

University of Notre Dame and Amazon

TACL 2026

PART 1: TRAINING

The three objectives learn at different speeds

Accuracy

final answer matches the reference

Conciseness

shorter than the recent average length

Clarity

explicit step markers (first, second, ...)

PART 1: TRAINING

The three objectives learn at different speeds

Accuracy

final answer matches the reference

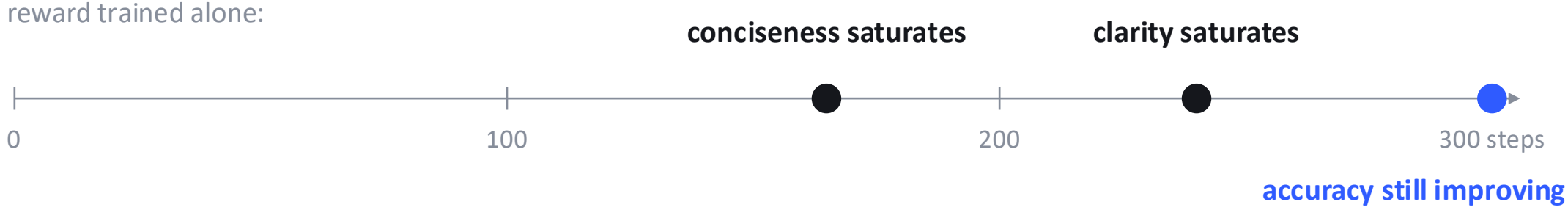
Conciseness

shorter than the recent average length

Clarity

explicit step markers (first, second, second, ...)

Each reward trained alone:



PART 1: TRAINING

The three objectives learn at different speeds

Accuracy

final answer matches the reference

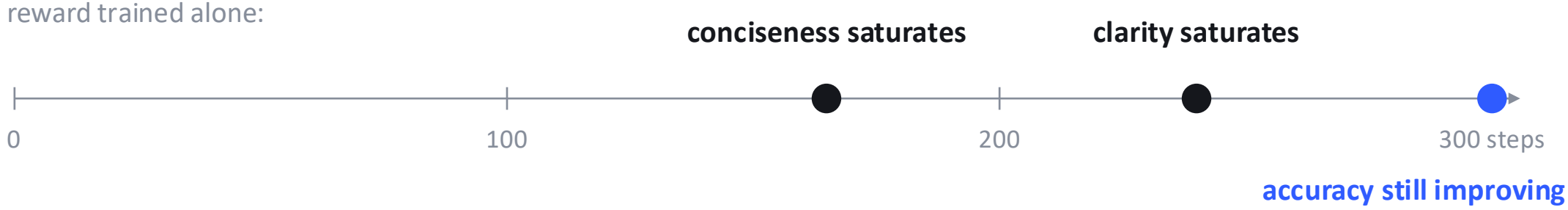
Conciseness

shorter than the recent average length

Clarity

explicit step markers (first, second, ...)

Each reward trained alone:



With fixed weights, an objective that is already learned keeps its share of every update. We move weight to the objectives that still have room to improve.

PART 1: TRAINING

Learn the weights from gradients

When the user has no preferences

WEIGHT UPDATE

$$w_i \leftarrow w_i \exp\left(\eta I_i / \mu\right)$$

INFLUENCE OF OBJECTIVE i

$$I_i = \left\langle \nabla J_i, \sum_k \nabla J_k \right\rangle = \underbrace{\|\nabla J_i\|^2}_{\text{room to improve}} + \underbrace{\sum_{k \neq i} \langle \nabla J_i, \nabla J_k \rangle}_{\text{alignment with the others}}$$

A weight grows when its gradient points along the total gradient. This is mirror descent on the simplex, and we simplex, and we prove the weight ratios stay bounded.

PART 1: TRAINING

Learn the weights from gradients

When the user has no preferences

WEIGHT UPDATE

$$w_i \leftarrow w_i \exp\left(\eta I_i / \mu\right)$$

INFLUENCE OF OBJECTIVE i

$$I_i = \left\langle \nabla J_i, \sum_k \nabla J_k \right\rangle = \underbrace{\|\nabla J_i\|^2}_{\text{room to improve}} + \underbrace{\sum_{k \neq i} \langle \nabla J_i, \nabla J_k \rangle}_{\text{alignment with the others}}$$

A weight grows when its gradient points along the total gradient. This is mirror descent on the simplex, and we prove the weight ratios stay bounded.

PART 1: TRAINING

Learn the weights from gradients

When the user has no preferences

WEIGHT UPDATE

$$w_i \leftarrow w_i \exp\left(\eta I_i / \mu\right)$$

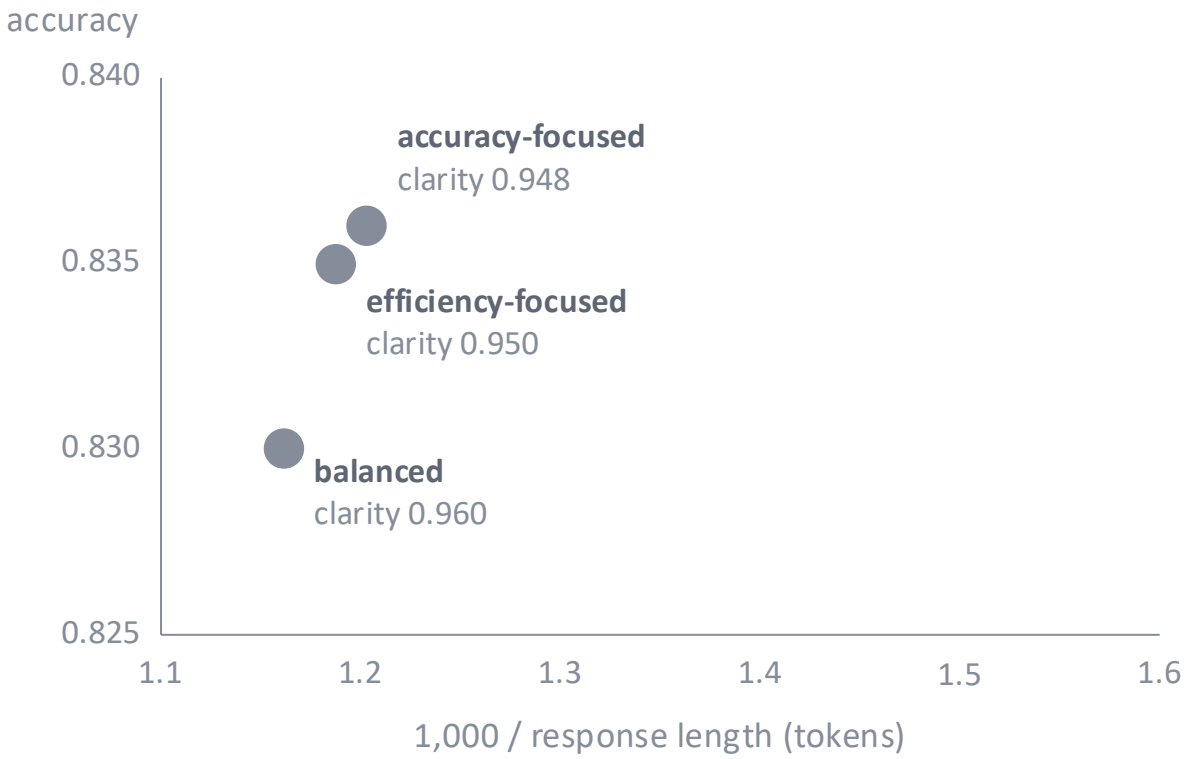
INFLUENCE OF OBJECTIVE i

$$I_i = \left\langle \nabla J_i, \sum_k \nabla J_k \right\rangle = \underbrace{\|\nabla J_i\|^2}_{\text{room to improve}} + \underbrace{\sum_{k \neq i} \langle \nabla J_i, \nabla J_k \rangle}_{\text{alignment with the others}}$$

A weight grows when its gradient points along the total gradient. This is mirror descent on the simplex, and we prove the weight ratios stay bounded.

PART 1: TRAINING

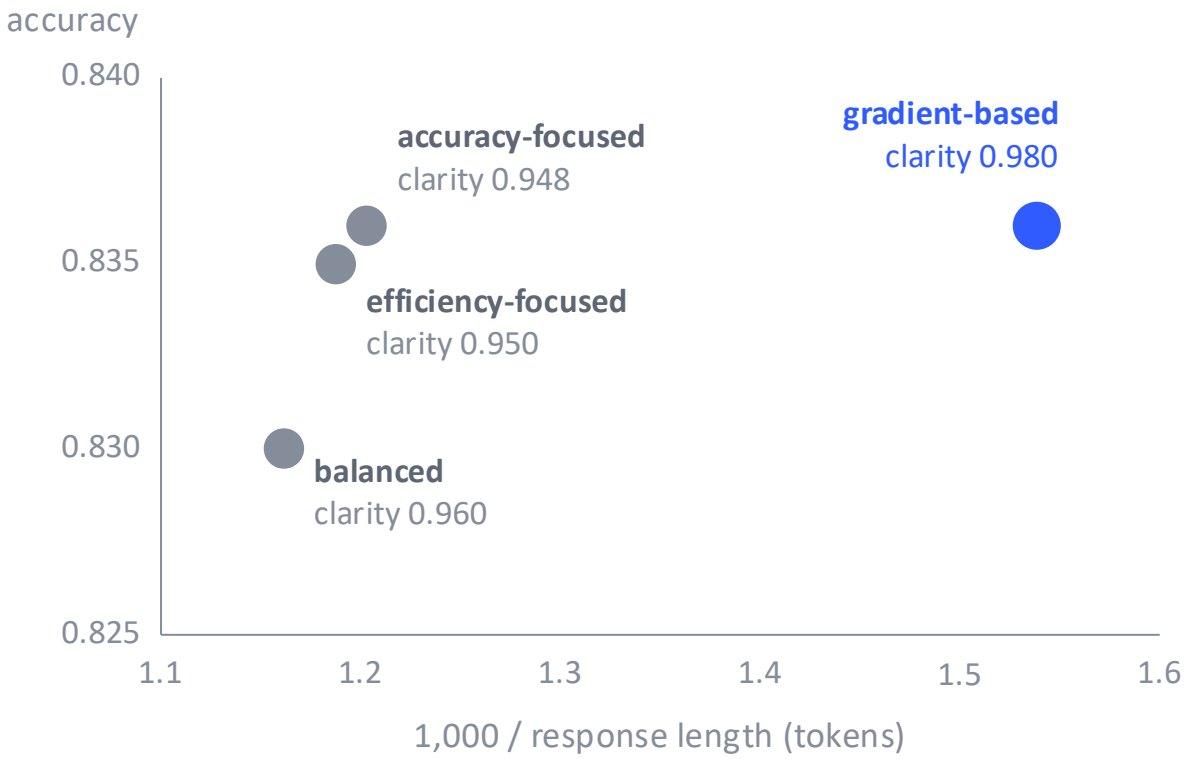
Same accuracy, shorter and clearer answers



GRPO, Qwen3-8B, MATH500.

PART 1: TRAINING

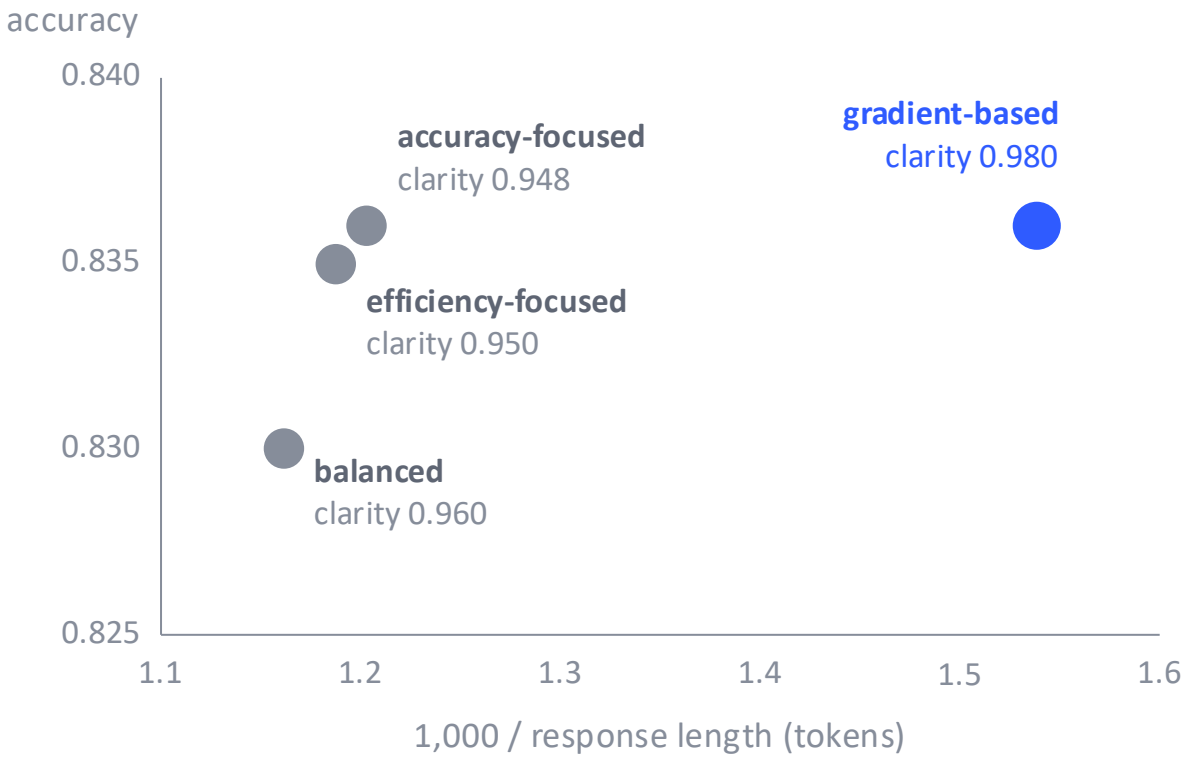
Same accuracy, shorter and clearer answers



GRPO, Qwen3-8B, MATH500.

PART 1: TRAINING

Same accuracy, shorter and clearer answers



GRPO, Qwen3-8B, MATH500.

accuracy

0.836

same as the best fixed weights

response length

650 tokens

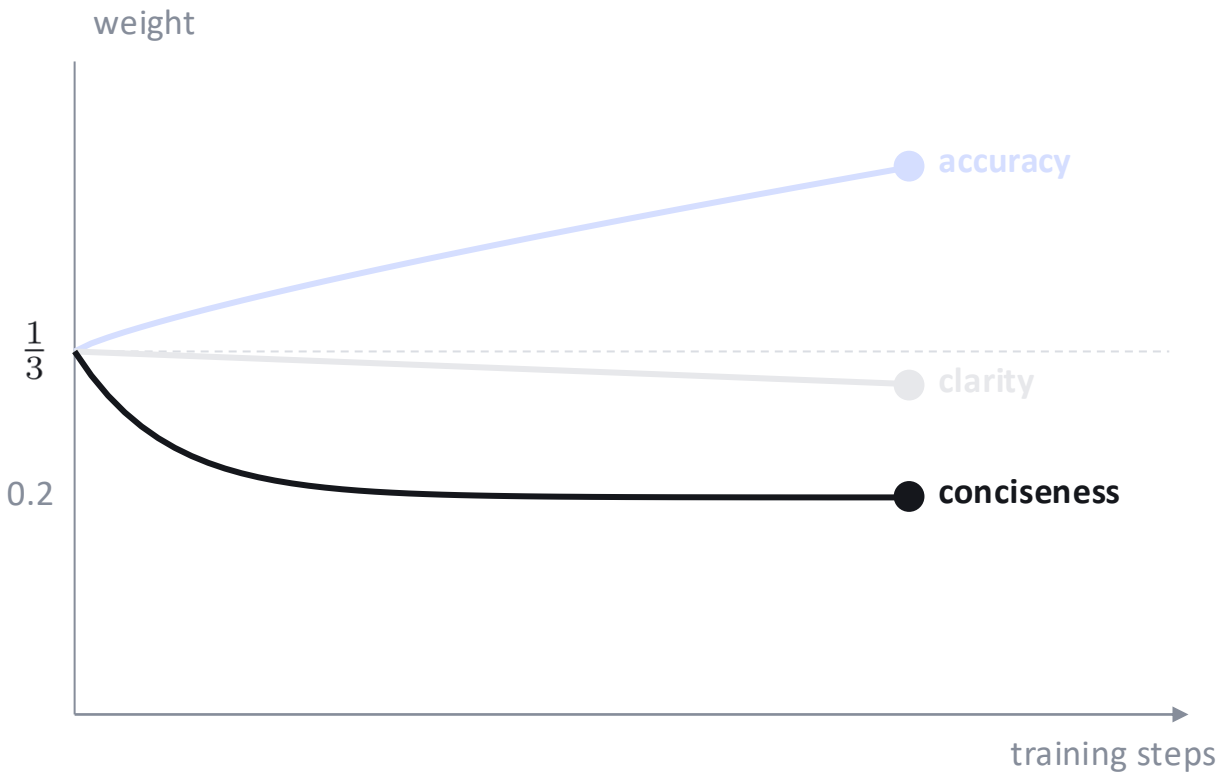
831 to 861 with fixed weights

training steps to reach the front

8 to 9 fewer

with GRPO and RLOO

The learned weights move toward accuracy

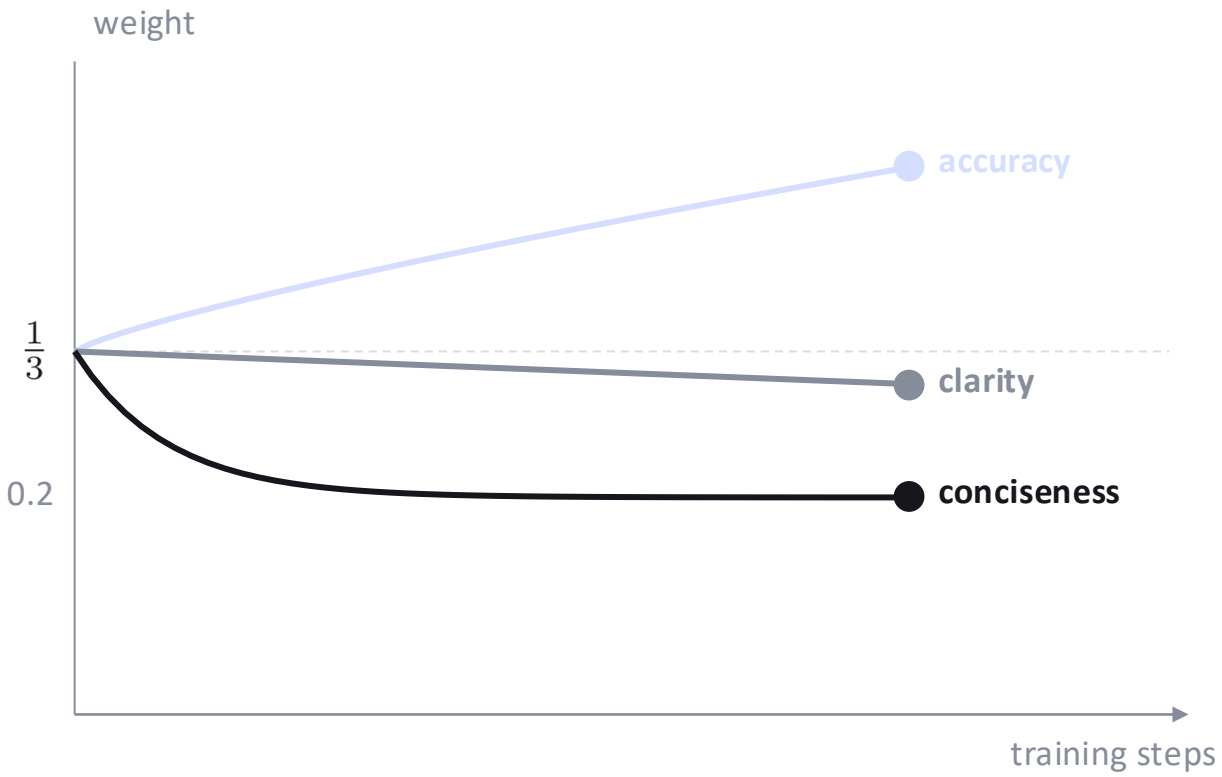


Conciseness is learned early. Its weight drops to about 0.2.

Clarity stays near its starting weight.

Accuracy gains weight for the whole run.

The learned weights move toward accuracy

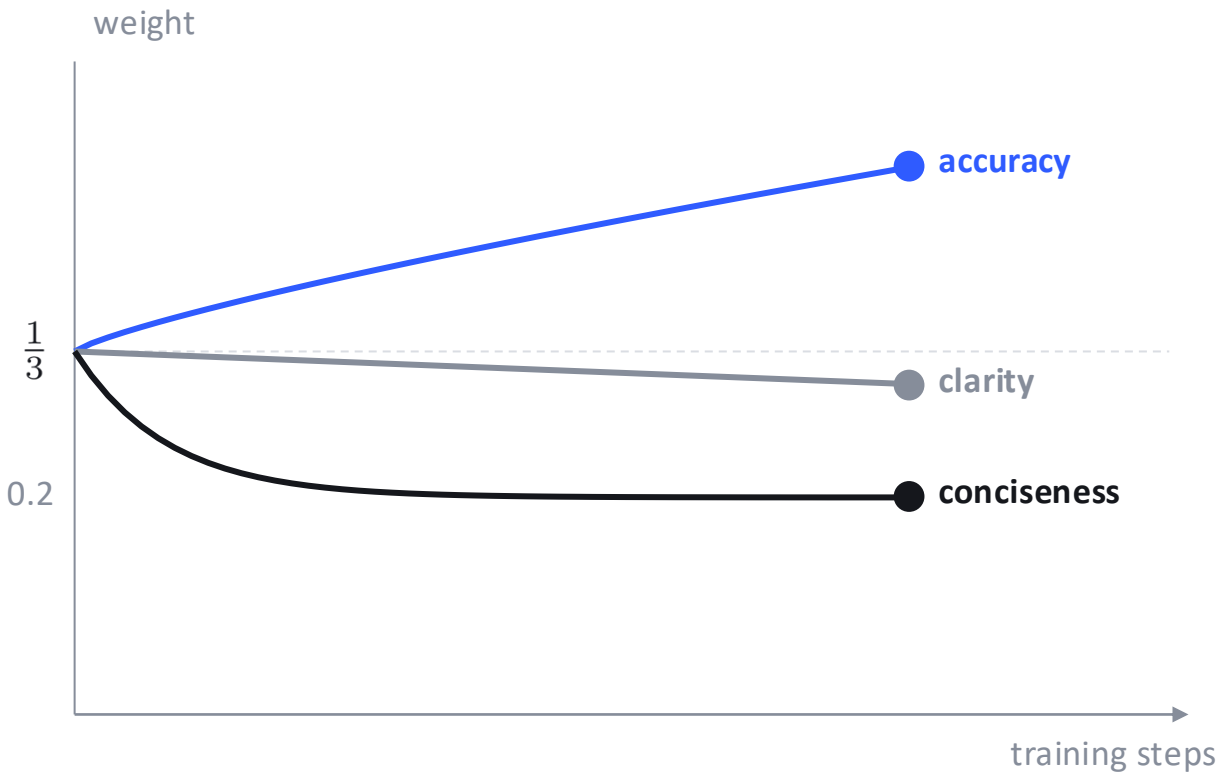


Conciseness is learned early. Its weight drops to about 0.2.

Clarity stays near its starting weight.

Accuracy gains weight for the whole run.

The learned weights move toward accuracy



Conciseness is learned early. Its weight drops to about 0.2.

Clarity stays near its starting weight.

Accuracy gains weight for the whole run.

On some models, every weighting still traded accuracy for brevity.

We saw this with Ministral-8B-Instruct and Llama-3.1-8B-Instruct.

2

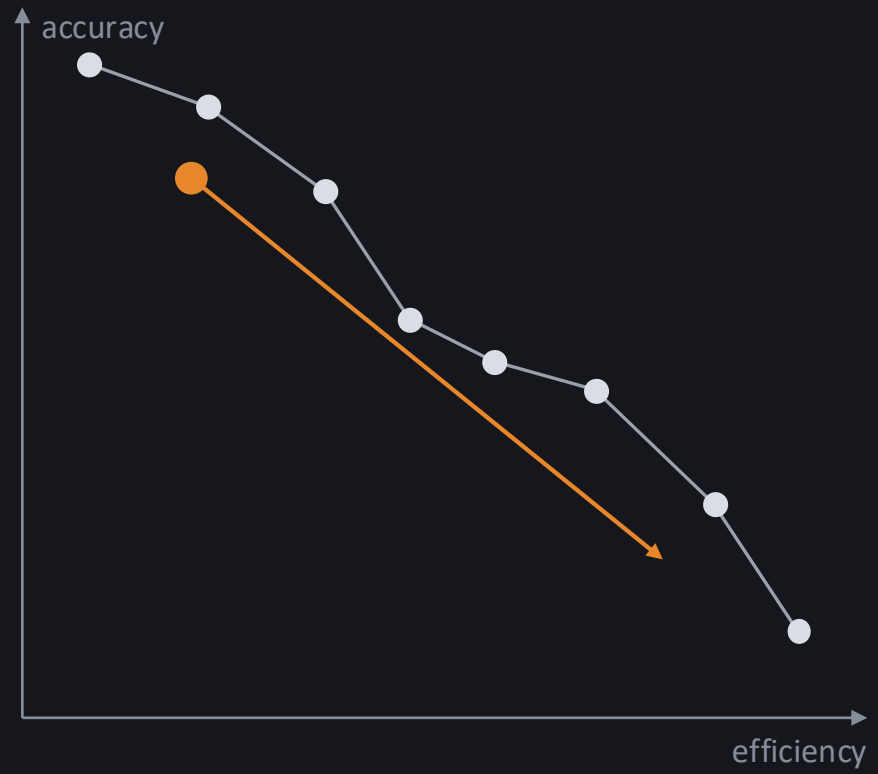
Interference

Why does training sometimes sacrifice one objective for another?

Uncovering Cross-Objective Interference in Multi-Objective Alignment

Yining Lu, Meng Jiang

University of Notre Dame



PART 2: INTERFERENCE

Cross-objective interference

The weighted score keeps going up while at least one objective goes down.

	objective that drops	objective that wins
Math MATH-500	accuracy 0.50 → **0.36**	length (tokens) 989 → **113**
Code MBPP	pass rate 0.57 → **0.42**	length (tokens) 73 → **19**
Summarization XSum	coverage 0.24 → **0.05**	length (tokens) 44 → **12**

Qwen2.5-1.5B models with linear scalarization. Accuracy and pass rate are peak to final.

PART 2: INTERFERENCE

Cross-objective interference

The weighted score keeps going up while at least one objective goes down.

	objective that drops	objective that wins
Math MATH-500	accuracy 0.50 → **0.36**	length (tokens) 989 → **113**
Code MBPP	pass rate 0.57 → **0.42**	length (tokens) 73 → **19**
Summarization XSum	coverage 0.24 → **0.05**	length (tokens) 44 → **12**

Qwen2.5-1.5B models with linear scalarization. Accuracy and pass rate are peak to final.

PART 2: INTERFERENCE

Cross-objective interference

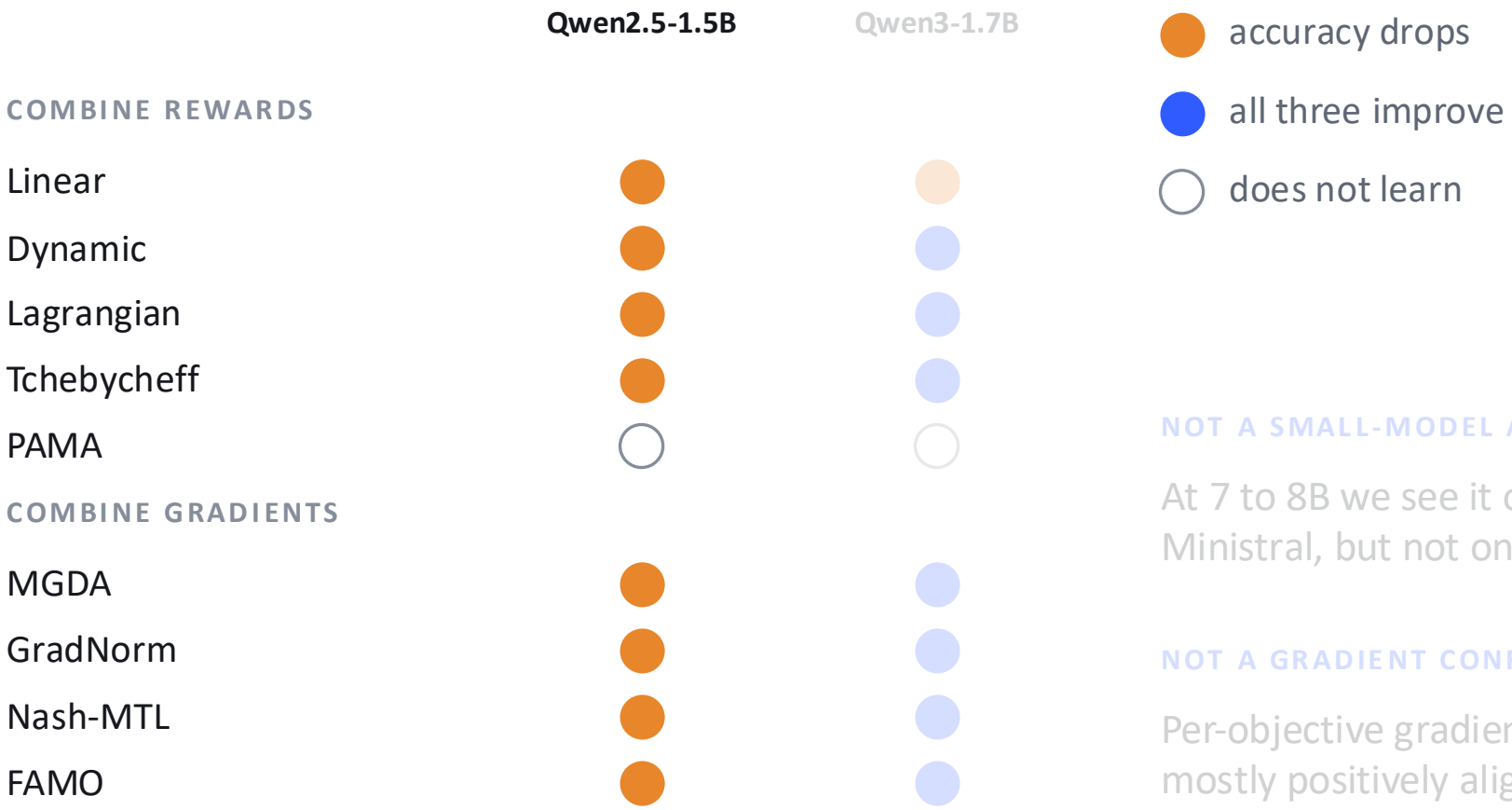
The weighted score keeps going up while at least one objective goes down.

	objective that drops	objective that wins
Math MATH-500	accuracy 0.50 → **0.36**	length (tokens) 989 → **113**
Code MBPP	pass rate 0.57 → **0.42**	length (tokens) 73 → **19**
Summarization XSum	coverage 0.24 → **0.05**	length (tokens) 44 → **23**

Qwen2.5-1.5B models with linear scalarization.

PART 2: INTERFERENCE

Interference across nine methods



accuracy drops

all three improve

does not learn

NOT A SMALL-MODEL ARTIFACT

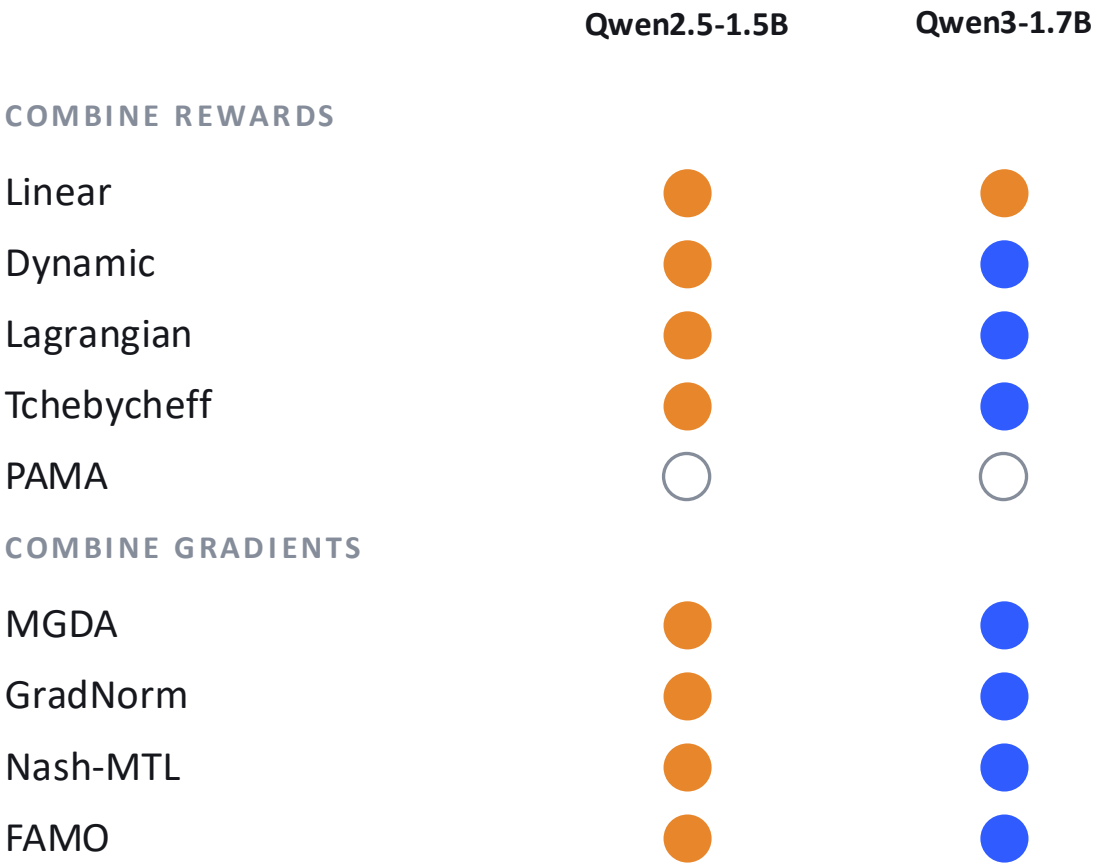
At 7 to 8B we see it on Qwen2.5, Llama-3.1, and Ministral, but not on Qwen3.

NOT A GRADIENT CONFLICT

Per-objective gradients stay mostly positively aligned.

PART 2: INTERFERENCE

Interference across nine methods



accuracy drops

all three improve

does not learn

NOT A SMALL-MODEL ARTIFACT

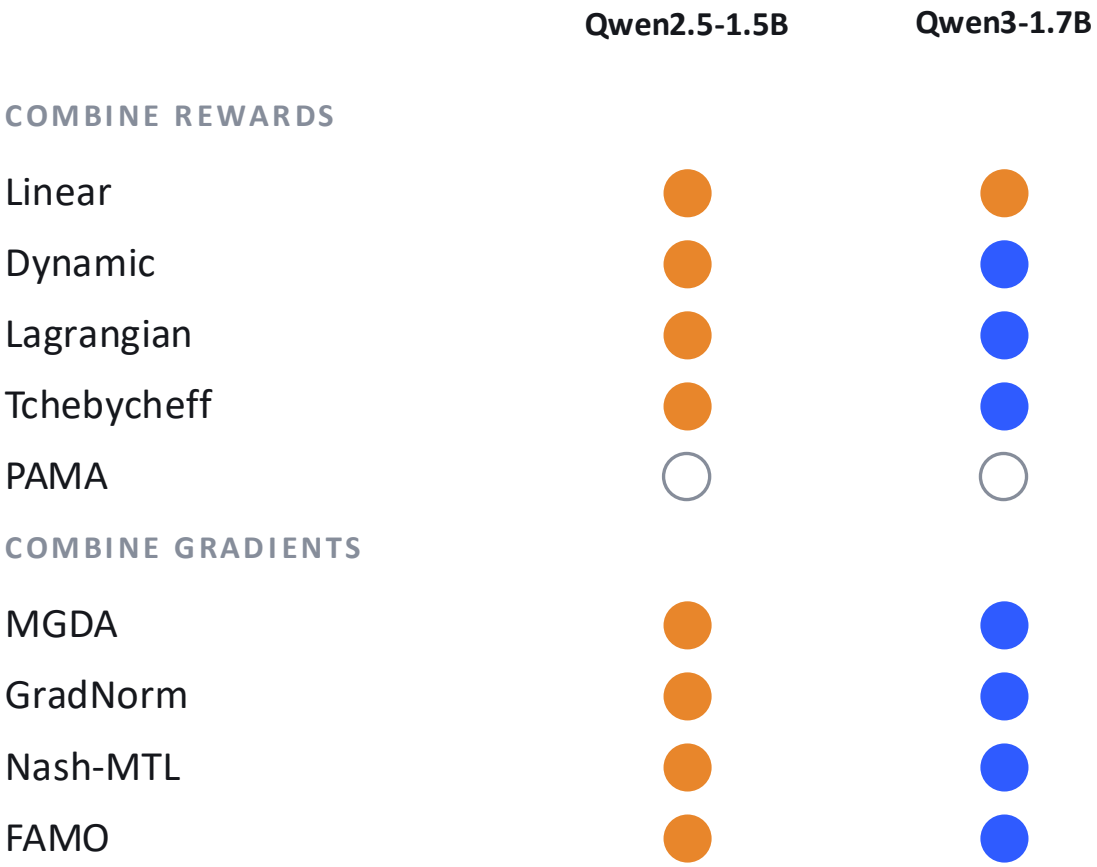
At 7 to 8B we see it on Qwen2.5, Llama-3.1, and Ministral, but not on Qwen3.

NOT A GRADIENT CONFLICT

Per-objective gradients stay mostly positively aligned.

PART 2: INTERFERENCE

Interference across nine methods



accuracy drops

all three improve

does not learn

NOT A SMALL-MODEL ARTIFACT

At 7 to 8B we see it on Qwen2.5, Llama-3.1, and Ministral.

NOT A GRADIENT CONFLICT

Per-objective gradients stay mostly positively aligned.

A local covariance law

$$r_m(\pi^+) - r_m(\pi) \;=\; \eta\, \mathbb{E}_x\big[\, \mathrm{Cov}_{y\sim\pi}(r_m,\, s)\,\big] \;+\; O(\eta^2)$$

change in objective m after one KL-regularized improvement step on the weighted score s

FOR A WEIGHTED SUM

$$\mathrm{Cov}\big(r_m,\, \textstyle\sum_j \lambda_j r_j\big) \;=\; \textstyle\sum_j \lambda_j\, \mathrm{Cov}(r_m,\, r_j)$$

To first order, an objective improves only if it is positively correlated with the score on the model's own samples.

A local covariance law

$$r_m(\pi^+) - r_m(\pi) \;=\; \eta\, \mathbb{E}_x\big[\, \mathrm{Cov}_{y\sim\pi}(r_m,\, s)\,\big] \;+\; O(\eta^2)$$

change in objective m after one KL-regularized improvement step on the weighted score s

FOR A WEIGHTED SUM

$$\mathrm{Cov}\big(r_m,\, \textstyle\sum_j \lambda_j r_j\big) \;=\; \textstyle\sum_j \lambda_j\, \mathrm{Cov}(r_m,\, r_j)$$

To first order, an objective improves only if it is positively correlated with the score on the model's own samples.

A local covariance law

$$r_m(\pi^+) - r_m(\pi) \;=\; \eta\, \mathbb{E}_x\big[\, \mathrm{Cov}_{y\sim\pi}(r_m,\, s)\,\big] \;+\; O(\eta^2)$$

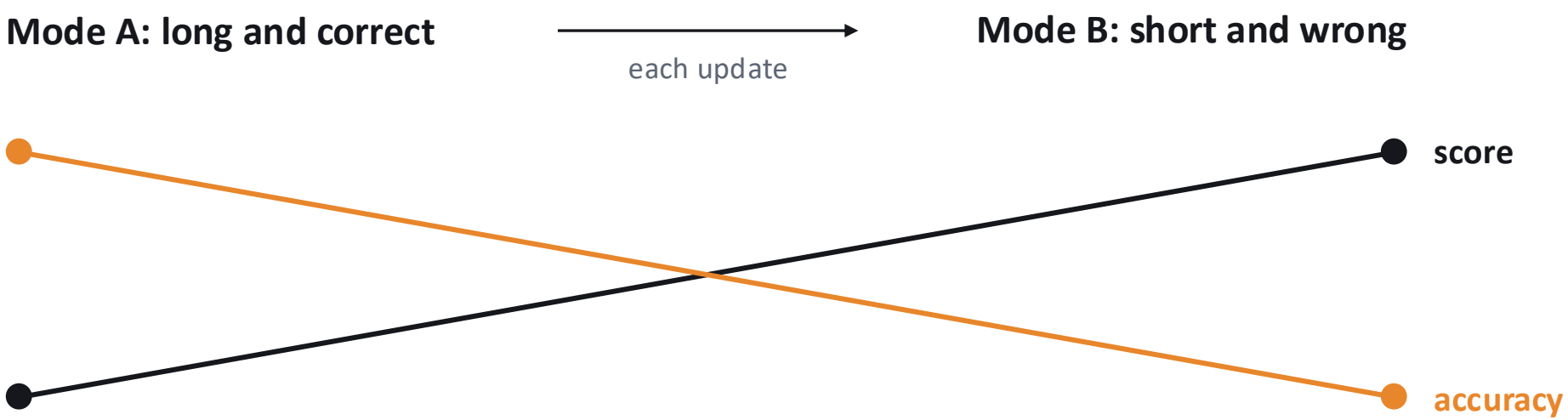
change in objective m after one KL-regularized improvement step on the weighted score s

FOR A WEIGHTED SUM

$$\mathrm{Cov}\big(r_m,\, \textstyle\sum_j \lambda_j r_j\big) \;=\; \textstyle\sum_j \lambda_j\, \mathrm{Cov}(r_m,\, r_j)$$

To first order, an objective improves only if it is positively correlated with the score on the model's own samples.

A two-mode example

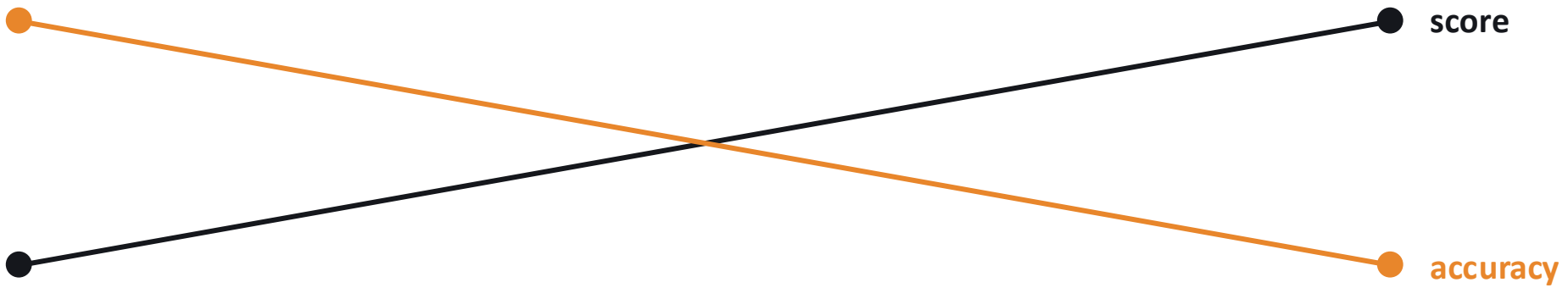


$$\mathrm{Cov}(r_m,\, s) \;=\; \underbrace{p(1-p)}_{>\,0}\;\underbrace{\big[r_m(A) - r_m(B)\big]}_{>\,0}\;\underbrace{\big[s(A) - s(B)\big]}_{<\,0} \;<\; 0$$

A two-mode example

Mode A: long and correct → **Mode B: short and wrong**

each update



$$\mathrm{Cov}(r_m,\, s) \;=\; \underbrace{p(1-p)}_{>0}\; \underbrace{\big[r_m(A) - r_m(B)\big]}_{>0}\; \underbrace{\big[s(A) - s(B)\big]}_{<0} \;<\; 0$$

COVER: a covariance floor for each objective

If an objective's covariance drops below its floor, increase its weight.

Otherwise, leave the weights unchanged.

$$u_m \leftarrow u_m + \eta \left[c_m^* - \bar{c}_m \right]_+, \qquad \lambda_m = e^{u_m}$$

- $\bar{c}_m$ covariance of the objective with the update, averaged over recent steps
- c_m^* its floor
- λ_m its weight in the score

COVER: a covariance floor for each objective

If an objective's covariance drops below its floor, increase its weight.

Otherwise, leave the weights unchanged.

$$u_m \leftarrow u_m + \eta \left[c_m^* - \bar{c}_m \right]_+, \qquad \lambda_m = e^{u_m}$$

$\bar{c}_m$	covariance of the objective with the update, averaged over recent steps
c_m^*	its floor
λ_m	its weight in the score

COVER: a covariance floor for each objective

If an objective's covariance drops below its floor, increase its weight.

Otherwise, leave the weights unchanged.

$$u_m \leftarrow u_m + \eta \left[c_m^* - \bar{c}_m \right]_+, \qquad \lambda_m = e^{u_m}$$

$\bar{c}_m$	covariance of the objective with the update, averaged over recent steps
c_m^*	its floor
λ_m	its weight in the score

It only needs statistics from the current batch, with no per-objective gradients. When objectives already improve together, it is the same as linear scalarization.

PART 2: INTERFERENCE

COVER reduces the accuracy drop

Linear → **COVER**

accuracy drop from peak, Qwen2.5-1.5B

0.149 → **0.101**

accuracy drop from peak, Qwen3-1.7B

0.20 → **0.04**

pass-rate drop, code (MBPP)

0.141 → **0.090**

when objectives already improve together (GSM8K)

within 0.02 of linear

Why does interference depend on the model?

PL CONDITION

$$\frac{1}{2}\|\nabla V\|^2 \geq \mu\left(V^* - V\right)$$

WITH CONSTANT

$$\mu = \frac{1}{2B}\left(\frac{p^*}{1-p^*}\, s^*\, \gamma - 2B\,\sigma_{\max}\right)$$

μ IS SMALL WHEN

p^*	the best completion has low probability
s^*	the best completion has a low score
γ	token-level gradients cancel out

PART 2: INTERFERENCE

Two sources of interference

ALGORITHMIC

Covariance misalignment

MODEL GEOMETRY

A small μ

All objectives improve together only when μ is large and every objective's objective's covariance with the score stays positive.

A trained model still commits to one trade-off.

3

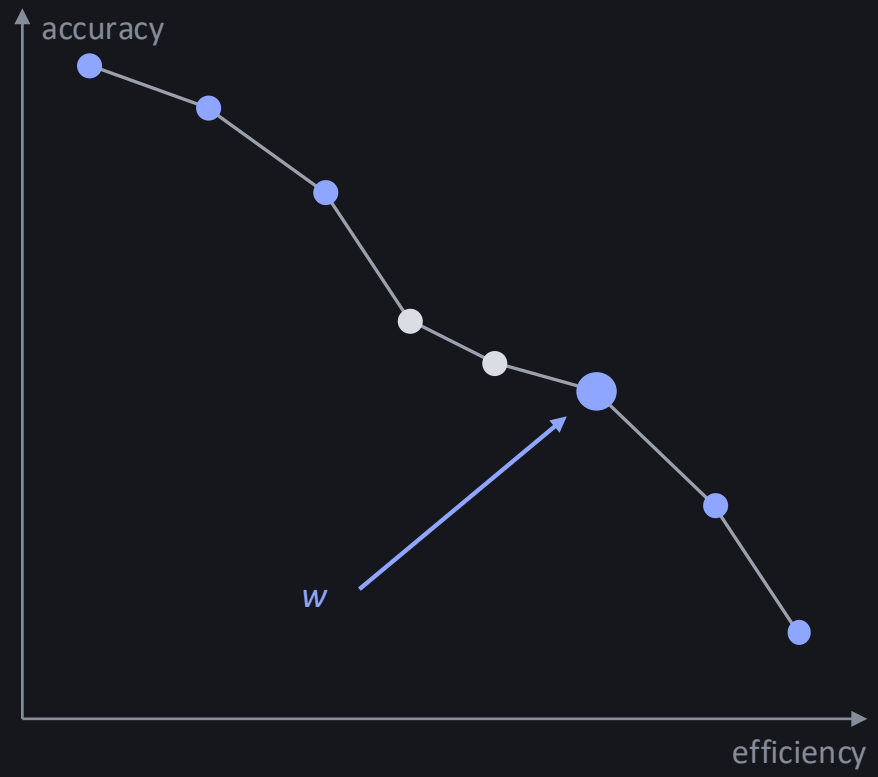
Planning

Can one system serve any trade-off without retraining?

MoFlow: Multi-Objective Agentic Workflow Generation

Yining Lu, Aurelie Lozano, Xi Yang, Naoki Abe, Yu Deng, Meng Jiang

University of Notre Dame and IBM



Agentic workflows with five objectives



Accuracy	Cost	Latency	Robustness	Consistency
task success	tokens per problem	sequential LLM calls	accuracy under paraphrased inputs	accuracy across repeated runs

Current workflow generators optimize accuracy, or accuracy plus cost, and return one workflow per run.

Agentic workflows with five objectives



Accuracy	Cost	Latency	Robustness	Consistency
task success	tokens per problem	sequential LLM calls	accuracy under paraphrased inputs	accuracy across repeated runs

Current workflow generators optimize accuracy, or accuracy plus cost, and return one workflow per run.

PART 3: PLANNING

Agentic workflows with five objectives



Accuracy

task success

Cost

tokens per problem

Latency

sequential LLM calls

Robustness

accuracy under paraphrased inputs

Consistency

accuracy across repeated runs

Current workflow generators optimize accuracy, or accuracy plus cost, and return one workflow per run.

PART 3: PLANNING

One search keeps the whole front

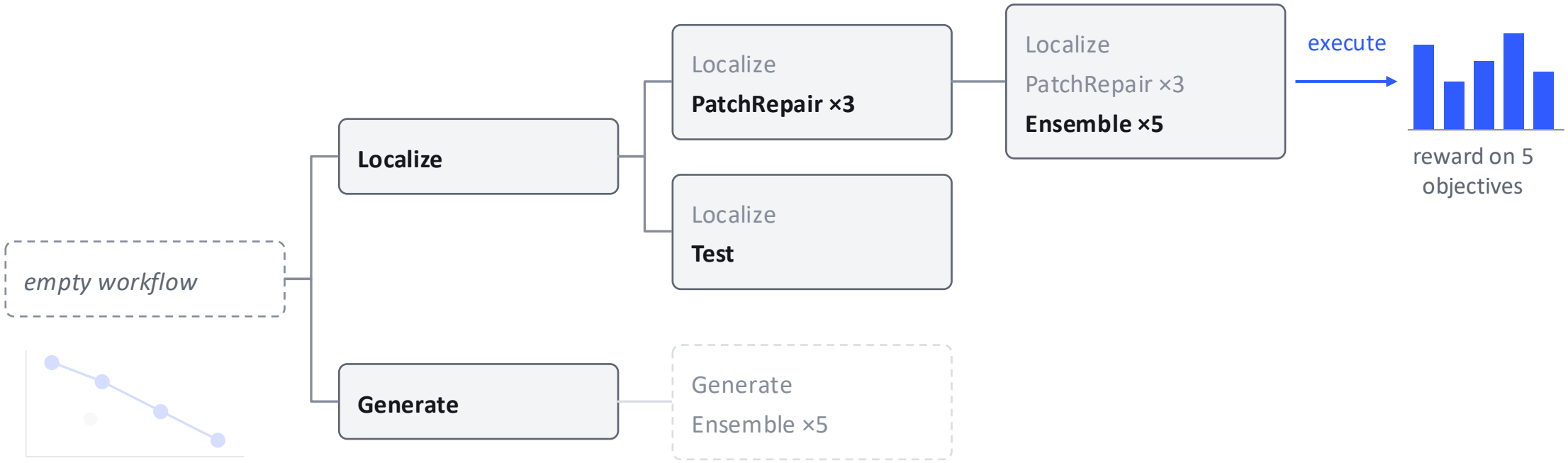


A state is a partial workflow. Each step adds one operator, a branch, or a loop, or stops.

A finished workflow is executed once and scored on all five objectives.

Each node keeps the non-dominated reward vectors below it, not a single value.

One search keeps the whole front

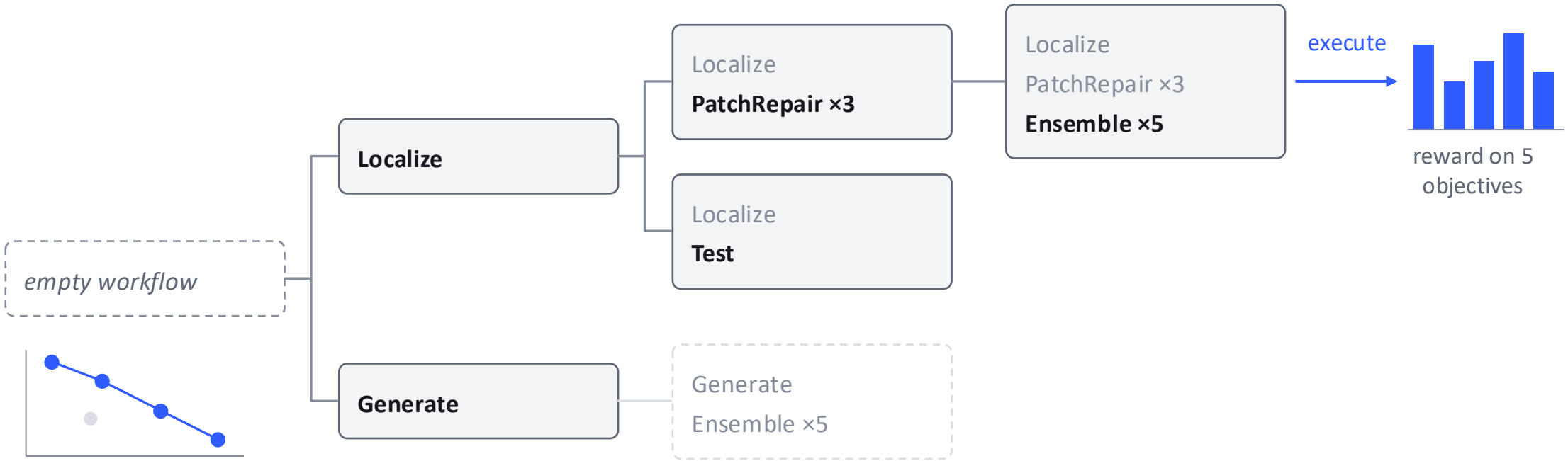


A state is a partial workflow. Each step adds one operator, a branch, or a loop, or stops.

A finished workflow is executed once and scored on all five objectives.

Each node keeps the non-dominated reward vectors below it, not a single value.

One search keeps the whole front

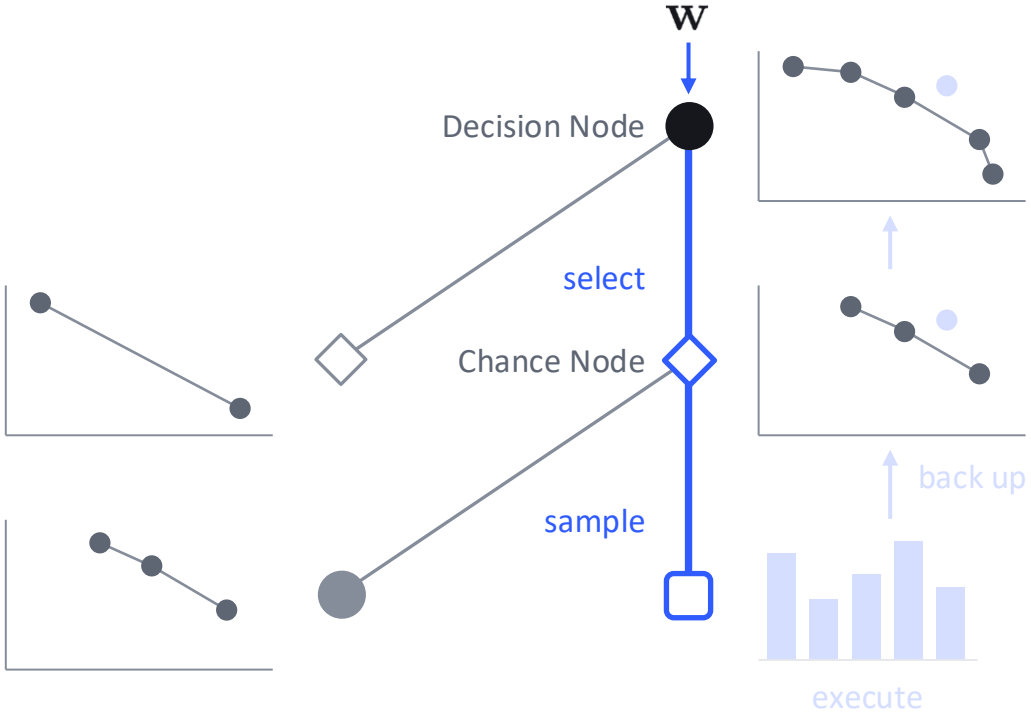


A state is a partial workflow. Each step adds one operator, a branch, or a loop, or stops.

A finished workflow is executed once and scored on all five objectives.

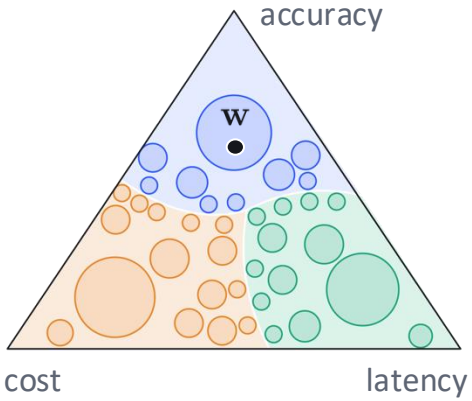
Each node keeps the non-dominated reward vectors below it, not a single value.

Convex Hull Monte Carlo Tree Search



Nothing is scalarized inside the tree.

SELECT: CONTEXTUAL ZOOMING



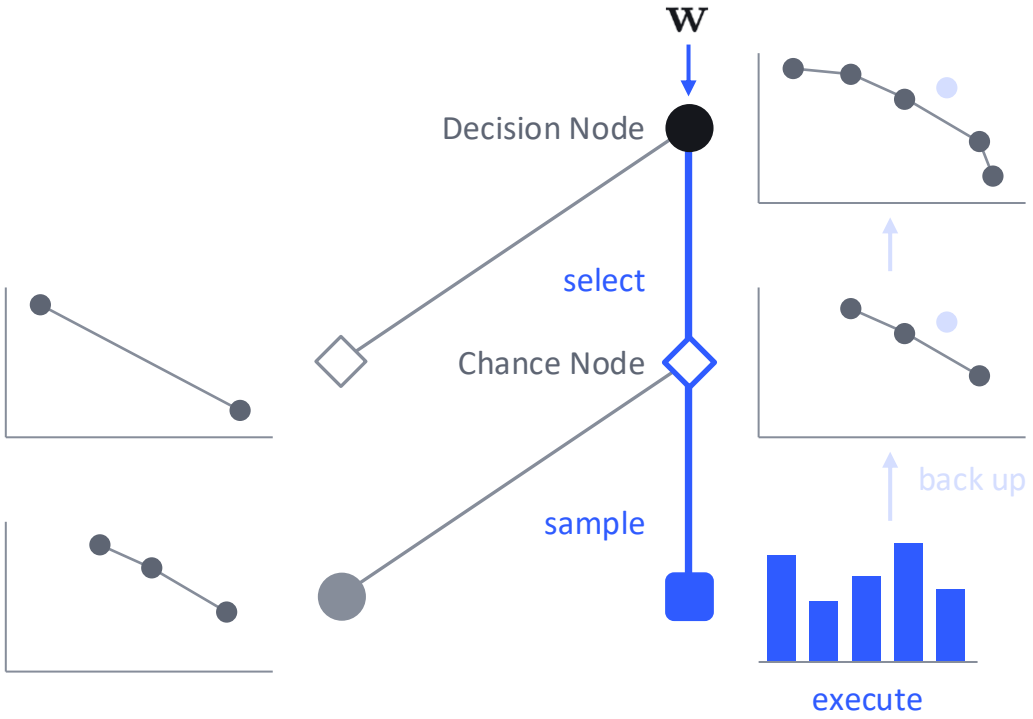
Split only where the best action changes.

BACK UP: UNION, THEN PRUNE

$$\diamond\ \hat{Q}(s,a) = \text{prune}\Big(\bigcup_{s'} [\, \bar{R}(s,a,s') + \hat{V}(s') \,] \Big)$$

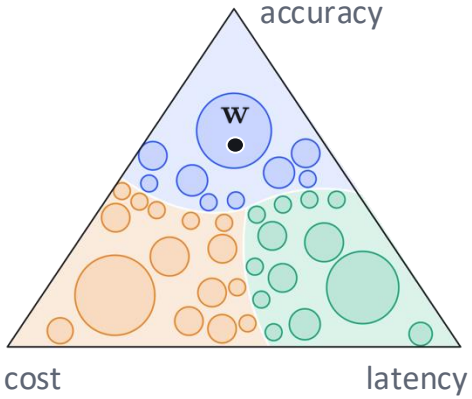
$$\bullet\ \hat{V}(s) = \text{prune}\Big(\bigcup_{a} \hat{Q}(s,a) \Big)$$

Convex Hull Monte Carlo Tree Search



Nothing is scalarized inside the tree.

SELECT: CONTEXTUAL ZOOMING



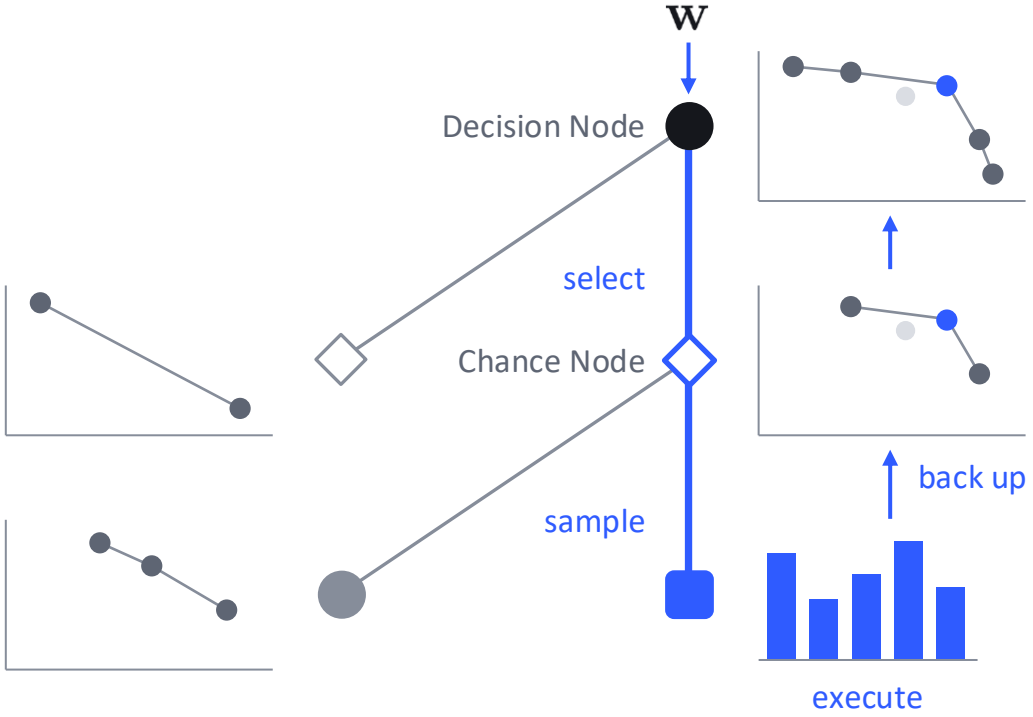
Split only where the best action changes.

BACK UP: UNION, THEN PRUNE

$$\diamond \quad \hat{Q}(s,a) = \text{prune}\Big(\bigcup_{s'} [\, \bar{R}(s,a,s') + \hat{V}(s') \,] \Big)$$

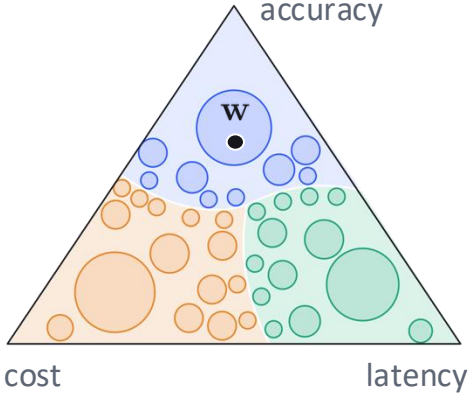
$$\bullet \quad \hat{V}(s) = \text{prune}\Big(\bigcup_{a} \hat{Q}(s,a) \Big)$$

Convex Hull Monte Carlo Tree Search



Nothing is scalarized inside the tree.

SELECT: CONTEXTUAL ZOOMING



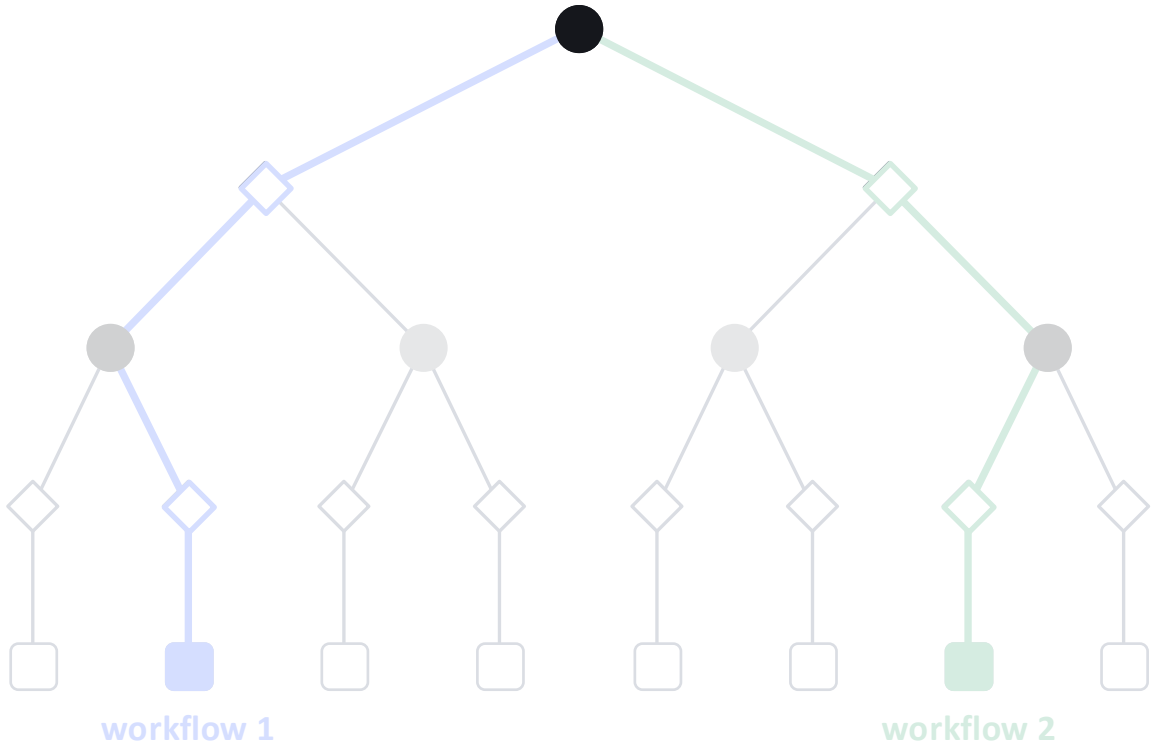
Split only where the best action changes.

BACK UP: UNION, THEN PRUNE

◇ $\hat{Q}(s,a) = \text{prune}\Big(\bigcup_{s'} [\, \bar{R}(s,a,s') + \hat{V}(s') \,] \Big)$

● $\hat{V}(s) = \text{prune}\Big(\bigcup_{a} \hat{Q}(s,a) \Big)$

Serving any preference by lookup



AT A DECISION NODE

$$a^* = \arg\max_a \max_{\mathbf{v} \in \hat{Q}(s,a)} \mathbf{w}^\top \mathbf{v}$$

AT A CHANCE NODE

sample an outcome by visit count

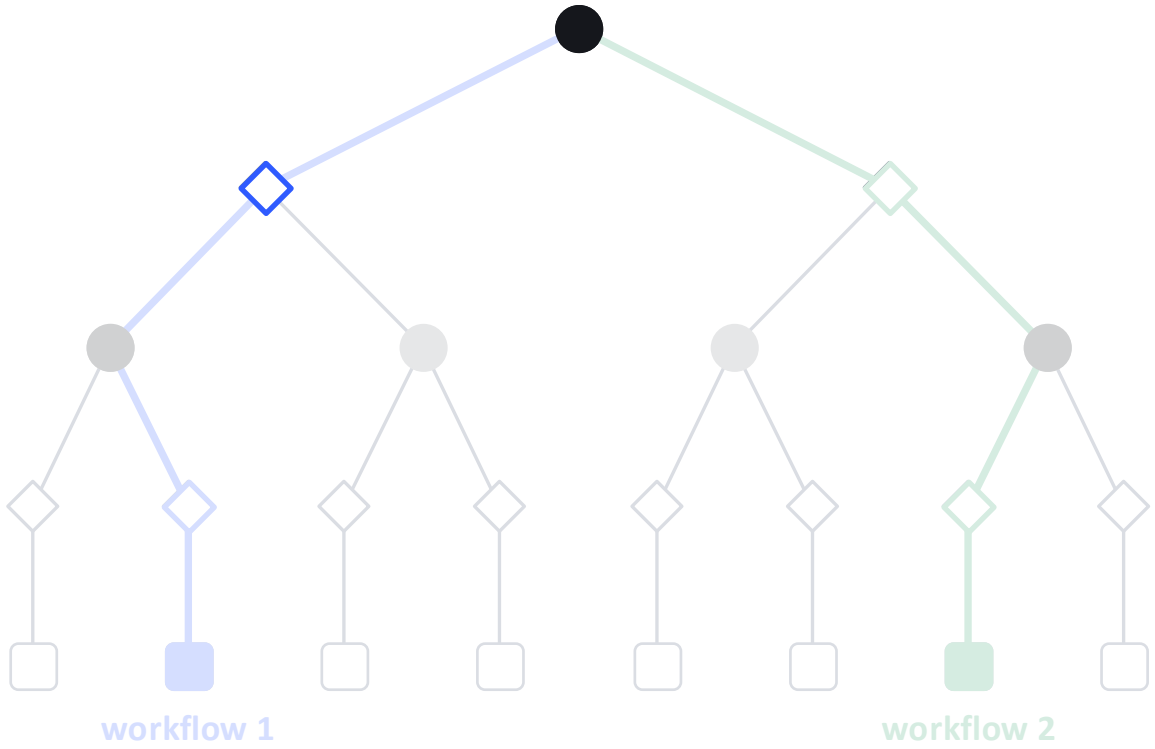
EXAMPLE: SWE-BENCH

$\mathbf{w}_1$ **accuracy first: workflow 1**
branch, vote, repair; 194k tokens

$\mathbf{w}_2$ **cost first: workflow 2**
one patch-repair step; 25k tokens

It needs no LLM calls and no retraining.

Serving any preference by lookup



AT A DECISION NODE

$$a^* = \arg\max_a \max_{\mathbf{v} \in \hat{Q}(s,a)} \mathbf{w}^\top \mathbf{v}$$

AT A CHANCE NODE

sample an outcome by visit count

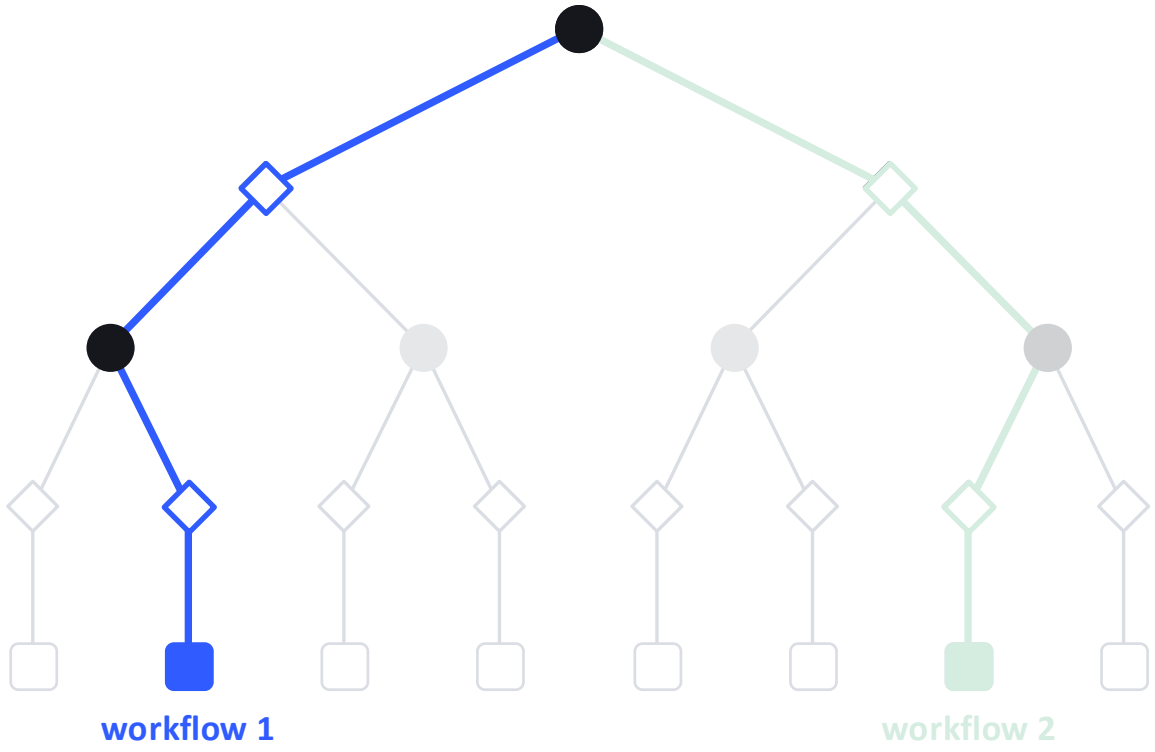
EXAMPLE: SWE-BENCH

$\mathbf{w}_1$ **accuracy first: workflow 1**
branch, vote, repair; 194k tokens

$\mathbf{w}_2$ **cost first: workflow 2**
one patch-repair step; 25k tokens

It needs no LLM calls and no retraining.

Serving any preference by lookup



● AT A DECISION NODE

$$a^* = \arg\max_a \max_{\mathbf{v} \in \hat{Q}(s,a)} \mathbf{w}^\top \mathbf{v}$$

◇ AT A CHANCE NODE

sample an outcome by visit count

EXAMPLE: SWE-BENCH

$\mathbf{w}_1$ **accuracy first: workflow 1**

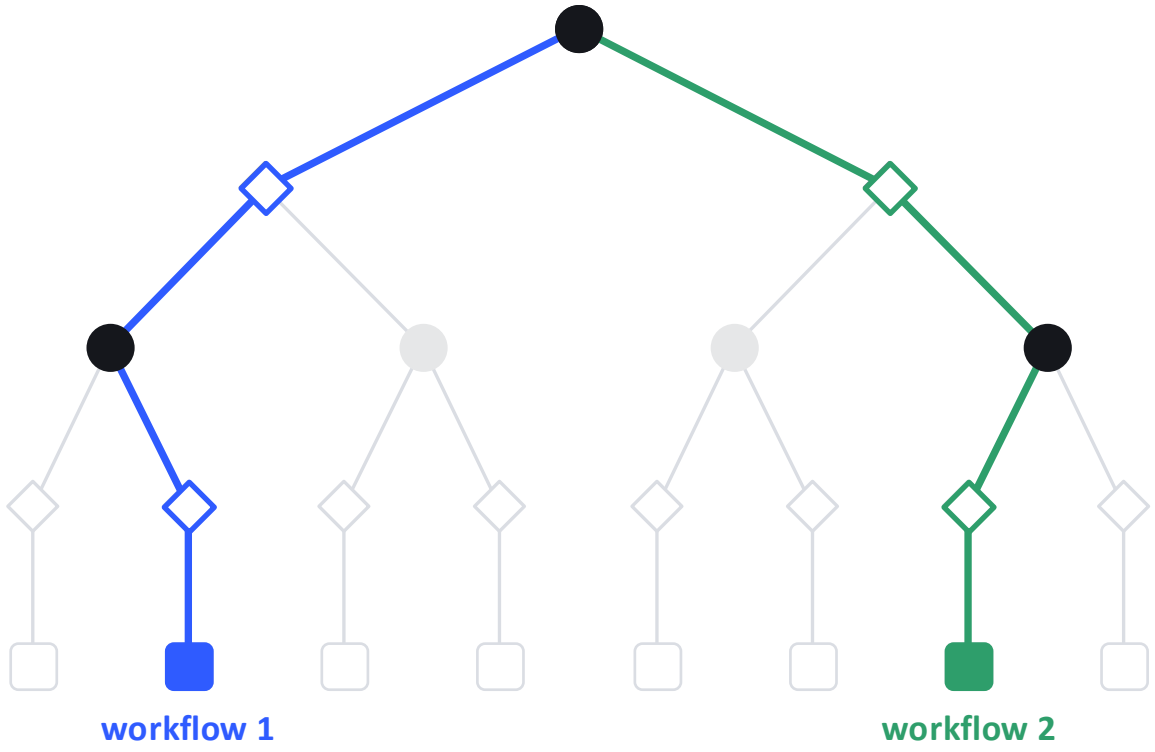
branch, vote, repair; 194k tokens

$\mathbf{w}_2$ **cost first: workflow 2**

one patch-repair step; 25k tokens

It needs no LLM calls and no retraining.

Serving any preference by lookup



● AT A DECISION NODE

$$a^* = \arg\max_{a} \max_{\mathbf{v} \in \hat{Q}(s,a)} \mathbf{w}^\top \mathbf{v}$$

◇ AT A CHANCE NODE

sample an outcome by visit count

EXAMPLE: SWE-BENCH

$\mathbf{w}_1$ **accuracy first: workflow 1**

branch, vote, repair; 194k tokens

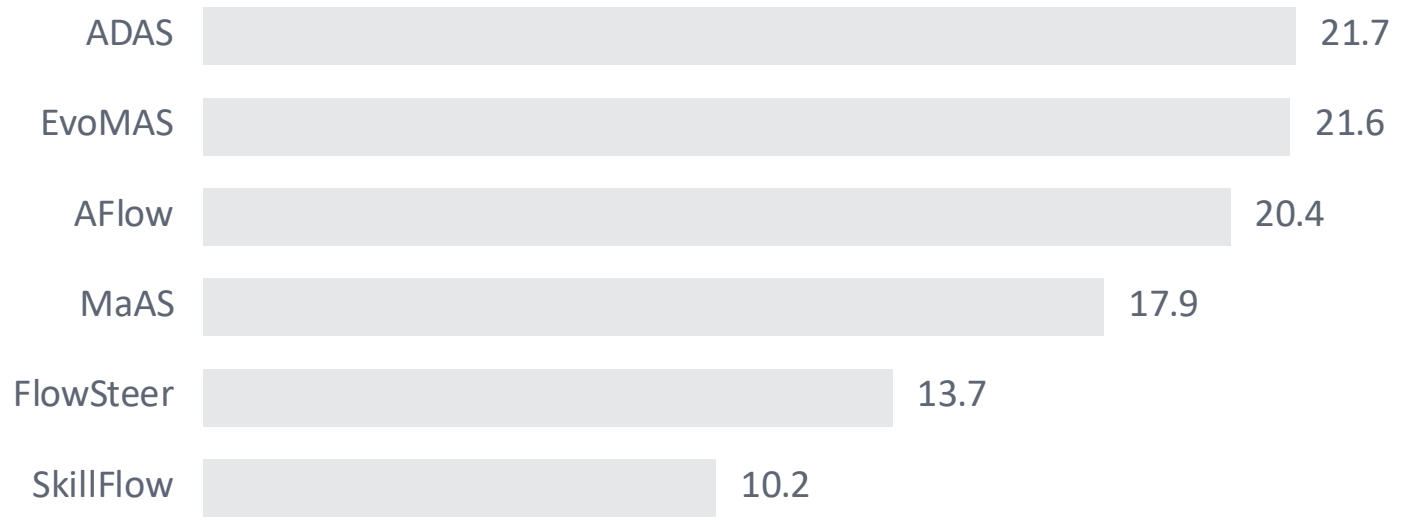
$\mathbf{w}_2$ **cost first: workflow 2**

one patch-repair step; 25k tokens

It needs no LLM calls and no retraining.

PART 3: PLANNING

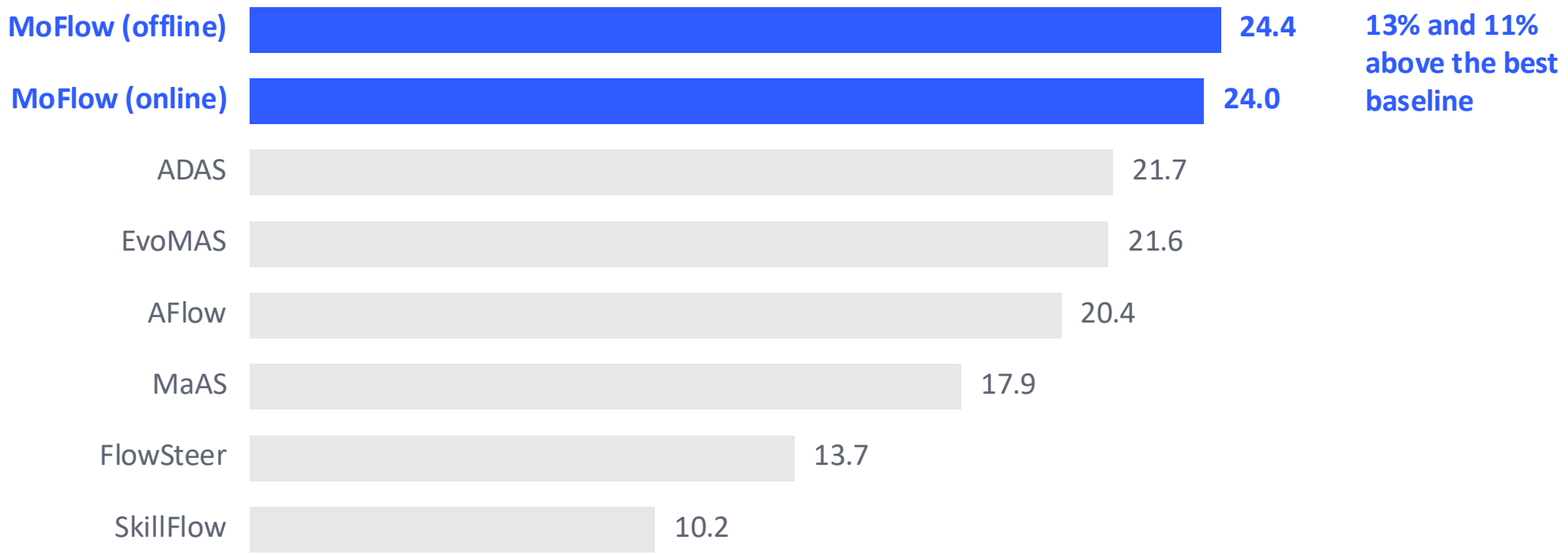
Best coverage of the trade-off space



Mean hypervolume over six benchmarks in math, code, and QA with GPT-5 Mini. Each baseline is adapted to the five objectives and rerun for each of 11 test preferences. MoFlow runs one search and never sees them.

PART 3: PLANNING

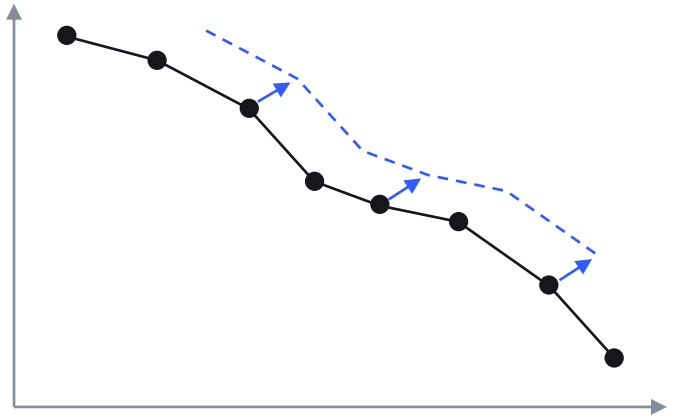
Best coverage of the trade-off space



Mean hypervolume over six benchmarks in math, code, and QA with GPT-5 Mini. Each baseline is adapted to the five objectives and rerun for each of 11 test preferences. MoFlow runs one search and never sees them.

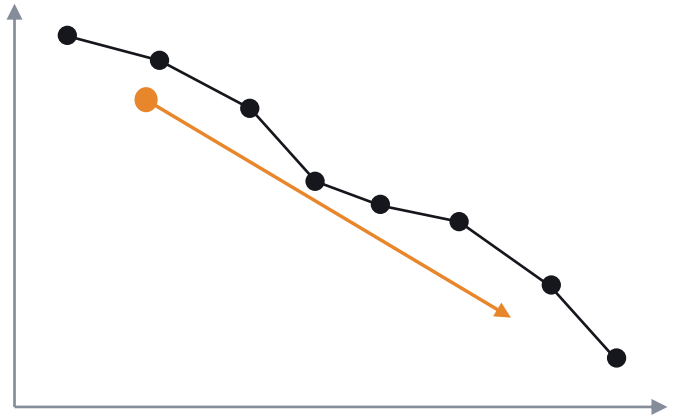
CLOSING

Takeaways



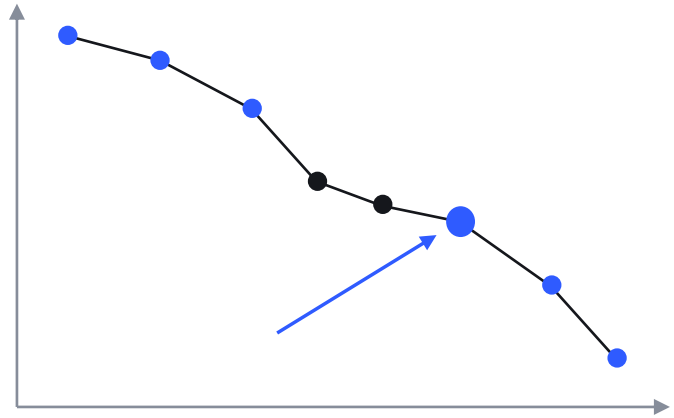
1 Training

Letting the weights follow learning progress gives better fronts in fewer steps.



2 Interference

An objective improves only when it when it covaries with the score. score. Model geometry decides decides whether training converges.



3 Planning

Searching over sets of trade-offs lets one search serve any preference.

Thank you

Yining Lu

University of Notre Dame
https://yining610.github.io

