SAT: Sequential Agent Tuning for Coordinator-Free Plug-and-Play Multi-LLM Training with Monotonic Improvement Guarantees

Yi Xie
Department of Electrical & Computer Engineering, University of Arizona
Tucson, Arizona, USA
yix@arizona.edu

Yangyang Xu
Department of Mathematical Sciences, Rensselaer Polytechnic Institute
Troy, New York, USA
xuy21@rpi.edu

Yi Fan
Amazon Web Services
New York, USA
fnyi@amazon.com

Bo Liu
Department of Electrical & Computer Engineering, University of Arizona
Tucson, Arizona, USA
boliu@arizona.edu

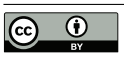
ABSTRACT

Large language models (LLMs) with a large number of parameters achieve strong performance but are often prohibitively expensive to deploy. Recent work explores using teams of smaller, more efficient LLMs that collectively match or even outperform a single large model. However, jointly updating multiple agents introduces compounding distribution shifts, making coordination and stability during training difficult. We address this by introducing Sequential Agent Tuning (SAT), a coordinator-free training paradigm. SAT represents the team as a factorized policy and employs block-coordinate updates over agents, enabling scalable, decentralized training without a central controller. Specifically, we develop a sequence-aware, on-policy advantage estimator that conditions on the evolving team policy, coupled with per-agent KL trust regions that isolate occupancy drift. Theoretically, this framework provides two critical guarantees. First, it ensures monotonic improvement, stabilizing the training process. Second, it establishes provable plug-and-play invariance: any agent can be upgraded to a stronger model without retraining the rest of the team, with a formal guarantee that the performance bound improves. Empirically, a team of three 4B agents (12B total) trained with SAT surpasses the much larger Qwen3-32B on AIME24/25 benchmarks by 3.9% on average. We validate our plug-and-play theory by swapping in two 8B agents, which boosts the composite score by 10.4%. We provide code and appendix of proof at https://github.com/Yydc/SAT-AAMAS

KEYWORDS

Multi-Agent System, LLM, Reinforcement Learning

ACM Reference Format:
Yi Xie, Yangyang Xu, Yi Fan, and Bo Liu. 2026. SAT: Sequential Agent Tuning for Coordinator-Free Plug-and-Play Multi-LLM Training with Monotonic Improvement Guarantees. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/6701030F



Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Author list: Yi Xie, Yangyang Xu, Yi Fan(the work is not related to their position at Amazon Inc), Bo Liu, May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/6701030F

1 INTRODUCTION

Large language models (LLMs) have become effective problem solvers across a wide range of domains [15, 23, 26, 27, 33]. Despite their impressive capabilities, state-of-the-art LLMs demand significant memory, computational, and energy resources, making them unavailable for resource-limited scenarios [1, 16]. This disconnect between potential and deployability motivates a key question: *Can teams of small, efficient models collectively achieve or surpass the performance of one well-tuned large model?*

Recent research has investigated multi-agent LLM systems by employing various coordination strategies, such as role assignment (planner, solver), hierarchical workflow, and ensemble refinement protocols [9, 11, 12, 25, 28, 33]. While these methods show promising empirical results, they are constrained by two main limitations. First, rely on predefined role assignment, which enforces interaction dynamics and may constrain the potential of LLMs systems [35]. Second, existing methods generally lack robust theoretical foundations that guarantee multi-agent LLM systems achieve a strong ability, leaving open questions around why multi-LLMs [23, 25].

We address these gaps through a theoretical analysis of sequential agent training with factorized product policies and per-agent trust regions. We establish three properties under a sequence-aware, on-policy advantage estimator that conditions on intermediate policies. Sequence-agnostic improvement states that stage-wise bounds hold regardless of the agent update sequence chosen before each update within the stage. Plug-and-play invariance ensures that agents can be upgraded via a stage-0 KL projection without retraining others while preserving certificates. Certificate tightening indicates that upgrades can increase surrogate values at fixed radii or achieve comparable gains with smaller radii, thereby reducing the cumulative penalty that scales with the sum of the square roots of the radii. The main challenge, covariate shift arising from sequential updates, is addressed by evaluating the advantages under the current intermediate occupancy and constraining the per-agent per-state KL divergence; we also demonstrate that naive rollout reuse without such conditioning fails to guarantee improvement, whereas our approach yields monotonic bounds.

Empirical evaluation across general reasoning, active reasoning and planning (AutoLogi PlanBench) demonstrates that SAT-trained small-model teams match or exceed large baselines under matched evaluation protocols. These gains align with our order-agnostic stage bounds and certificate tightening under per-agent KL trust regions. We summarize our main contributions as follows:

- We present a theoretical framework for multi-agent LLM finetuning with monotonic improvement bounds, sequence-agnostic guarantees, and plug-and-play invariance (Sec.4).
- We instantiated the theoretical framework through Sequential Agent Tuning (SAT), a coordinator-free training paradigm utilizing sequence-aware block optimization (Sec.5).
- We demonstrate that SAT-trained teams outperform strong baselines on 7 benchmarks, with further improvements achieved through plug-and-play paradigm upgrades (Sec . 6).

2 RELATED WORK

Multi-agent LLM Systems and Orchestration. A line of work explores orchestrating multiple LLMs via roles, tool use, and multi-turn protocols, often with an explicit judge or controller that assigns subtasks or aggregates opinions [34]. Ensembling-style procedures improve reliability by aggregating diverse completions before selection [26]. Structured search over thoughts extends this idea with deliberate branching and pruning at the sequence level [33]. Planning-inspired search further guides generation with MCTS-style rollouts that critique and refine candidates [5, 31]. In contrast, our method is coordinator-free and employs stage-wise sequential updates with per-agent trust regions and an analysis-driven surrogate, yielding provable joint-stage improvement.

Monotonic Policy Improvement and Trust Regions. Conservative Policy Iteration establishes monotonic improvement under carefully controlled policy updates [7]. Trust Region Policy Optimization enforces KL trust regions to stabilize updates with theoretical guarantees [18]. Proximal Policy Optimization adopts a clipped surrogate that approximates trust-region behavior in practice [20]. And some early works applying BCD to RL includes [29, 37]. Our development departs by updating agents sequentially while conditioning analysis on the intermediate product policy, and by providing a joint-stage lower bound where per-agent KL penalties accumulate as $\sum_{i=1}^{n} \sqrt{\delta_i}$, justifying small per-agent trust regions.

Off-Policy Advantage Estimation under Distribution Shift. Generalized Advantage Estimation (GAE) trades bias for variance reduction by utilizing multi-step returns and a value baseline [11, 19]. Retrace introduces safe multi-step targets via truncated importance weights for off-policy data [14, 30]. V-trace extends this idea to scalable actor–critic training with robust corrections [4]. In LLM pipelines, iterative procedures such as self-consistency can shift the data distribution across rounds [26]. Sequence-level search similarly modifies the sampling distribution as the tree expands and prunes candidates [33]. Our estimator conditions on the updated intermediate policy and uses clipped multi-step ratios to manage bias, leveraging sequence information gathered within a stage.

Dynamic Routing and Plug-and-Play Compute. Cascaded inference routes easier inputs to cheaper models and escalates only when necessary, reducing latency without sacrificing accuracy [8]. Previous multi-agent frameworks may swap tools or models, but they rarely provide guarantees that persist after replacement. Our plug-and-play analysis shows that monotonic improvement certificates remain invariant under agent replacement, provided that the surrogate and trust-region constraints are respected.

3 PRELIMINARIES

Environment and policies. Let $\mathcal{M} = (\mathcal{S}, \{\mathcal{A}_i\}_{i=1}^{n}, P, r, \gamma)$ be a discounted MDP with $\gamma \in (0,1)$ and bounded rewards $|r| \le R_{\max}$. The joint action space is the product $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$. A team of n execution agents forms a factorized policy over *joint actions*

$$\pi(a \mid s) = \prod_{i=1}^{n} \pi^{(i)}(a_i \mid s), \qquad a = (a_1, \ldots, a_n) \in \mathcal{A}.$$

Thus $\pi(\cdot \mid s)$ is a valid distribution on $\mathcal{A}$, not n agents "taking the same action." Let d^π denote the discounted state visitation measure:

$$J(\pi) = \mathbb{E}_{\mu, s_0 \sim \mu, \pi}\Big[\sum_{t \ge 0} \gamma^t r_t\Big] = \frac{1}{1-\gamma}\, \mathbb{E}_{s \sim d^\pi,\, a \sim \pi(\cdot|s)}[\, r(s,a)\,].$$

In practice, some tasks activate only a subset of agents at a given state (for example, token heads, tool routers, or role-switching schedulers). To faithfully cover both simultaneous and interleaved execution while preserving the product-policy analysis, we adopt a masked activation view: at each state s, let $\mathcal{I}(s) \subseteq \{1, \ldots, n\}$ denote the set of active agents and interpret the joint policy over active heads as $\pi(a \mid s) = \prod_{i \in \mathcal{I}(s)} \pi^{(i)}(a_i \mid s)$, while inactive heads take a fixed no-op. All per-state divergences and trust-region constraints below are evaluated over the active heads at that state; when all agents act simultaneously, $\mathcal{I}(s) = \{1, \ldots, n\}$ and the definitions reduce to the standard full-product case. This formalization of masking is consistent with our later sequence-level objective and does not alter the statements that only depend on adjacent intermediate policies. If an implementation never uses masked activation, setting $\mathcal{I}(s)$ to the full set recovers the original expressions.

We write Q^π, V^π for action/value functions and $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$. We update agents sequentially in order $\sigma(1), \ldots, \sigma(n)$:

$$\hat{\pi}^0 = \pi_{\text{cur}}, \;\; \hat{\pi}^i = (\pi_{\text{tar}}^{\sigma(1)}, \ldots, \pi_{\text{tar}}^{\sigma(i)}, \pi_{\text{cur}}^{\sigma(i+1)}, \ldots, \pi_{\text{cur}}^{\sigma(n)}), \;\; \bar{\pi} = \hat{\pi}^n.$$

Per-state divergences and trust regions. For policies π_1, π_2 define the per-state maximal divergences

$$D_{\text{KL}}^{\max}(\pi_1 \| \pi_2) = \sup_s D_{\text{KL}}\big(\pi_1(\cdot \mid s) \| \pi_2(\cdot \mid s)\big),$$
$$D_{\text{TV}}^{\max}(\pi_1 \| \pi_2) = \sup_s \tfrac{1}{2} \|\pi_1(\cdot \mid s) - \pi_2(\cdot \mid s)\|_1.$$

and use Pinsker's inequality to link them: $D_{\text{TV}}^{\max} \le \sqrt{\frac{1}{2} D_{\text{KL}}^{\max}}$. At step i, agent $\sigma(i)$ obeys the per-state trust region

$$D_{\text{KL}}^{\max}\big(\pi_{\text{tar}}^{\sigma(i)} \| \pi_{\text{cur}}^{\sigma(i)}\big) \le \delta_i \quad \text{(uniform case: } \delta_i \equiv \delta\text{)}.$$

We also define $A_{\max} := \sup_{s,a} |A^\pi(s,a)| \le \frac{2R_{\max}}{1-\gamma}$ for later bounds.

Sequence-aware surrogate objective. When updating agent $\sigma(i)$ we evaluate on-policy under the current intermediate occupancy:

$$L_i^{\text{SEQ}}(\pi_{\text{tar}}^{\sigma(i)}) = \frac{1}{1-\gamma}\, \mathbb{E}_{s \sim d^{\hat{\pi}^{i-1}},\, a \sim \hat{\pi}^i(\cdot|s)}\big[\widehat{A}_{\text{ON}}^{i-1}(s,a)\big],$$

where $\widehat{A}_{\text{ON}}^{i-1}$ is a multi-step on-policy estimator computed from trajectories of $\hat{\pi}^{i-1}$ (for example, GAE with trace λ). This evaluates

full episodes under the intermediate policy and is consistent with masked activation, as interpreted by considering the joint action as the product of the active heads at each visited state.

Two standard tools. The performance difference lemma states that $J(\pi') - J(\pi) = \frac{1}{1-\gamma}\,\mathbb{E}_{s\sim d^{\pi'},\,a\sim\pi'}[A^{\pi}(s,a)]$. An occupancy-shift bound for bounded test functions f is given by $\left|\mathbb{E}_{d^{\pi'}}[f] - \mathbb{E}_{d^{\pi}}[f]\right| \le \frac{2\gamma}{1-\gamma}\, D_{\mathrm{TV}}^{\max}(\pi'\|\pi)\,\|f\|_{\infty}$.

4 THEORETICAL FRAMEWORK

We study a coordinator-free, plug-and-play training paradigm in which a team of n execution agents is updated sequentially via block-coordinate ascent. The main technical challenge is distribution shift: when one agent updates, the effective evaluation distribution for subsequent agents changes. We address this issue with a sequence-aware, on-policy advantage estimator that conditions on the current intermediate team policy, along with per-agent. These per-state KL trust regions limit occupancy drift. These ingredients yield single-step and joint-stage monotonic improvement bounds, sequence-agnostic guarantees compatible with learned schedulers and plug-ins, and information-theoretic envelopes that quantify what is achievable under a fixed sampling budget. Full proofs are presented in the Appendix, with a table that maps the results.

4.1 Main Theoretical Guarantee

Statement (one stage, sequence-agnostic, plug-and-play). For a full training stage of n sequential agent updates with per-agent, per-state KL radii $\{\delta_i\}_{i=1}^n$ and on-policy budgets $\{N_i\}_{i=1}^n$, we provide a high-probability certificate of monotonic improvement. In particular, with probability at least $1-\delta_{\mathrm{conf}}$,

$$J(\bar\pi) - J(\pi_{\mathrm{cur}}) \;\ge\; \underbrace{\sum_{i=1}^{n}\left(\kappa_i^{\mathrm{reg}}\sqrt{\delta_i} - a_i^{\mathrm{reg}}\delta_i\right)}_{\text{Information-Geometric Gain}} - \underbrace{\frac{2\gamma}{(1-\gamma)^2}A_{\max}\sum_{i=1}^{n}\sqrt{\tfrac{1}{2}\,\delta_i}}_{\text{Occupancy-Shift Penalty}}$$
$$- \underbrace{\frac{1}{1-\gamma}\sum_{i=1}^{n}\zeta_i}_{\text{Estimator-Bias Penalty}} - \underbrace{\sum_{i=1}^{n}\frac{A_{\max}}{1-\gamma}\sqrt{\frac{\log(2n/\delta_{\mathrm{conf}})}{2N_i}}}_{\text{Finite-Sample Error}}.$$

where the gain coefficients $\kappa_i^{\mathrm{reg}} = \sqrt{2\, g_i^{\top}(F_i^{\mathrm{reg}})^{-1} g_i}$ and curvature terms $a_i^{\mathrm{reg}} = L_i^{\mathrm{loc}}/\lambda_{\min}(F_i^{\mathrm{reg}})$ are defined in Theorem 4.7. The three penalty terms have precise interpretations: the *occupancy-shift penalty* $\frac{2\gamma}{(1-\gamma)^2}A_{\max}\sum_{i=1}^n\sqrt{\frac{1}{2}\,\delta_i}$ captures the cumulative cost of distribution shift as agents update sequentially (Theorem 4.4); the *estimator-bias penalty* $\frac{1}{1-\gamma}\sum_{i=1}^n\zeta_i$ arises from our sequence-aware advantage estimator (Theorem 4.2); and the *finite-sample error* $\sum_{i=1}^n\frac{A_{\max}}{1-\gamma}\sqrt{\frac{\log(2n/\delta_{\mathrm{conf}})}{2N_i}}$ accounts for statistical uncertainty from using N_i samples per step (Theorem 4.9).

Crucially, this policy improvement guarantee is realized via a provably convergent optimization procedure. The sequential projected block updates on the stage surrogate $G(\theta) = \sum_i L_i^{\mathrm{SEQ}}$ satisfy the standard $O(1/K)$ convergence rate for the projected-gradient mapping over K block steps (Theorem 4.8). All statements are *sequence-agnostic* in σ and remain valid under *plug-and-play* replacements, provided the new agent is initialized within the same per-state KL trust region (via Stage-0 alignment in the Appendix).

Remark (High-level interpretation). The inequality above certifies that the *information-geometric gain*—scaled by natural-gradient structure through $\sqrt{\delta_i}$ terms—minus three *controllable costs* (distribution shift, estimator bias, and sampling noise) remains nonnegative in aggregate. At the same time, the optimizer reliably finds such updates at a rate $O(1/K)$. Proof ingredients are modular: occupancy shift via per-state TV/KL bounds (§4.2), single-step $\Rightarrow$ joint-stage telescoping (Theorems 4.2–4.4), information-geometric lower bounds for $\kappa_i^{\mathrm{reg}}, a_i^{\mathrm{reg}}$ (Theorem 4.7), finite-sample concentration (Theorem 4.9), and sequential projected-gradient convergence (Theorem 4.8). For tighter certificates, one may replace δ_i by any expected-KL radius $\bar\delta_i \le \delta_i$ in the gain terms (Theorem 4.7).

4.2 Occupancy Shift: definition and per-state control

Definition (discounted occupancy shift). For two policies π' and π, the discounted occupancy shift measures how much their state visitation distributions differ:

$$\Delta_{\mathrm{occ}}(\pi',\pi;f) \;\triangleq\; \left|\mathbb{E}_{d^{\pi'}}[f] - \mathbb{E}_{d^{\pi}}[f]\right|, \qquad \Delta_{\mathrm{occ}}(\pi',\pi) \;\triangleq\; \sup_{\|f\|_\infty\le 1}\Delta_{\mathrm{occ}}(\pi',\pi;f).$$

Lemma 4.1 (Occupancy-shift bound). *With per-state divergences* $D_{\mathrm{TV}}^{\max}(\pi'\|\pi) \triangleq \operatorname{ess\,sup}_s D_{\mathrm{TV}}\big(\pi'(\cdot|s),\pi(\cdot|s)\big)$ *and* $D_{\mathrm{KL}}^{\max}(\pi'\|\pi) \triangleq \operatorname{ess\,sup}_s D_{\mathrm{KL}}\big(\pi'(\cdot|s)\|\pi(\cdot|s)\big)$*, one has the worst-case bound*

$$\Delta_{\mathrm{occ}}(\pi',\pi) \;\le\; \frac{2\gamma}{1-\gamma}\, D_{\mathrm{TV}}^{\max}(\pi'\|\pi) \;\le\; \frac{2\gamma}{1-\gamma}\sqrt{\tfrac{1}{2}\, D_{\mathrm{KL}}^{\max}(\pi'\|\pi)}.$$

When plugged into the performance-difference identity (which carries an extra factor $1/(1-\gamma)$), this yields a *single-step penalty* $\frac{2\gamma}{(1-\gamma)^2}\,A_{\max}\sqrt{\frac{1}{2}\,D_{\mathrm{KL}}^{\max}}$ in Theorem 4.2.

Per-state control (our remedy). At sequential step i, we *cap* the per-state KL between the updated agent and its current version:

$$D_{\mathrm{KL}}\big(\pi_{\mathrm{tar}}^{\sigma(i)}(\cdot|s)\,\big\|\,\pi_{\mathrm{cur}}^{\sigma(i)}(\cdot|s)\big) \;\le\; \delta_i(s) \quad \forall s.$$

This implies $D_{\mathrm{KL}}^{\max}\big(\hat\pi^{\,i}\|\hat\pi^{\,i-1}\big) \le \delta_i^{\max} := \operatorname{ess\,sup}_s \delta_i(s)$ and therefore

$$\Delta_{\mathrm{occ}}\big(\hat\pi^{\,i},\hat\pi^{\,i-1}\big) \;\le\; \frac{2\gamma}{1-\gamma}\sqrt{\tfrac{1}{2}\,\delta_i^{\max}}.$$

Consequently, the only distribution-shift cost entering our single-step and joint-stage guarantees is the explicit $\sqrt{\delta_i}$ penalty (Theorems 4.2 and 4.4), which we control directly by choosing the per-state radii $\{\delta_i(s)\}$ (and enforcing them via trust-region updates / Stage-0 KL projection).

4.3 Monotonic Improvement Guarantees

Theorem 4.2 (Single-step monotonic improvement). *For step* $i \in \{1,\dots,n\}$*,*

$$J(\hat\pi^{\,i}) - J(\hat\pi^{\,i-1}) \;\ge\; L_i^{SEQ}(\pi_{\mathrm{tar}}^{\sigma(i)}) - \frac{2\gamma}{(1-\gamma)^2}\,A_{\max}\sqrt{\tfrac{1}{2}\,D_{\mathrm{KL}}^{\max}\big(\hat\pi^{\,i}\|\hat\pi^{\,i-1}\big)} - \frac{\zeta_i}{1-\gamma}.$$

Since $A_{\max} \le \frac{2R_{\max}}{1-\gamma}$*, the penalty equals* $\frac{4\gamma R_{\max}}{(1-\gamma)^3}\sqrt{\frac{1}{2}\,D_{\mathrm{KL}}^{\max}\big(\hat\pi^{\,i}\|\hat\pi^{\,i-1}\big)}$.

Proof sketch. The performance difference lemma gives

$$J(\hat{\pi}^{\,i}) - J(\hat{\pi}^{\,i-1}) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\hat{\pi}^{\,i}},a\sim\hat{\pi}^{\,i}(\cdot|s)}[A^{\hat{\pi}^{\,i-1}}(s,a)].$$

Apply the Lemma 4.1 above to replace $d^{\hat{\pi}^{\,i}}$ by $d^{\hat{\pi}^{\,i-1}}$ (TV/KL penalty). Then replace $A^{\hat{\pi}^{\,i-1}}$ by the on-policy estimator and account for ζ_i. Full details are in the Appendix (Single-step). □

Lemma 4.3 (KL accumulation for product policies). *Let* $p(\mathbf{a}|s) = \prod_{j=1}^{n} p_j(a^{(j)}|s)$ *and* $q(\mathbf{a}|s) = \prod_{j=1}^{n} q_j(a^{(j)}|s)$. *Then*

$$D_{\mathrm{KL}}\big(p(\cdot|s)\|q(\cdot|s)\big) = \sum_{j=1}^{n} D_{\mathrm{KL}}\big(p_j(\cdot|s)\|q_j(\cdot|s)\big).$$

At step i, $\hat{\pi}^{\,i}$ *and* $\hat{\pi}^{\,i-1}$ *differ only in factor* $\sigma(i)$, *hence for all* s,

$$D_{\mathrm{KL}}\big(\hat{\pi}^{\,i}(\cdot|s)\|\hat{\pi}^{\,i-1}(\cdot|s)\big) = D_{\mathrm{KL}}\big(\pi_{\mathrm{tar}}^{\sigma(i)}(\cdot|s)\|\pi_{\mathrm{cur}}^{\sigma(i)}(\cdot|s)\big) \le \delta_i.$$

Theorem 4.4 (Joint-stage monotonic improvement). *After a stage with radii* $\{\delta_i\}_{i=1}^{n}$,

$$J(\bar{\pi}) - J(\pi_{\mathrm{cur}}) \;\ge\; \sum_i L_i^{SEQ} - \frac{2\gamma}{(1-\gamma)^2}\sum_i A_{\max}^{(i)}\sqrt{\tfrac{1}{2}\,\delta_i} - \frac{1}{1-\gamma}\sum_i \zeta_i.$$

In the uniform case $\delta_i \equiv \delta$, *the penalty scales as* $O(n\sqrt{\delta})$, *where* $A_{\max}^{(i)} \triangleq \sup_{s,a}\big|A^{\hat{\pi}^{\,i-1}}(s,a)\big|$.

Proof sketch. Telescoping gives $J(\bar{\pi}) - J(\pi_{\mathrm{cur}}) = \sum_{i=1}^{n}[J(\hat{\pi}^{\,i}) - J(\hat{\pi}^{\,i-1})]$. Apply Theorem 4.2 to each term. KL penalties add directly because each step constrains only one agent (Lemma 4.3); there are no cross-terms. Full details are in the Appendix (Joint-stage). □

Structural properties. Sequence-agnosticism: Theorem 4.4 holds for any update order σ, including data-dependent choices; the numeric value of the lower bound can still depend on the realized order. Plug-and-play invariance: Replacing agents with stronger models preserves guarantees if they optimize the same surrogate under the same constraints; this is realized via a Stage-0 alignment that starts within the trust region (Appendix Stage-0). Certificate tightening: Upgrades either increase $\sup L_i^{\mathrm{SEQ}}$ at fixed δ_i or achieve the same surrogate with smaller radii $\delta_i' < \delta_i$, reducing $\sum_i \sqrt{\delta_i}$. A high-probability relaxation can replace δ_i with $\delta_i + \epsilon(N_i,\eta)$, where $\epsilon(N_i,\eta) = \sqrt{\log(2/\eta)/(2N_i)}$ (DKW; see Appendix Stage-0).

4.4 Information-Theoretic Envelopes

We now establish fundamental limits on achievable improvements under KL constraints and finite sampling budgets. Let N_i denote the number of on-policy episodes at step i under $d^{\hat{\pi}^{\,i-1}}$.

Information-geometric preliminaries. Assume L_i^{SEQ} is twice differentiable wrt the parameters θ_i of agent $\sigma(i)$ near $\pi_{\mathrm{cur}}^{\sigma(i)}$. Let $g_i = \nabla_{\theta_i} L_i^{\mathrm{SEQ}}$ and

$$F_i = \mathbb{E}_{s\sim d^{\hat{\pi}^{\,i-1}}}\Big[\mathbb{E}_{a\sim\pi_{\theta_i}(\cdot|s)}\big[\nabla_{\theta_i}\log\pi_{\theta_i}(a|s)\nabla_{\theta_i}\log\pi_{\theta_i}(a|s)^{\top}\big]\Big].$$

Local smoothness: $L_i^{\mathrm{SEQ}}(\theta_i+\Delta) \ge L_i^{\mathrm{SEQ}}(\theta_i) + g_i^{\top}\Delta - \frac{L_i^{\mathrm{loc}}}{2}\|\Delta\|^2$. Fisher–KL bridge: $\mathbb{E}_s[D_{\mathrm{KL}}(\pi_{\theta_i+\Delta}\|\pi_{\theta_i})] \approx \frac{1}{2}\Delta^{\top}F_i\Delta$ for small $\|\Delta\|$.

Theorem 4.5 (Oracle single-step upper bound). *Under* $D_{\mathrm{KL}}^{\max}(\hat{\pi}^{\,i}\|\hat{\pi}^{\,i-1}) \le \delta_i$ *and* $|A| \le A_{\max}$,

$$J(\hat{\pi}^{\,i}) - J(\hat{\pi}^{\,i-1}) \;\le\; \frac{A_{\max}}{1-\gamma}\sqrt{2\,\delta_i}.$$

Theorem 4.6 (Finite-budget single-step envelope). *With* N_i *on-policy episodes at step* i, *with probability at least* $1-\delta$,

$$J(\hat{\pi}^{\,i}) - J(\hat{\pi}^{\,i-1}) \;\le\; \frac{A_{\max}}{1-\gamma}\sqrt{2\,\delta_i} \;+\; \frac{A_{\max}}{1-\gamma}\sqrt{\frac{\log(2/\delta)}{2N_i}}.$$

Theorem 4.7 (Budget-aware stage lower bound). *Under the local smoothness and Fisher–KL bridge assumptions, with* $F_i^{\mathrm{reg}} = F_i + \epsilon I$ *and noting that* $\mathbb{E}_s D_{\mathrm{KL}} \le D_{\mathrm{KL}}^{\max}$ *(so any effective radius* $\bar{\delta}_i \le \delta_i$ *may be used), with probability at least* $1-\delta_{\mathrm{conf}}$,

$$J(\bar{\pi}) - J(\pi_{\mathrm{cur}}) \;\ge\; \sum_{i=1}^{n}\big(\kappa_i^{\mathrm{reg}}\sqrt{\bar{\delta}_i} - a_i^{\mathrm{reg}}\bar{\delta}_i\big) - \frac{2\gamma}{(1-\gamma)^2}A_{\max}\sum_{i=1}^{n}\sqrt{\tfrac{1}{2}\,\delta_i} - \frac{1}{1-\gamma}\sum_{i=1}^{n}\zeta_i \;-\; \sum_{i=1}^{n}\frac{A_{\max}}{1-\gamma}\sqrt{\frac{\log(2n/\delta_{\mathrm{conf}})}{2N_i}},$$

where $\kappa_i^{\mathrm{reg}} = \sqrt{2\,g_i^{\top}(F_i^{\mathrm{reg}})^{-1}g_i}$ *and* $a_i^{\mathrm{reg}} = L_i^{\mathrm{loc}}/\lambda_{\min}(F_i^{\mathrm{reg}})$.

4.5 Convergence and Finite-Sample Analysis

Stage objective and block updates. Let $\theta = (\theta_1,\dots,\theta_n)$ parameterize the team policy $\pi_\theta = \prod_{j=1}^{n}\pi_{\theta_j}^{(j)}$. Given the sequence-aware surrogates $\{L_i^{\mathrm{SEQ}}\}_{i=1}^{n}$ under the intermediate occupancies $\{d^{\hat{\pi}^{\,i-1}}\}$, define

$$G(\theta) \;\triangleq\; \sum_{i=1}^{n} L_i^{\mathrm{SEQ}}(\theta) \;=\; \sum_{j=1}^{n} G_j(\theta_j), \qquad G_j(\theta_j) := L_{\sigma^{-1}(j)}^{\mathrm{SEQ}}(\theta_j).$$

By construction, each L_i^{SEQ} depends only on block $\theta_{\sigma(i)}$ under fixed $d^{\hat{\pi}^{\,i-1}}$, hence mixed second derivatives vanish.

At the *start of the stage*, fix per-agent trust regions.

$$\Theta_j := \Big\{\theta_j : D_{\mathrm{KL}}\Big(\pi_{\theta_j}^{(j)}(\cdot\mid s)\,\Big\|\,\pi_{\mathrm{cur}}^{(j)}(\cdot\mid s)\Big) \le \delta_j(s)\ \forall s\Big\}, \qquad \Theta := \prod_{j=1}^{n}\Theta_j,$$

which remain *fixed within the stage.* Perform one sweep of *sequential* block updates following σ:

$$\theta_j^{\,i} \;=\; \mathrm{Proj}_{\Theta_j}\Big(\theta_j^{\,i-1} + \eta_j\,\nabla_{\theta_j}G(\theta^{\,i-1})\Big), \qquad \theta_{-j}^{\,i} = \theta_{-j}^{\,i-1}, \quad j = \sigma(i),$$

And define the block projected-gradient mapping. $g_{\eta_j}^{(j)}(\theta^{\,i-1}) := \frac{1}{\eta_j}\big(\theta_j^{\,i} - \theta_j^{\,i-1}\big)$.

Theorem 4.8 (Sequential block-coordinate gradient descent (BCGD) on a fixed trust region: monotonicity and rate). *Assume* $|\widehat{A}_{\mathrm{ON}}^{\,i-1}| \le A_{\max}$ *and bounded log-policy derivatives* $\|\nabla_\theta\log\pi_\theta(\cdot\mid s)\| \le B_1$, $\|\nabla_\theta^2\log\pi_\theta(\cdot\mid s)\|_{\mathrm{op}} \le B_2$ *on* Θ. *Then each* G_j *is* L_{blk}*-smooth with* $L_{\mathrm{blk}} = \frac{A_{\max}}{1-\gamma}(B_2 + B_1^2)$. *If* $\eta_j \le 1/L_{\mathrm{blk}}$, *every block step is ascent and*

$$G(\theta^{\,i}) - G(\theta^{\,i-1}) \;\ge\; \Big(\eta_j - \tfrac{L_{\mathrm{blk}}\eta_j^2}{2}\Big)\big\|g_{\eta_j}^{(j)}(\theta^{\,i-1})\big\|^2 \;\ge\; \tfrac{\eta_j}{2}\big\|g_{\eta_j}^{(j)}(\theta^{\,i-1})\big\|^2.$$

After K *block updates (e.g., one sweep has* $K = n$),

$$\frac{1}{K}\sum_{t=1}^{K}\big\|g_{\eta_{j_t}}^{(j_t)}(\theta^{\,t-1})\big\|^2 \;\le\; \frac{2\,(G^\star - G(\theta^{\,0}))}{(\min_t\eta_{j_t})\,K}, \qquad G^\star = \sup_{\theta\in\Theta}G(\theta).$$

Proof sketch. *Block smoothness from bounded Hessians, one-step descent lemma on the active block, and Euclidean projection optimality; telescope across blocks. Full proof: Appendix (PGD).*

Theorem 4.9 (Finite-sample concentration for L_i^{SEQ}). *Let $\widehat{L}_i^{SEQ}$ be the empirical surrogate from N_i on-policy episodes at step i and set $B := A_{\max}/(1-\gamma)$. For any $\delta \in (0,1)$, with i.i.d. episodes,*

$$\left|\widehat{L}_i^{SEQ} - L_i^{SEQ}\right| \le B\sqrt{\frac{\log(2/\delta)}{2N_i}} \quad \text{with probability at least } 1-\delta.$$

Under β-mixing episodes with $\sum_{t\ge1}\beta(t) < \infty$, the same bound holds with N_i replaced by $N_{i,\text{eff}} = N_i/(1+2\sum_{t\ge1}\beta(t))$. Variance-aware and robust MoM variants are given in the Appendix: Finite-sample.

5 ALGORITHM

We instantiate our framework with SAT, which updates agents sequentially using sequence-level optimization. Unlike token-level baselines (e.g., PPO), SAT uses a group-relative, sequence-level objective that evaluates complete trajectories and normalizes advantages within prompt-level groups such as GRPO and DAPO [21, 36]. This is consistent with the masked-activation product-policy view (Sec.3): only active heads contribute to joint actions and divergences at each state, and standard factorization is recovered in the fully simultaneous case. Each part below reflects an assumption or value in Sec . 4: the sequence-level surrogate targets L_i^{SEQ}, clipped advantages ensure boundary conditions for $A_{\max}$, and the trust-region controller enforces per-agent per-state $D_{\text{KL}}^{\max}$.

Sequence-level optimization. Token-level RLHF methods compute advantages per action and update policies using per-step likelihood ratios. In multi-agent settings with sequential updates, such token-wise advantages may induce off-policy errors as agents update one by one; credit assignment across agents is also ambiguous. SAT operates at the sequence level, evaluating complete trajectories and normalizing advantages within groups of completions for the same prompt. This matches our theory, where the surrogate L_i^{SEQ} evaluates full episodes under the intermediate policy $\hat{\pi}^{\,i-1}$, reducing distribution mismatch and fitting stage-wise bounds in Sec . 4.

Sequential updates with intermediate policies. At each stage, agents are updated in order $\sigma(1), \dots, \sigma(n)$. When updating agent $\sigma(i)$, we form the intermediate policy

$$\hat{\pi}^{\,i-1} = \left(\pi_{\text{tar}}^{\sigma(1)}, \dots, \pi_{\text{tar}}^{\sigma(i-1)}, \pi_{\text{cur}}^{\sigma(i)}, \dots, \pi_{\text{cur}}^{\sigma(n)}\right).$$

The sequence-aware advantage estimator conditions on $\hat{\pi}^{\,i-1}$. At the beginning of each stage, we collect fresh on-policy rollouts under $\hat{\pi}^{\,0} = \pi_{\text{cur}}$, and within the stage, we reuse these trajectories by recomputing per-timestep importance ratios

$$\rho_t = \frac{\hat{\pi}^{\,i-1}(a_t \mid s_t)}{\pi_{\text{cur}}(a_t \mid s_t)} \quad \text{with truncated weights} \quad c_t = \min\{1, \rho_t\}$$

in the multi-step estimator; advantages use GAE with $\lambda = 0.95$. This realizes the "sequence-aware" evaluation required by Sec . 4, and the truncation-induced bias is explicitly captured by the estimator term ζ_i that appears in the improvement bounds.

Algorithm 1 Sequential Agent Tuning (SAT)

Input: Team $\{\pi^{(j)}\}_{j=1}^{n}$, prompts $\mathcal{D}$, KL radii $\{\delta_j\}$, group size G_{grp}, clip ε, penalty β
Initialize: $\pi_{\text{cur}} \leftarrow (\pi^{(1)}, \dots, \pi^{(n)})$
for stage $k = 1, 2, \dots$ **do**
  $\mathcal{B} \leftarrow \text{ROLLOUT}(\pi_{\text{cur}}, \mathcal{D})$; $\sigma \leftarrow \text{ORDERAGENTS}(\mathcal{B})$ ▷ On-policy data; update order
  for $i = 1$ to n **do**
    $\hat{\pi}^{\,i-1} \leftarrow (\pi_{\text{tar}}^{\sigma(1)}, \dots, \pi_{\text{tar}}^{\sigma(i-1)}, \pi_{\text{cur}}^{\sigma(i)}, \dots, \pi_{\text{cur}}^{\sigma(n)})$
    $\{\tilde{A}_g\} \leftarrow \text{SEQADVANTAGES}(\mathcal{B}, \hat{\pi}^{\,i-1}, G_{\text{grp}})$ ▷ GAE + group-norm + clip to $A_{\max}$
    $\mathcal{L}_i \leftarrow \mathbb{E}[\min\{r_i\tilde{A}_g, \exp(\text{clip}(u_i, \log(1 \pm \varepsilon)))\tilde{A}_g\}] - \beta\mathbb{E}_s[D_{\text{KL}}(\pi^{\sigma(i)} \| \pi_{\text{cur}}^{\sigma(i)})]$
    $\pi_{\text{tar}}^{\sigma(i)} \leftarrow \text{OPTIMIZE}(\mathcal{L}_i)$ ▷ Trust-region update
    if $\text{Quantile}_{1-\alpha}[D_{\text{KL}}(\pi_{\text{tar}}^{\sigma(i)} \| \pi_{\text{cur}}^{\sigma(i)})] > \delta_i$ **then backtrack** and increase β ▷ Enforce $D_{\text{KL}}^{\max} \le \delta_i$
    end if
    $\pi_{\text{cur}}^{\sigma(i)} \leftarrow \pi_{\text{tar}}^{\sigma(i)}$
  end for
  $\pi_{\text{cur}} \leftarrow (\pi_{\text{cur}}^{(1)}, \dots, \pi_{\text{cur}}^{(n)})$ ▷ Complete stage: $\bar{\pi}$ in Thm. 4.4
end for
return π_{cur}

Group-relative normalization. For each prompt, we sample G_{grp} trajectories under $\hat{\pi}^{\,i-1}$ and compute the group-normalized advantage $\tilde{A}_g$ for $g \in \{1, \dots, G_{\text{grp}}\}$:

$$\tilde{A}_g = \frac{\widehat{A}_g^{i-1} - \mu}{\sigma + \epsilon}, \quad \mu = \frac{1}{G_{\text{grp}}}\sum_{j=1}^{G_{\text{grp}}}\widehat{A}_j^{i-1}, \quad \sigma^2 = \frac{1}{G_{\text{grp}}}\sum_{j=1}^{G_{\text{grp}}}(\widehat{A}_j^{i-1} - \mu)^2.$$

Here, $\widehat{A}_g^{i-1}$ is the aggregated per-timestep advantage. We use $G_{\text{grp}} \in \{4, 8\}$ to balance variance and cost. Finally, we apply symmetric clipping $|\tilde{A}_g| \le A_{\text{clip}}$ (setting $A_{\max} := A_{\text{clip}}$) to satisfy the bounded-advantage assumption in Sec. 4.

Trust-region update for agent $\sigma(i)$. Let $\mathcal{T}_i(\tau)$ denote the timesteps where agent $\sigma(i)$ acts along trajectory τ.

$$u_i(\tau) = \sum_{t \in \mathcal{T}_i(\tau)} \Big(\log \pi^{\sigma(i)}(a_{i,t} \mid s_t) - \log \pi_{\text{cur}}^{\sigma(i)}(a_{i,t} \mid s_t)\Big),$$
$$r_i(\tau) = \exp\big(u_i(\tau)\big).$$

We optimize a clipped sequence-level objective with a per-agent KL penalty and a high-quantile monitor for $D_{\text{KL}}^{\max}$ constraint:

$$\mathcal{L}_i = \mathbb{E}\Big[\min\big\{r_i(\tau)\,\tilde{A}_g,\ \exp\big(\text{clip}(u_i(\tau), \log(1-\varepsilon), \log(1+\varepsilon))\big)\,\tilde{A}_g\big\}\Big] - \beta \cdot \mathbb{E}_s\Big[D_{\text{KL}}\big(\pi^{\sigma(i)}(\cdot \mid s) \,\|\, \pi_{\text{cur}}^{\sigma(i)}(\cdot \mid s)\big)\Big],$$

with $\varepsilon = 0.2$ by default. The penalty coefficient β is adapted online, and updates are backtracked whenever the empirical $(1-\alpha)$-quantile of per-state KL exceeds the target radius δ_i. This enforces the per-agent trust region in the sense required by Sec. 4: the quantile controller yields a high-probability relaxation of $D_{\text{KL}}^{\max}$ that maps the theoretical radius to an effective $\delta_i + \epsilon(N, \eta)$, closing the gap between implementation and the bounds in Theorems 4.2–4.4.

Plug-and-play agent upgrades. When replacing agent j with a stronger pretrained model $\pi_{\text{pre}}^{(j)}$, we perform a Stage-0 alignment step that projects $\pi_{\text{pre}}^{(j)}$ onto the trust region around the $\pi_{\text{cur}}^{(j)}$:

$$\pi_{\text{new}}^{(j)} = \arg\min_{\pi} \mathbb{E}_{s\sim d^{\pi_{\text{cur}}}}\Big[D_{\text{KL}}\big(\pi(\cdot \mid s) \,\|\, \pi_{\text{pre}}^{(j)}(\cdot \mid s)\big)\Big]$$
$$\text{subject to} \quad D_{\text{KL}}\big(\pi(\cdot \mid s) \,\|\, \pi_{\text{cur}}^{(j)}(\cdot \mid s)\big) \le \delta_0(s) \;\; \forall s.$$

The Lagrangian yields a closed-form geometric mixture for state s:

$$\pi_{\text{new}}^{(j)}(a \mid s) = \frac{\pi_{\text{pre}}^{(j)}(a \mid s)^{1/(1+\lambda(s))}\,\big(\pi_{\text{cur}}^{(j)}(a \mid s)\big)^{\lambda(s)/(1+\lambda(s))}}{\sum_{a'} \pi_{\text{pre}}^{(j)}(a' \mid s)^{1/(1+\lambda(s))}\,\big(\pi_{\text{cur}}^{(j)}(a' \mid s)\big)^{\lambda(s)/(1+\lambda(s))}},$$

where the state-dependent Lagrange multiplier $\lambda(s) \ge 0$ is chosen by binary search to satisfy the per-state KL constraint with equality when the pretrained policy would otherwise violate it. This initialization ensures $\pi_{\text{new}}^{(j)}$ starts within the trust region, so Theorems 4.2–4.4 remain valid. Moreover, if $\pi_{\text{pre}}^{(j)}$ is superior, the stage surrogate L_i^{SEQ} improves at fixed δ_i, or the same surrogate value can be achieved with tighter radii, reducing the occupancy-shift.

6 EXPERIMENTS

We evaluate SAT across three domains: mathematical reasoning, active reasoning, and planning. Our experiments address three questions in a single unified protocol: whether SAT-trained small-model teams can match or exceed much larger monolithic models, whether empirical improvements align with the theory in Sec.4, and how plug-and-play upgrades affect performance under fixed trust-region budgets. Unless explicitly noted, we follow the masked-activation product-policy view for action composition.

6.1 Experimental Setup

6.1.1 Models and Datasets. We evaluate SAT by finetuning three small LLMs with parameters ranging from 1.5B to 8B: including LLaMA-3.3 (3B-instruct) [3], Qwen2.5 (1.5B-instruct, 3B-instruct, 7B-instruct) [17], and Qwen3 (4B-instruct, 8B-instruct) [32]. To demonstrate that SAT-trained small models can achieve competitive performance, we compare them against larger baseline models ranging from 30B to 70B parameters, such as LLaMA-3.3 (70B-instruct), Qwen2.5 (32B-instruct, 72B-instruct), and Qwen3 (30B-A3B-Instruct-2507, 32B), as well as publicly available thinking/non-thinking systems where applicable. For mathematical reasoning, we use AIME 2024/2025[2], ZebraLogic [10], and MATH-500 [6]. For active reasoning, we utilize ARBench [38] to evaluate the capacity of LLMs to formulate suitable questions to acquire additional information. For planning, we use AutoLogi[39]and PlanBench [24], including BlocksWorld and logistics domains. If the exact released parameter counts of some checkpoints differ slightly from the labels above, this variance does not alter the evaluation protocol; the citations remain as the authoritative references.

6.1.2 Training Data. We prepare our training code with the VeRL framework [22] with temperature set to 0.8, top_p set to 1.0, and maximum output length set to 32,768 tokens for inference. Due to the high variance of the outputs from reasoning models, we report avg@K (pass@1 performance averaged over K outputs) and pass@K for each benchmark. For benchmarks with few samples (AIME24/25 and ZebraLogic), we set a larger $K = 64$. We use $K = 25$ for ARBench, $K = 8$ for planning benchmarks, and $K = 4$ for MATH-500. To ensure accurate evaluation, we adopt the verification functions from DeepScaleR for mathematics problems. However, whether all external baselines used the same verifiers is uncertain; therefore, we retain their reported numbers and citations as is. For math reasoning ability, we use datasets from DeepScaleR [13] and DAPO [36]; for active reasoning and planning, we use the training sets from ARBench and PlanBench directly.

6.2 Main Results

Table 1 presents performance comparisons across general reasoning, active reasoning, and planning. A team of three Qwen3-4B models (12B total parameters) achieves 70.3% on AIME24, 63.4% on AIME25, and 86.1% on MATH-500, while the three-agent LLaMA 3.1-8B SAT configuration (24B total) reaches 86.3% (AIME24), 76.7% (AIME25), 99.1% (MATH-500), 93.1% (ZebraLogic), and 42.7% (PlanBench-BW NL). These results indicate that SAT-trained teams of small models can be competitive with, and in some cases surpass, much larger baselines in the listed settings, while using substantially fewer parameters. The precise relative ranking varies by benchmark, and we maintain every cited baseline as reported. The empirical trends are consistent with the theoretical guidance that per-agent KL radii and sequence-aware estimation control drift, yielding near-monotonic stage-wise improvements.

6.3 Heterogeneous Models Analysis

Table 2 explores parameter-size and model-family heterogeneity under SAT without predefined roles. Performance improves from 22.5 (2×0.6B+1.7B configuration) to 52.1 (2×4B+8B configuration) as total capacity increases, with clear diminishing returns: moving from sub-2B to 4B models yields substantial improvements, while adding 8B models provides more modest gains. This pattern aligns with our theoretical analysis, which shows super-linear prefix penalties for large cumulative updates. Model selection significantly impacts team performance—the heterogeneous configuration (2×Qwen3-4B + LLaMA-8B) achieves an average score of 50.4 relative to the homogeneous 3×Qwen3-4B baseline (50.7). Adding a stronger agent (Gemma-3-12B-IT) improves performance to 53.4, although at an increased computational cost. Mixing thinking and non-thinking agents often degrades performance, consistent with the need for agents trained under compatible sequence-level surrogates.

6.4 Stage-wise Improvement Analysis

Figure 1 illustrates training dynamics and trust-region behavior across the full optimization horizon. Panels (a) and (b) show SAT training curves on AIME24 and ARBench, respectively, compared against DAPO and GRPO baselines using Qwen2.5-32B. Horizontal dashed lines denote the performance of the 3-round 3×GPT-4o-mini debate and GPT-o1-preview as reference points. On AIME24, SAT exhibits rapid early gains before stabilizing around 53% accuracy, while ARBench demonstrates more gradual but steady improvement to 41% success rate. Panel (c) provides empirical validation of our trust-region analysis during AIME24 training. The violation rate, measuring the fraction of states where the per-state KL constraint is exceeded, scales approximately as $\sqrt{\delta}$ (black dashed line), consistent with the occupancy-shift penalty in Theorem 4.4. Across

Table 1: Performance comparison across general reasoning, active reasoning, and planning. Publicly verifiable entries are kept as reported. Baseline results for Qwen3 and GPT-4 are from [32]. We mark the best result in bold and the second-best with underline.

Method	Size	General Reasoning (%)					Active Reasoning (%)			Planning (%)
		AIME24	AIME25	MATH-500	ZebraLogic	AutoLogi	DC	SP	GN	PlanBench-BW (NL)
Base Model Baselines										
Qwen3-Base	14B	31.7	23.3	90.1	33.1	79.1	37.9	32.7	36.5	29.7
Qwen3-Base	32B	31.0	20.2	88.6	29.2	78.5	41.3	35.7	39.1	34.3
Qwen2.5-Base	72B	18.9	15.0	83.6	26.6	76.7	39.5	34.1	38.1	33.7
GPT-4o-mini-2024-07-18	/	8.1	8.8	78.2	20.1	62.5	44.0	40.8	43.6	34.7
Thinking Model Baselines										
Qwen3-A3B	30B	80.4	70.9	98.0	89.5	88.1	43.1	38.4	42.3	39.1
QwQ	32B	79.5	69.5	98.0	76.8	86.3	41.9	36.1	39.5	37.9
Qwen3 (thinking)	32B	81.4	72.9	97.2	88.8	87.3	42.7	38.9	41.1	40.3
DeepSeek-R1 Distill Llama	70B	70.0	56.3	94.5	71.3	83.5	42.1	35.4	40.7	39.1
OpenAI o3-mini (medium)	/	79.6	74.8	98.0	88.9	86.3	**46.7**	**43.1**	**45.1**	41.2
Single-Agent Fine-tuning Baselines										
Qwen3-Base-GRPO	8B	39.1	27.9	90.7	35.4	80.3	40.1	35.2	38.9	33.1
Qwen3-Base-DAPO	8B	41.3	30.1	91.5	36.7	80.9	40.7	35.9	39.3	33.5
Multi-Agent with Fine-tuning Baselines. A Judge serves as a coordinator.										
Qwen3-Base Debate	3×8B	68.7	59.1	95.2	84.1	85.1	43.1	38.0	42.3	38.3
Qwen3-Base Debate + Judge	3×8B	71.1	61.7	95.9	86.3	85.9	44.3	39.4	43.5	39.1
Qwen3-Base Role-play	3×8B	69.9	60.5	95.7	85.1	85.5	43.7	38.7	43.1	38.7
SAT (Ours)										
Llama 3.1-Base SAT	3×8B	70.3	63.4	86.1	80.7	83.9	35.9	32.7	34.9	37.1
Qwen3-Base SAT	3×4B	**86.3**	**76.7**	**99.1**	**93.1**	**89.1**	45.3	41.5	43.9	**42.7**

Table 2: Heterogeneous team configurations under SAT (3 agents, no predefined roles). Datasets: AIME24, AIME25, ARBench-DC (process@25), PlanBench-BW (NL). *Params (B)* is the sum of the three agents' parameters (no weight sharing). *Heterogeneity tags*: size = different parameter scales (capacity), model = different model families / pretraining methods and corpora, thinking = different thinking styles (e.g., deliberate/CoT-distilled vs non-thinking). We mark the best column for the average result in bold and the second-best with underline.

Configuration	Params (B)	AIME24	AIME25	ARBench-DC	PlanBench-BW (NL)	Average
Parameter-Size Heterogeneity						
2×Qwen3-0.6B Base + Qwen3-1.7B Base	3.0	22.1	17.3	26.1	24.5	22.5
Qwen3-0.6B Base+ Qwen3-1.7B Base+ Qwen3-4B Base	6.3	49.7	38.5	31.5	32.9	38.2
2×Qwen3-1.7B Base+ Qwen3-4B Base	7.4	55.9	44.3	33.1	34.9	42.1
Qwen3-1.7B Base+ Qwen3-4B Base+ Qwen3-8B Base	13.7	66.9	56.3	35.5	36.7	48.9
2×Qwen3-4B Base+ Qwen3-8B Base	16.0	72.1	61.1	36.7	38.5	**52.1**
Model Heterogeneity						
2×Qwen3-4B Base+ Qwen2.5-3B Base	11.0	68.1	57.3	35.1	36.3	49.2
LLaMA 3.1-8B Base+ LLaMA 3.3-4B Base+ Qwen2.5-3B Base	15.0	64.7	54.3	34.7	35.9	47.4
2×Qwen3-4B Base+ LLaMA 3.1-8B Base	16.0	69.5	58.7	35.7	37.7	50.4
Qwen3-4B Base+ LLaMA 3.1-8B Base+ Gemma-3-12B-IT	24.0	74.3	64.1	36.9	38.3	**53.4**
Thinking Modes + Non-thinking Mix						
DeepSeek-R1-Distill-(Qwen-1.5B + Llama3.1-8B Base)↓ + Qwen3-4B (non-thinking)	13.5	58.3	47.1	33.3	35.1	43.5
2×Qwen3-0.6B (thinking) + Qwen3-4B (non-thinking)	5.2	41.7	32.5	30.1	31.9	34.1
2×Qwen3-4B (thinking) + Qwen3-1.7B (non-thinking)	9.7	62.9	52.3	34.3	36.1	46.4
2×Qwen3-4B (thinking) + Qwen3-8B (non-thinking)	16.0	64.3	53.7	34.7	36.7	**47.3**

multiple agents and training stages, violations concentrate in early steps and diminish as agents align, matching Theorem 4.9.

6.5 Plug-and-Play Upgrades

Table 3 evaluates plug-and-play capabilities under SAT without role assignments, comparing PnP upgrades against static heterogeneous training from scratch using the composite metric. Single-agent

Table 3: Plug-and-play under SAT. The baseline is Qwen3 trained by SAT. "Composite" is the mean of AIME24, AIME25, ARBench-DC under the same evaluation protocol as Table 1. We compare static heterogeneity (train from scratch) with PnP upgrades on the same baseline; KL is per-agent per-state; Viol.% is the first-stage monotonicity violation rate; Cost is relative to baseline parameters. The best performance is in bold.

Block / Strategy	Change	KL	Composite			Δ vs Baseline (AIME24/AIME25/ARBench-DC)			Viol.%	Cost
			Before	After	Gain	AIME24	AIME25	ARBench-DC		
Baseline: Qwen3 SAT (3×4B)	–	–	56.5	56.5	–	–	–	–	2.7	1×
Static hetero (train from scratch)										
Direct hetero (2×4B + LLaMA 8B)	+1× capacity	–	56.5	55.5	-1.0	-1.2	-1.0	-0.6	–	1.33×
Direct hetero (2×4B + Qwen3 8B)	+1× capacity	–	56.5	57.5	+1.0	+1.4	+1.2	+0.4	–	1.33×
Direct hetero (1×4B + 2×Qwen3 8B)	+2× capacity	–	56.5	66.5	+10.0	+11.8	+12.8	+5.6	–	1.67×
PnP hetero (plug-and-play; continue SAT)										
Replace any one with LLaMA 3.1-8B	+1× capacity	0.010	56.5	55.9	-0.6	-0.8	-0.6	-0.2	4.7	1.33×
Replace any one with Qwen2.5-3B	-1× capacity	0.010	56.5	55.5	-1.0	-1.2	-1.2	-0.6	3.9	0.92×
Replace any one with Qwen3-1.7B	-1× capacity	0.010	56.5	54.8	-1.7	-2.2	-2.2	-0.6	3.7	0.81×
Replace any one with Qwen3-0.6B	-1× capacity	0.010	56.5	51.9	-4.6	-5.8	-6.0	-2.0	3.3	0.72×
Replace any one with Gemma-3-12B-IT	+1× capacity	0.010	56.5	59.7	+3.2	+4.0	+4.8	+1.0	4.1	1.67×
Replace two with Qwen3-8B	+2× capacity	0.010	56.5	66.9	+10.4	+12.0	+13.2	+6.0	3.3	1.67×
Full PnP to Qwen3 (3×8B)	+3× capacity	0.004	56.5	**70.4**	**+13.9**	**+16.2**	**+17.6**	**+8.0**	**2.5**	2.00×

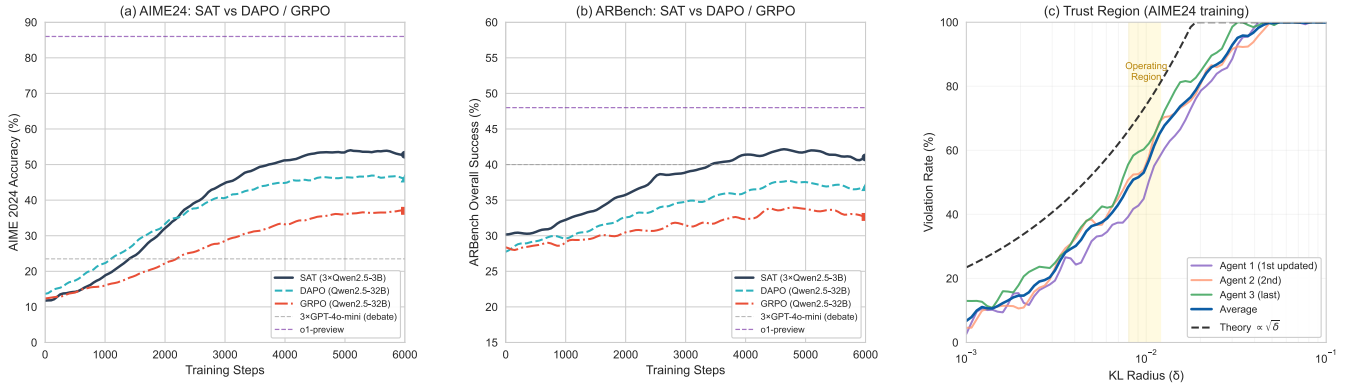


Figure 1: Stage-wise performance and trust-region validation with SAT. (a) AIME24 2024 accuracy over training steps; (b) ARBench overall success rate; (c) trust-region violation rate vs. KL radius δ on AIME24 training. Horizontal dashed lines in (a) and (b) denote 3-round 3×GPT-4o-mini debate and GPT-o1-preview baselines. SAT is compared against Qwen2.5-32B trained with DAPO and GRPO. SAT demonstrates rapid early gains on AIME24 and steady improvements on ARBench. Panel ((c) shows that violation rate scales as $\sqrt{\delta}$ (dashed theory line), with the operating region (yellow) achieving low violations while enabling effective exploration, consistent with Theorems 4.4 and 4.9.

replacement with LLaMA 3.1-8B slightly decreases the composite from 55.2 to 54.6 under $\delta = 0.010$, while replacing two agents with 8B models increases it to 65.6. Full PnP to three 8B models reaches 69.1 composite with the lowest violation rate (2.5%) at 2.00× cost. Downgrade experiments quantify capacity performance trade offs: replacing with Qwen3-0.6B reduces composite to 50.6 at 0.72× cost.

7 CONCLUSION

We have introduced Sequential Agent Tuning (SAT), a coordinator-free framework that empowers teams of small LLMs to surpass the performance of larger models. Our method integrates sequential agent updates with a sequence-aware advantage estimator and per-agent KL trust regions, ensuring stable and monotonic improvements. The central element of our theoretical contribution is the principle of "plug-and-play" invariance, which allows individual agents to be upgraded without incurring the cost of retraining the entire team. We further validated our plug-and-play theory by demonstrating that modular agent upgrades yield significant performance gains, in line with analytical expectations. These results position SAT as a practical and scalable path for deploying high-performance AI systems, especially in resource-constrained environments. We hope this work encourages a shift from simply scaling monolithic models to strategically advancing teams of smaller agents for LLM community.

ACKNOWLEDGMENTS

Y. Fan's work is not related to the position at Amazon Inc. Y. Xu's work is funded by the National Science Foundation (NSF) under grant NSF IIS1910794 and DMS-2406896;

REFERENCES

[1] Keivan Alizadeh, Iman Mirzadeh, Dmitry Belenko, S. Karen Khatamifard, Minsik Cho, Carlo C. Del Mundo, Mohammad Rastegari, and Mehrdad Farajtabar. 2024. LLM in a Flash: Efficient Large Language Model Inference with Limited Memory. *Proceedings of the 62nd Annual Meeting of the ACL* (2024). https://aclanthology.org/2024.acl-long.678.pdf

[2] Art of Problem Solving. 2024. 2024 AIME I: Problems and Solutions. https://artofproblemsolving.com/wiki/index.php/2024_AIME_I. Last accessed: 2025-09-23.

[3] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv e-prints* (2024), arXiv–2407.

[4] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. 2018. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*. PMLR, 1407–1416.

[5] Xidong Feng, Ziyu Wan, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. 2023. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179* (2023).

[6] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. In *NeurIPS 2021 Datasets and Benchmarks Track*. https://arxiv.org/abs/2103.03874

[7] Sham Kakade and John Langford. 2002. Approximately Optimal Approximate Reinforcement Learning. In *Proceedings of the 19th International Conference on Machine Learning (ICML)*. Often cited via Conservative Policy Iteration (CPI).

[8] Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*. PMLR, 19274–19286.

[9] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. CAMEL: Communicative Agents for "Mind" Exploration of Large-Scale Language Model Society. *arXiv preprint arXiv:2303.17760* (2023).

[10] Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. 2025. ZebraLogic: On the Scaling Limits of LLMs for Logical Reasoning. In *Forty-second International Conference on Machine Learning*. https://openreview.net/forum?id=sTAJ9QyA6l

[11] Siao Liu, Zhaoyu Chen, Yang Liu, Yuzheng Wang, Dingkang Yang, Zhile Zhao, Ziqing Zhou, Yi Xie, Wei Li, Wenqiang Zhang, and Zhongxue Gan. 2023. Improving Generalization in Visual Reinforcement Learning via Conflict-aware Gradient Agreement Augmentation. In *Proceedings of ICCV*. 23436–23446.

[12] Siao Liu, Yang Liu, Zhaoyu Chen, Ziqing Zhou, Zhile Zhao, Yi Xie, Wei Li, and Zhongxue Gan. 2025. Improving Robotic Grasp Detection Under Sparse Annotations Via Grasp Transformer With Pixel-Wise Contrastive Learning. *IEEE Transactions on Industrial Electronics* (2025). https://doi.org/10.1109/TIE.2025.3569940

[13] Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, et al. 2025. Deepcoder: A fully open-source 14b coder at o3-mini level. *Notion Blog* (2025).

[14] Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. 2016. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems* 29 (2016).

[15] OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774* (2023). https://arxiv.org/abs/2303.08774

[16] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon Emissions and Large Neural Network Training. *arXiv preprint arXiv:2104.10350* (2021).

[17] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 Technical Report. arXiv:2412.15115 [cs.CL] https://arxiv.org/abs/2412.15115

[18] John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. 2015. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. https://proceedings.mlr.press/v37/schulman15.html

[19] John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. 2016. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1506.02438

[20] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[21] Zhenda Shao et al. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv preprint arXiv:2402.03300* (2024). Introduces Group-Relative Policy Optimization (GRPO).

[22] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. HybridFlow: A Flexible and Efficient RLHF Framework. *arXiv preprint arXiv: 2409.19256* (2024).

[23] Noah Shinn, Federico Cassano, Ashwin Gopinath, Edward Berman, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*. https://arxiv.org/abs/2303.11366

[24] Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2022. PlanBench: An Extensible Benchmark for Evaluating Large Language Models on Planning and Reasoning about Change. *arXiv preprint arXiv:2206.10498* (2022). https://arxiv.org/abs/2206.10498 NeurIPS 2023 Poster; widely used in 2024–2025 planning evaluations.

[25] Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. 2024. Mixture-of-Agents Enhances Large Language Model Capabilities. *arXiv preprint arXiv:2406.04692* (2024). https://arxiv.org/abs/2406.04692

[26] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/2203.11171

[27] Jason Wei, Xuezhi Wang, Dale Schuurmans, et al. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv preprint arXiv:2201.11903* (2022).

[28] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. 2023. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. *arXiv preprint arXiv:2308.08155* (2023).

[29] Tengyang Xie, Bo Liu, Yangyang Xu, Mohammad Ghavamzadeh, Yinlam Chow, Daoming Lyu, and Daesub Yoon. 2018. A block coordinate ascent algorithm for mean-variance optimization. *Advances in Neural Information Processing Systems* 31 (2018).

[30] Yi Xie, Ziqing Zhou, Chun Ouyang, Siao Liu, Linqiang Hu, and Zhongxue Gan. 2025. ACORN: Acyclic Coordination with Reachability Network to Reduce Communication Redundancy in Multi-Agent Systems. In *Proceedings of AAMAS*. 2190–2198.

[31] Yi Xie, Ziqing Zhou, Chun Ouyang, Siao Liu, Linqiang Hu, and Zhongxue Gan. 2025. Heuristics-Assisted Experience Replay Strategy for Cooperative Multi-Agent Reinforcement Learning. In *Proceedings of AAMAS*. 2798–2800.

[32] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 Technical Report. arXiv:2505.09388 [cs.CL] https://arxiv.org/abs/2505.09388

[33] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *arXiv preprint arXiv:2305.10601* (2023). https://arxiv.org/abs/2305.10601

[34] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations (ICLR)*. arXiv:2210.03629 https://arxiv.org/abs/2210.03629

[35] Xie Yi, Zhanke Zhou, Chentao Cao, Qiyu Niu, Tongliang Liu, and Bo Han. 2025. From Debate to Equilibrium: Belief-Driven Multi-Agent LLM Reasoning via Bayesian Nash Equilibrium. In *Forty-second International Conference on Machine Learning*. https://openreview.net/forum?id=RQwexjUCxm

[36] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476* (2025).

[37] Shangtong Zhang, Bo Liu, and Shimon Whiteson. 2021. Mean-variance policy iteration for risk-averse reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 10905–10913.

[38] Zhanke Zhou, Xiao Feng, Zhaocheng Zhu, Jiangchao Yao, Sanmi Koyejo, and Bo Han. 2025. From Passive to Active Reasoning: Can Large Language Models Ask the Right Questions under Incomplete Information? *arXiv preprint arXiv:2506.08295* (2025).

[39] Qin Zhu, Fei Huang, Runyu Peng, Keming Lu, Bowen Yu, Qinyuan Cheng, Xipeng Qiu, Xuanjing Huang, and Junyang Lin. 2025. AutoLogi: Automated Generation of Logic Puzzles for Evaluating Reasoning Abilities of Large Language Models. arXiv:2502.16906 [cs.CL] https://arxiv.org/abs/2502.16906