# FLOWWEAVER: WEAVING REASONING PATHS WITH CROSS-TRAJECTORY FLOW FOR FINE-GRAINED CREDIT ASSIGNMENT

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

A language model may make hundreds of token-level decisions before receiving a single outcome reward, while responses sampled for the same prompt can follow different reasoning paths and reach different outcomes. Turning this cross-trajectory variation into fine-grained credit remains challenging: within-trajectory attribution can identify important tokens but evaluates each response largely in isolation, whereas cross-trajectory methods exploit outcome variation at coarser granularity or rely on shared prefixes or states. We introduce **FlowWeaver**, which weaves sampled reasoning paths into a shared attribution graph through cross-trajectory information flow for fine-grained credit assignment signals. We build on attention-based flow within each response, constructing paths from prompt to answer using attention weights and quantifying the share of graph propagation routed through each token. Using cumulative flow as a common coordinate, we connect sampled responses through aligned token positions, forming a shared distribution over graph paths from the prompt to different outcomes. We then condition propagation on each response's observed outcome, measuring how much of the group's flow toward that outcome passes through each token. We use these importance scores to shape token-level credit by reweighting GRPO advantages. RL experiments across mathematical reasoning, general reasoning, and agentic tasks demonstrate the effectiveness of FlowWeaver, highlighting outcome-conditioned flow as a fine-grained signal for allocating outcome supervision.

## 1 INTRODUCTION

A language model may make hundreds of intermediate decisions before receiving a single outcome reward. Such supervision can drive substantial gains in reasoning [29; 5], but it leaves a basic credit-assignment problem: a final reward evaluates a completed response, while policy optimization operates at the token level. Responses sampled for the same prompt can reach different outcomes, exposing variation in both computation and success. The challenge is to turn this variation into guidance about which intermediate positions should receive greater emphasis during learning.

To refine outcome supervision, token-level methods use signals such as predictive entropy, attention structure, and information flow to distinguish positions within a response [38; 16; 6]. These signals provide fine-grained attribution, but draw primarily on the computation within that response. Process reward models offer intermediate evaluations using an additional source of supervision [17; 15]. Cross-trajectory methods draw on the outcome variation among samples. Beyond the response-level comparisons in GRPO [29], tree-based methods compare continuations of shared prefixes [32], and GiGPO obtains step-level credit by grouping repeated agent states [8]. Their intermediate comparisons rely on shared structure that may be sparse across freely generated responses with different lengths, wording, and reasoning paths. A remaining challenge is to connect fine-grained attribution with the outcome variation already present in sampled trajectories.

We introduce **FlowWeaver** (Figure 1), whose central idea is to assess each token within a shared distribution over prompt-to-outcome paths in an auxiliary attribution graph. We build on information flow within individual responses [6]: attention weights define paths from the prompt to the generated answer, and the flow through a token quantifies the share of this graph propagation passing through



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

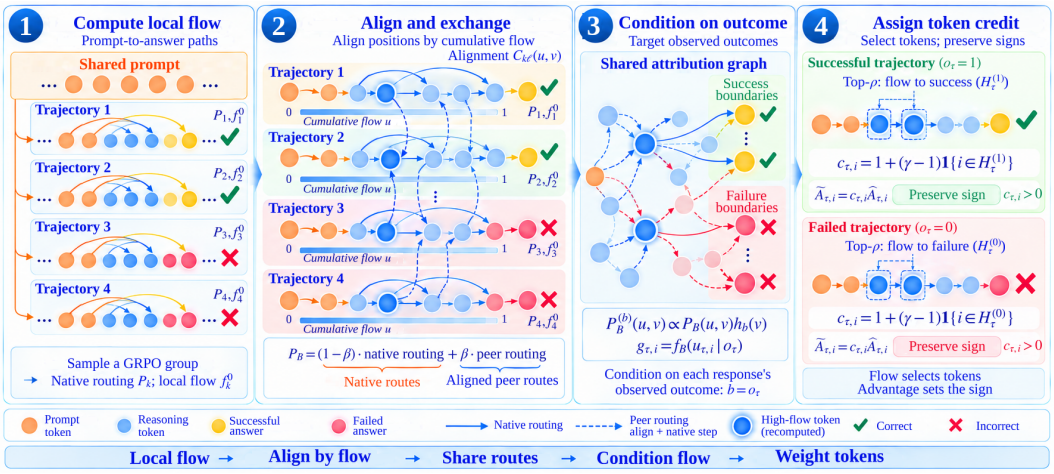


Figure 1: **Overview of FlowWeaver.** (1) Compute local prompt-to-answer flow. (2) Align tokens by cumulative flow to form a shared attribution graph. (3) Score tokens by flow to answers sharing their response's observed outcome. (4) Upweight GRPO advantages at high-flow tokens, preserving signs. Successful and failed responses supply routes; cross-trajectory arrows mark constructed links.

it. In cumulative form, this flow provides a common coordinate for aligning positions across responses. Through this alignment, we connect their onward paths, combining each response's native routes with routes drawn from its peers. Both successful and unsuccessful responses contribute to the resulting graph, giving token attribution a common reference built from the group's attention structure and observed outcomes.

This shared path distribution allows token importance to be conditioned on each response's observed outcome. Within an isolated response, the information paths all lead to the generated answer; coupling makes alternative outcomes reachable in the same attribution graph. On this graph, we measure how much flow toward answers with a target outcome passes through each token. For each response, we take its observed outcome as the target, with all sampled answers sharing that outcome serving as possible destinations. Because conditioning selects paths by where they end, a path toward success can still use routing contributed by an unsuccessful response. The resulting importance measures token participation in the group's paths toward the outcome supplying that response's supervision.

We use these scores for credit assignment in the operational sense of allocating trajectory-level supervision across tokens: the scores determine the relative weights, while GRPO advantages supply the signs. The graph scores rank participation in constructed paths and should not be read as local action advantages or signed causal reward contributions (Appendix B.6). Our evaluation follows the construction from cross-trajectory correspondence, through outcome-conditioned token ranking, to supervision allocation in RL. Cumulative-flow correspondence matches prefixes with more similar success probabilities and outcome certainty than normalized token or step progress (Table 1). Masking tokens promoted by the outcome-conditioned ranking changes correctness more often than masking equally many tokens displaced from local-flow selection (Table 2).

Using these rankings to allocate supervision, FlowWeaver improves on GRPO and trajectory-local Flow across mathematical reasoning, general reasoning, and agentic tasks on Qwen and Llama models (Section 5.2), with average agentic gains of 8.2 and 2.7 percentage points, respectively. Controlled comparisons support cumulative-flow correspondence and outcome conditioning (Appendix D.1), while mean-credit normalization retains most of the gain over GRPO in the tested setting (Appendix D.3). Peer-resampling tests assess selection stability, and five-seed evaluations confirm the RL gains (Appendices D.2 and D.5). These results support outcome-conditioned flow as an effective signal for allocating token-level credit using outcomes of sampled responses.

## 2 Related Works

**Credit Assignment in LLM Reinforcement Learning.** Credit assignment has been studied through value-based advantage estimation [27; 28; 24], critic-based reasoning optimization [46; 20], and group-relative outcome supervision [29]. Process supervision includes step verification [17; 37],

generative assessment and credit assignment [15; 41], and agent process rewards [4]. Token-level methods use entropy-based weighting and shaping [38; 36; 31], contrastive token estimates [18], and targeted exploration [34]. For cross-trajectory credit, TEMPO exploits prefix trees [32], and GiGPO forms step-level groups at repeated environment states [8]. FlowWeaver brings cross-trajectory outcome evidence into token-level credit through a shared information-flow graph.

**Tracing Computation in Language Models.** Studies of model computation examine attention-head specialization [3; 26], factual recall and context mixing [9; 23], and feature-level attribution graphs [19]. Other work investigates activation steering [33; 35], mechanisms of stepwise reasoning and planning [7; 22], and influential reasoning steps [2]. Attention-based credit [16] and trajectory-local information flow [6] translate internal structure into token weighting for RL. We extend the latter to a shared distribution of prompt-to-outcome paths across responses, with outcome conditioning providing the token attribution.

## 3 PRELIMINARIES

### 3.1 INFORMATION FLOW WITHIN A SINGLE TRAJECTORY

Information flow within a single trajectory constructs an attention-based representation of prompt-to-answer paths for attribution [6]. Using the attention of a completed response, the computation first works backward from the answer to weight answer-reaching routes, then propagates mass forward from the prompt to score the tokens on those routes. These scores provide a structural signal for allocating token-level credit.

For a prompt $q$ and sampled response $y = (x_1, \ldots, x_T)$, let $\tau = (q, y)$ denote the resulting trajectory. The construction represents the token sequence as a time-ordered directed acyclic graph (DAG), with token occurrences as nodes and edges $u \to v$ from earlier to later positions. Nonnegative edge weights $W(u, v)$ are derived from attention aggregated over selected layers and heads. These raw weights summarize attention connectivity without isolating answer-reaching paths or conserving outgoing mass.

An absorbing boundary $z$ marks the answer region, where propagation terminates. Starting from this boundary and proceeding in reverse token order, a backward potential $b$ accumulates the total weight of paths from each node to the answer:

$$b(z) = 1, \qquad b(u) = \sum_{v:u\to v} W(u, v)b(v). \tag{1}$$

For each node with $b(u) > 0$, normalizing its outgoing answer-reaching path weights gives the routing kernel

$$P(u, v) = \frac{W(u, v)b(v)}{b(u)}, \qquad \sum_{v} P(u, v) = 1. \tag{2}$$

The product $W(u, v)b(v)$ combines the direct connection from $u$ to $v$ with the weight of downstream paths from $v$ to the answer. Normalization turns these products into fractions: $P(u, v)$ specifies how much of the mass at $u$ is sent to $v$. Each retained nonterminal node redistributes all its mass, assigning none to continuations that cannot reach the answer.

A unit of source mass is then distributed over the prompt tokens according to $\sigma$ (uniform in this work), with zero source mass elsewhere. Propagating it forward through $P$ induces the reference flow $f^0$:

$$f^0(v) = \sigma(v) + \sum_{u:u\to v} f^0(u)P(u, v). \tag{3}$$

In token order, each node adds up the mass arriving from its predecessors and passes it onward according to $P$. Thus, $f^0(v)$ is the prompt-originating mass that traverses $v$ on its way to the answer in this attribution graph. High-flow positions carry more of this mass and can be selected for greater emphasis in policy updates; the outcome reward supplies the supervision. This local flow measures participation in paths to the generated answer, whether correct or incorrect, using the attention structure of that response.

Applied independently to responses sampled for the same prompt, the construction yields a native routing kernel $P_k$ and reference flow $f_k^0$ for each trajectory. These play distinct roles in Section 4.1:

the kernels specify the routing available for exchange, while the reference flows are used to establish correspondence across responses. They provide the structural basis on which the group's alternative computations and observed outcomes can refine attribution.

# 4 METHODOLOGY

Building on trajectory-local flow, we construct outcome-conditioned token importance on a shared graph (Section 4.1), examine how the correspondence and rankings relate to outcome behavior (Section 4.2), and use the rankings to reweight GRPO updates (Section 4.3).

## 4.1 OUTCOME-CONDITIONED INFORMATION FLOW ACROSS TRAJECTORIES

Outcome supervision assigns a learning signal to each response, while the other responses sampled for the same problem expose alternative computations and outcomes. Our goal is to use this group context to reassess which tokens should receive emphasis. We define token importance by outcome-conditioned flow on a shared graph: coupling supplies routes to the group's answer boundaries, and conditioning resolves token participation in those routes by their terminal outcome.

Let $\mathcal{B} = \{\tau_k\}_{k=1}^{B}$ be a GRPO group sampled for prompt $q$. Each $\tau_k$ has a native kernel $P_k$ and reference flow $f_k^0$ from Section 3.1, an answer boundary $z_k$, and a binary outcome $o_k \in \{0, 1\}$.

**Constructing the shared path distribution.** Responses to the same problem may follow different reasoning paths and have unequal lengths. We use cumulative reference flow as a shared coordinate for exchange across these paths, without requiring semantic equivalence between matched positions. Each response's token flows are normalized and accumulated in token order, assigning each token a span of this normalized mass. Normalized overlaps between these spans define correspondence weights $C_{k\ell}(u, v)$ between positions in different trajectories. This preserves order while allowing one position to match several peer positions; the precise construction is in Appendix B.1.

At a response position $u$ in trajectory $k$, a fraction $1 - \beta$ follows native routing, while $\beta$ follows routing borrowed from corresponding positions in uniformly selected peers. Writing $\mathcal{R}_\ell$ for the response tokens of peer $\ell$, the coupled kernel for $B > 1$ is

$$P_{\mathcal{B}}(u, \cdot) = (1 - \beta)P_k(u, \cdot) + \frac{\beta}{B - 1} \sum_{\ell \neq k} \sum_{v \in \mathcal{R}_\ell} C_{k\ell}(u, v) P_\ell(v, \cdot). \tag{4}$$

The peer term selects an aligned token $v$ as a routing donor and moves directly to a successor under its native row $P_\ell(v, \cdot)$. Token flow counts visits to states of this composed kernel; selecting $v$ as a donor does not itself add a visit to $v$. Both successful and unsuccessful trajectories contribute routing to the shared attribution graph, allowing token scores to draw on structure beyond their own response. The graph is constructed from the fixed sampled responses. The composed routing conserves mass and remains acyclic; proofs and boundary conventions are in Appendix B.2.

**Conditioning the shared paths on an outcome.** The shared graph can lead to answers with different outcomes. To align token importance with the outcome that supplies a response's supervision, we extend answer targeting from a single answer boundary to all boundaries sharing a target outcome $b \in \{0, 1\}$. Let $h_b(u)$ be the probability that a path starting at $u$ reaches such a boundary under the coupled kernel $P_{\mathcal{B}}$. It is computed backward through the shared graph, with boundary values indicating whether each answer has outcome $b$ (Appendix B.3). The conditioned routing is

$$P_{\mathcal{B}}^{(b)}(u, v) = P_{\mathcal{B}}(u, v) \frac{h_b(v)}{h_b(u)}, \qquad h_b(u) > 0. \tag{5}$$

The factor $h_b(v)$ weights each possible next step by its downstream probability of reaching the target outcome, and division by $h_b(u)$ preserves a unit outgoing distribution. Conditioning therefore selects paths by where they end. For example, a path targeting success can still borrow routing from an unsuccessful response before reaching a successful answer boundary. Both outcome classes remain sources of computational structure, while the target outcome stays fixed along the path.

Table 1: Mean prefix gaps. Normalized Token/Step progress vs cumulative Flow.

**Prefix matching**

| Rule | $\lvert p - p' \rvert$ (pp) ↓ | $\lvert c - c' \rvert$ (pp) ↓ |
|---|---|---|
| Random | 48.8 | 38.9 |
| Token | 42.4 | 33.5 |
| Step | 45.7 | 36.1 |
| **Flow** | **33.6** | **16.6** |

Table 2: Token masking: rate % (Δ pp vs shared unmasked resampling, Ref.). Local/Ours: local/observed-outcome flow; flips count both directions. In (b), entering/leaving the top-20% set each cover 7.23% of tokens overall; random masks match counts per response.

**(a) Mask 20%**

| Selection | Answer change | Correctness flip |
|---|---|---|
| Random | 31.2 (+10.4) | 14.0 (+3.2) |
| Local top | 44.1 (+23.3) | 23.9 (+13.1) |
| Ours bottom | 26.4 (+5.6) | 11.2 (+0.4) |
| **Ours top** | **65.7** (+44.9) | **47.4** (+36.6) |

**(b) Set change**

| Selection | Answer change | Correctness flip |
|---|---|---|
| Ref. | 20.8 (—) | 10.8 (—) |
| Random | 30.0 (+9.2) | 11.5 (+0.7) |
| Leaving | 33.4 (+12.6) | 12.9 (+2.1) |
| **Entering** | **47.4** (+26.6) | **24.1** (+13.3) |

**Observed-outcome flow provides a common token-importance signal.** With the conditioned routes in place, forward propagation proceeds as in Section 3.1. We split a common prompt source equally across trajectories and keep it fixed across outcome targets, so the target changes routing rather than the amount injected into each response. Let $f_{\mathcal{B}}(u \mid b)$ denote the accumulated mass visiting token $u$ under routing conditioned on outcome $b$. The exact recurrence, source reachability requirements, and computation are given in Appendix B.4. For a token $u_{\tau,i}$ in trajectory $\tau$, its importance is read from the flow toward that trajectory's observed outcome:

$$g_{\tau,i} = f_{\mathcal{B}}(u_{\tau,i} \mid o_\tau). \tag{6}$$

This gives successful and unsuccessful responses the same form of importance signal, assessed using the group's alternative routes to their respective outcomes. Section 4.2 assesses the revised ranking through token-masking interventions. Section 4.3 then uses these importance scores to reweight the GRPO policy surrogate, with the trajectory advantage supplying the supervision sign.

### 4.2 Flow Correspondence and Rankings Relate to Outcome Behavior

We examine the outcome behavior of prefixes and tokens connected by the shared attribution in Section 4.1. Prefix continuations assess how closely the correspondence matches success probabilities and outcome certainty; token-masking interventions assess the outcome sensitivity of positions emphasized by the resulting ranking. Both analyses use mixed-outcome groups of completed responses sampled from a frozen policy, with fresh continuations reserved for evaluation (Appendix C.3).

**Flow correspondence links prefixes with more similar success probabilities.** The correspondence in Section 4.1 follows naturally from coupling the ordered reference-flow distributions: matching their cumulative mass preserves token order and the normalized flow marginals (Appendix B.1). To examine its empirical meaning, we estimate each prefix's continuation success probability $p$ and outcome certainty $c = \max(p, 1-p)$. For matched prefixes, $|p - p'|$ directly compares success probabilities, while $|c - c'|$ compares certainty regardless of which outcome is favored. Cumulative flow reduces the mean success-probability gap from 42.4 to 33.6 percentage points, a 20.8% reduction relative to normalized token progress, the strongest alternative (Table 1); the mean certainty gap falls by 50.4%. Flow therefore matches prefixes with more similar continuation success probabilities as well as outcome certainty.

**Tokens emphasized by shared flow are more outcome-sensitive under masking.** To assess the resulting token selection at matched masking budgets, we block forward information from selected positions before a shared breakpoint and sample new continuations (Table 2). Masking the highest observed-outcome-flow tokens changes answers and correctness more often than local-flow or random selection, while masking low-flow tokens leaves correctness flips close to unmasked resampling. We further compare tokens newly entering the top-flow set with those displaced from the local-flow selection, excluding positions both methods already identify. At equal mask counts, blocking entering positions yields a correctness-flip rate 11.2 percentage points higher than blocking leaving positions. This comparison links promotion in the flow ranking to greater outcome sensitivity under masking, motivating the use of these rankings to allocate token-level supervision.

### 4.3 Shaping Token Credit with Outcome-Conditioned Importance

We use outcome-conditioned flow to allocate trajectory-level supervision across tokens. Building on information-flow-based token weighting [6], the ranking determines where supervision is em-

phasized, and the trajectory advantage supplies its sign. Credit assignment here uses participation in the shared graph's outcome-directed paths to shape the token-level policy update.

**Flow rankings determine where supervision is amplified.** We assign a fixed amplification to a selected fraction of each response. This uses the flow's relative ordering without requiring its raw magnitude to be calibrated as an update weight. For a trajectory $\tau$ with $T_\tau$ response tokens, rank positions by $g_{\tau,i}$ from Eq. (6). Let $\mathcal{H}_\tau$ contain the top-$\rho$ fraction, with $|\mathcal{H}_\tau| = \lceil \rho T_\tau \rceil$ and $0 < \rho \leq 1$; ties are resolved by token order. Thus, the amplified set is selected using success-conditioned flow for successful responses and failure-conditioned flow for unsuccessful responses. The weight is

$$c_{\tau,i} = 1 + (\gamma - 1)\mathbf{1}\{i \in \mathcal{H}_\tau\}, \qquad \gamma > 1. \tag{7}$$

Selected positions receive weight $\gamma$, and all other positions retain weight one. The fraction $\rho$ controls how broadly extra emphasis is distributed, while $\gamma$ controls its strength. With $T_\tau$, $\rho$, and $\gamma$ fixed, the total weight is $T_\tau + (\gamma - 1)\lceil \rho T_\tau \rceil$, independent of the ranking. Cross-trajectory information therefore changes which positions receive the additional emphasis under the same weighting budget.

**Positive weights preserve advantage signs.** We combine structural importance with outcome supervision through positive credit weights. Let $\widehat{A}_{\tau,i}$ be the original advantage computed from the full training reward by the underlying GRPO algorithm [29]. We use the resulting credit-weighted advantages in the GRPO policy surrogate:

$$\widetilde{A}_{\tau,i} = c_{\tau,i}\widehat{A}_{\tau,i} \tag{8}$$

Credits are computed from the rollout policy and sampled group, then held fixed with stop-gradient throughout policy optimization on that rollout batch. We apply the weighted advantage in the clipped policy term, retaining the remaining objective terms and loss aggregation. Positive credits preserve each advantage's sign, emphasizing its original update direction at the selected positions.

This defines a weighted GRPO surrogate and generally changes the expected gradient relative to unweighted GRPO. Appendix B.6 formalizes this optimization interpretation, including the role of stop-gradient and the resulting gradient shift.

For binary outcome-only GRPO, homogeneous-outcome groups have zero advantage under the zero-variance convention. Their credit computation can be bypassed with unit weights because, for any finite multiplicative token credit,

$$\widehat{A}_{\tau,i} = 0 \quad \Longrightarrow \quad \widetilde{A}_{\tau,i} = c_{\tau,i}\widehat{A}_{\tau,i} = 0. \tag{9}$$

This bypass preserves these groups' zero contribution to the advantage-weighted policy term.

Our agentic experiments use the same outcome-conditioned credit rule with scalar training rewards: binary task outcomes select the flow channels, while the full rewards determine the advantages (Appendix B.7). Appendices B.8 and D.7 explore a further credit rule that uses relative outcome affinity for selective attenuation and sign reversal.

# 5 Experiments

We evaluate learning with outcome-conditioned cross-trajectory credit across mathematical reasoning, general reasoning, and agentic tasks, comparing against uniform GRPO supervision, trajectory-local Flow, and other credit-assignment methods. Appendices D.4, D.5, and D.8 characterize lexical diversity, variation across training seeds, and runtime.

## 5.1 Experimental Setup

**Models and Benchmarks.** Our models include Qwen3-4B/8B-Base and post-trained Qwen3 [42], as well as DeepSeek-R1-Distill-Llama-8B [5], covering both the Qwen and Llama families. For *mathematical reasoning*, we train on DAPO-Math-17K [45] and evaluate on AIME24, AIME25, AMC23 [21], MATH500 [12; 17], and OlympiadBench [11]. For *general reasoning*, we train on the CrossThinkQA training split [1] and evaluate on its held-out test split, MMLU-Pro [40], GPQA-Diamond [25], and BBEH [14]. For *agentic tasks*, we use HotpotQA [43], WebShop [44], and ALFWorld (sub/full) [30], training on their respective training instances and evaluating on held-out tasks. We report benchmark scores for reasoning and exact match or task success for agentic tasks, with unweighted averages; differences are in percentage points. Appendix A provides checkpoint and dataset details.

Table 3: Math reasoning on Qwen3-Base at 1K/8K context. Avg.: unweighted mean of five scores (%). **Bold**/underlined: best/second best per model and context.

| Method | *Context length 1K* | | | | | | *Context length 8K* | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AIME24 | AIME25 | AMC23 | MATH500 | Olymp. | Avg. | AIME24 | AIME25 | AMC23 | MATH500 | Olymp. | Avg. |
| *Qwen3-4B-Base* | | | | | | | | | | | | |
| GRPO | 8.4 | 5.2 | 55.1 | 74.2 | 42.8 | 37.1 | 19.5 | 16.1 | 57.6 | 81.0 | 49.9 | 44.8 |
| Random | 8.7 | 5.5 | 55.2 | 74.4 | 42.0 | 37.2 | 18.0 | 16.6 | 57.1 | 82.0 | 50.0 | 44.7 |
| Entropy | 9.1 | 5.0 | 54.9 | 73.6 | 42.4 | 37.0 | 19.0 | 18.7 | 58.0 | 80.3 | 49.4 | 45.1 |
| Gradient | 8.6 | 4.2 | 53.2 | 74.1 | 43.3 | 36.7 | 20.3 | 19.6 | 57.8 | 82.3 | 51.3 | 46.3 |
| CAPO | 8.7 | 5.4 | 58.4 | 74.2 | 43.0 | 37.9 | 19.3 | 20.0 | 61.2 | 81.6 | 51.4 | 46.7 |
| ThinkPRM | 8.9 | 6.2 | 58.6 | 75.4 | 44.1 | 38.6 | 20.4 | 21.3 | 60.0 | 82.0 | 51.0 | 46.9 |
| AsyPPO | 9.2 | 6.0 | 58.0 | 74.4 | 44.0 | 38.3 | 20.3 | 19.9 | 59.6 | 82.4 | 51.0 | 46.6 |
| TEMPO | 8.4 | 5.2 | 54.4 | 74.0 | 43.6 | 37.1 | 19.1 | 17.0 | 57.4 | 82.2 | 50.0 | 45.1 |
| Attention | 10.5 | 5.9 | 58.4 | 74.9 | 43.1 | 38.6 | 22.4 | 20.4 | 59.3 | 82.3 | 51.5 | 47.2 |
| Flow | 10.9 | 6.9 | 59.0 | 75.9 | 44.2 | 39.4 | 22.7 | 21.9 | 62.4 | 83.1 | 53.0 | 48.6 |
| **FlowWeaver** | **11.8** | **7.8** | **61.7** | **77.4** | **45.7** | **40.9** | **24.6** | **23.1** | **65.0** | **84.6** | **54.6** | **50.4** |
| Δ vs. GRPO (pp) | **+3.4** | **+2.6** | **+6.6** | **+3.2** | **+2.9** | **+3.8** | **+5.1** | **+7.0** | **+7.4** | **+3.6** | **+4.7** | **+5.6** |
| *Qwen3-8B-Base* | | | | | | | | | | | | |
| GRPO | 9.3 | 7.3 | 59.1 | 77.1 | 44.2 | 39.4 | 24.9 | 20.8 | 66.3 | 85.1 | 54.4 | 50.3 |
| Random | 8.9 | 8.1 | 60.1 | 77.4 | 43.3 | 39.6 | 25.1 | 21.3 | 67.2 | 85.5 | 55.3 | 50.9 |
| Entropy | 10.3 | 10.5 | 60.5 | 77.6 | 40.9 | 40.0 | 24.6 | 20.2 | 63.0 | 85.6 | 54.7 | 49.6 |
| Gradient | 9.5 | 8.0 | 60.0 | 77.3 | 45.0 | 40.0 | 23.7 | 20.8 | 65.7 | 84.2 | 53.4 | 49.6 |
| CAPO | 10.5 | 10.9 | 61.1 | 78.4 | 45.9 | 41.4 | 25.1 | 20.6 | 65.9 | 85.2 | 54.8 | 50.3 |
| ThinkPRM | 10.2 | 10.6 | 61.0 | 77.6 | 45.3 | 40.9 | 24.4 | 20.4 | 68.0 | 84.4 | 56.0 | 50.6 |
| AsyPPO | 9.6 | 8.2 | 59.5 | 77.8 | 44.7 | 40.0 | 25.2 | 20.4 | 68.3 | 85.7 | 55.8 | 51.1 |
| TEMPO | 8.5 | 6.1 | 60.4 | 77.4 | 45.3 | 39.5 | 23.3 | 20.9 | 67.0 | 85.5 | 55.0 | 50.3 |
| Attention | 10.1 | 9.3 | 62.9 | 78.4 | 45.9 | 41.3 | 25.6 | 21.6 | 66.1 | 85.9 | 55.5 | 50.9 |
| Flow | 13.0 | 11.8 | 65.6 | 79.7 | 46.7 | 43.4 | 26.1 | 22.4 | 70.4 | 86.7 | 56.7 | 52.5 |
| **FlowWeaver** | **13.9** | **13.0** | **68.0** | **80.4** | **47.4** | **44.5** | **27.6** | **23.0** | **73.8** | **88.7** | **58.1** | **54.2** |
| Δ vs. GRPO (pp) | **+4.6** | **+5.7** | **+8.9** | **+3.3** | **+3.2** | **+5.1** | **+2.7** | **+2.2** | **+7.5** | **+3.6** | **+3.7** | **+3.9** |

**Baseline Methods.** FlowWeaver builds on GRPO [29] to assign token-level credit. We compare against GRPO and four categories of baselines: (1) Random: upweights the same fraction of tokens selected at random; (2) token-weighting methods: Entropy [38], Gradient [10], Attention [16], and Flow [6]; (3) PPO/GRPO variants: AsyPPO [20], TEMPO [32], GTPO [31], and GiGPO [8]; and (4) process reward model (PRM) methods: CAPO [41], ThinkPRM [15], and AgentPRM [4]. Some baselines are used for reasoning, some for agentic tasks, and others for both, as indicated in the result tables. Appendix C.2 provides further method details.

**Training Details.** All GRPO-based methods use the same dynamic-sampling protocol to fill each update batch with groups whose training rewards have nonzero within-group variance. All RL results in the main text are averaged over three training seeds. Unless varied in ablations, FlowWeaver uses an exchange fraction $\beta = 0.5$, a selected-token fraction $\rho = 0.4$, and an amplification factor $\gamma = 1.5$ across all tasks. Optimization, sampling, agentic reward definitions, and implementation details are in Appendix C.1.

### 5.2 Main Results

**Mathematical reasoning gains over local flow grow with context length.** FlowWeaver achieves the highest average score in all four Qwen3 model–context settings in Table 3, outperforming both uniform GRPO updates and trajectory-local Flow, the strongest baseline. At context lengths 1K and 8K, its average scores are 40.9 and 50.4 on Qwen3-4B-Base, and 44.5 and 54.2 on Qwen3-8B-Base, respectively. The corresponding gains over GRPO are 3.8/5.6 and 5.1/3.9 points. Relative to Flow, the margin widens from 1.5 to 1.8 points on Qwen3-4B-Base and from 1.1 to 1.7 points on Qwen3-8B-Base as the context limit increases from 1K to 8K. Longer contexts allow more intermediate computations to share a single terminal reward, making the allocation of that supervision especially relevant. These results support the benefit of using alternative trajectories to refine token importance across both context lengths, with larger gains over Flow at 8K for both model sizes.

Table 4: Agentic results using Qwen3-4B-Base on HotpotQA/WebShop and post-trained Qwen3-4B on ALFWorld. Succ.: HotpotQA exact match or task success elsewhere; F1: token F1. Avg.: unweighted mean of four Succ. scores. **Bold**/underlined: best/second best.

| Method | HotpotQA | | WebShop | | $\text{ALFWorld}_{\text{sub}}$ | | $\text{ALFWorld}_{\text{full}}$ | | Avg. |
|---|---|---|---|---|---|---|---|---|---|
| | F1 | Succ. | Score | Succ. | Score | Succ. | Score | Succ. | |
| GRPO | 64.0 | 52.0 | 90.4 | 89.8 | 58.9 | 59.8 | 71.6 | 71.6 | 68.3 |
| Random | 65.1 | 51.6 | 84.8 | 85.9 | 56.2 | 57.8 | 68.7 | 69.4 | 66.2 |
| Entropy | 68.1 | 52.3 | 88.9 | 84.5 | 62.8 | 63.0 | 72.1 | 72.4 | 68.1 |
| Attention | 70.0 | 59.0 | 96.3 | 91.6 | 64.8 | 65.0 | 73.3 | 73.5 | 72.3 |
| Flow | 69.4 | 59.8 | 96.5 | 92.8 | 66.4 | 66.7 | 76.0 | 76.0 | 73.8 |
| GTPO | 69.1 | 57.8 | 92.0 | 90.6 | 64.4 | 64.8 | 73.8 | 74.6 | 72.0 |
| GiGPO | 70.2 | 59.4 | 93.6 | 92.4 | 62.8 | 63.2 | 75.8 | 76.0 | 72.8 |
| AsyPPO | 69.8 | 58.8 | 92.2 | 91.6 | 65.0 | 65.3 | 72.5 | 73.8 | 72.4 |
| AgentPRM | 70.0 | 59.9 | 92.5 | 90.4 | 62.4 | 62.7 | 75.6 | 76.0 | 72.3 |
| **FlowWeaver** | **72.6** | **60.9** | **98.0** | **94.0** | **67.6** | **68.4** | **82.8** | **82.8** | **76.5** |
| $\Delta$ vs. GRPO (pp) | **+8.6** | **+8.9** | **+7.6** | **+4.2** | **+8.7** | **+8.6** | **+11.2** | **+11.2** | **+8.2** |

Table 5: General reasoning on Qwen3-8B-Base trained on CrossThinkQA. Avg.: unweighted mean of four scores (%). **Bold**/underlined: best/second best.

| Method | CrossThink QA | MMLU-Pro | GPQA-Diamond | BBEH | Avg. |
|---|---|---|---|---|---|
| GRPO | 48.0 | 57.1 | 41.9 | 11.4 | 39.6 |
| Random | 47.6 | 56.4 | 40.9 | 11.4 | 39.1 |
| Entropy | 47.6 | 60.3 | 41.5 | 10.9 | 40.1 |
| Attention | 49.6 | 59.2 | 42.4 | 12.0 | 40.8 |
| Flow | 50.2 | 61.4 | 43.4 | 11.6 | 41.7 |
| **FlowWeaver** | **52.4** | **65.6** | **45.5** | **13.0** | **44.1** |
| $\Delta$ vs. GRPO (pp) | **+4.4** | **+8.5** | **+3.6** | **+1.6** | **+4.5** |

Table 6: DeepSeek-R1-Distill-Llama-8B at 1K context. Avg.: unweighted mean of five math scores (%). HotpotQA: token F1 and exact match (Succ.). **Bold**/underlined: best/second best.

| Method | AIME 24 | AIME 25 | AMC 23 | MATH 500 | Olymp. | Avg. | HotpotQA F1 | HotpotQA Succ. |
|---|---|---|---|---|---|---|---|---|
| GRPO | 7.5 | 7.5 | 42.4 | 62.8 | 28.9 | 29.8 | 66.6 | 53.9 |
| Random | 7.3 | 6.7 | 41.6 | 62.5 | 29.4 | 29.5 | 66.2 | 53.2 |
| Entropy | 7.8 | 6.7 | 42.8 | 63.2 | 30.4 | 30.2 | 67.2 | 54.4 |
| Attention | 7.9 | 7.5 | 44.2 | 64.6 | 30.8 | 31.0 | 68.1 | 57.2 |
| Flow | 9.0 | 7.7 | 45.6 | 64.2 | 31.9 | 31.7 | 68.3 | 57.2 |
| **FlowWeaver** | **10.4** | **9.2** | **49.0** | **67.8** | **34.2** | **34.1** | **69.8** | **59.4** |
| $\Delta$ vs. GRPO (pp) | **+2.9** | **+1.7** | **+6.6** | **+5.0** | **+5.3** | **+4.3** | **+3.2** | **+5.5** |

**Refining token credit improves agent success under task-level supervision.** Agentic tasks test whether cross-trajectory credit remains useful when intermediate decisions shape subsequent observations and task progress. FlowWeaver achieves the highest outcome rate across HotpotQA, WebShop, and both ALFWorld settings (Table 4), improving the average by 8.2 points over GRPO and 2.7 over Flow. The largest gain over Flow occurs on the full ALFWorld task mixture, which includes longer interactions and more varied tasks. These results extend the benefit from mathematical derivations to question answering and sequential interaction: using peers' routing structures and outcomes to choose where to emphasize updates also improves learning from task-level feedback, without requiring an external process judge.

**General reasoning gains extend beyond the training benchmark.** Mathematics offers a structured setting for reasoning, so we also examine whether the credit signal remains useful across broader domains. After training on CrossThinkQA, FlowWeaver leads all four general reasoning benchmarks in Table 5, with an average gain of 2.4 points over Flow. Improvements appear both on the held-out CrossThinkQA split and on MMLU-Pro, GPQA-Diamond, and BBEH, extending beyond the training benchmark. This breadth supports a general role for outcome-conditioned cross-trajectory attribution: alternative computations and their outcomes provide useful context for allocating token-level supervision across tasks with different knowledge and reasoning demands.

**Cross-trajectory credit complements reasoning distillation on Llama.** DeepSeek-R1-Distill-Llama-8B extends the evaluation beyond the Qwen family to a model that has already acquired reasoning behavior through distillation. Table 6 shows that FlowWeaver leads all five math benchmarks and HotpotQA in this setting. At the same 8B scale and 1K context length, its average math gain over Flow is 2.2 points, compared with 1.1 on Qwen3-8B-Base. The larger gain in this setting suggests that cross-trajectory credit is complementary to existing reasoning ability: even a model already trained to produce reasoning traces benefits from using alternative trajectories to refine where subsequent RL updates are concentrated.

Table 7: Group-size ablation on Qwen3-4B-Base (8K). Scores: unweighted mean over five math benchmarks (%); $\Delta$: gain over GRPO (pp). $G$: responses per prompt; $N$ and $R$: retained prompts and responses per update. Shading (†): shared $G = 8$ reference.

| | **(a) Fixed responses** $R = R_0, \quad N = R_0/G$ | | | **(b) Fixed prompts** $N = N_0, \quad R = GN_0$ | | |
|---|---|---|---|---|---|---|
| $G$ | GRPO | FlowWeaver | $\Delta$ | GRPO | FlowWeaver | $\Delta$ |
| 4 | 44.0 | 49.7 | +5.7 | 44.9 | 50.2 | +5.3 |
| **8**† | 44.8 | 50.4 | +5.6 | 44.8 | 50.4 | +5.6 |
| 16 | 44.2 | 49.8 | +5.6 | 45.2 | 50.7 | +5.5 |
| 32 | 43.4 | 49.0 | +5.6 | 45.0 | 50.5 | +5.5 |

Table 8: Hyperparameter and design ablations of FlowWeaver on Qwen3-4B-Base (1K). Avg.: mean over five math benchmarks (%). Shading (†): shared default; **bold**: panel best. Top-$k$/Bottom-$k$ rank tokens by conditioned flow; Upscale denotes multiplier $\gamma$.

| **(a) Coupling** $\beta$ | Avg. ↑ |
|---|---|
| 0.1 | 39.6 |
| 0.3 | 40.1 |
| 0.5† | **40.9** |
| 0.7 | 39.4 |
| 0.9 | 39.7 |

| **(b) Selection** | $\rho$ | Avg. ↑ |
|---|---|---|
| Top-$k$ | 20% | 40.3 |
| Top-$k$ | 40%† | **40.9** |
| Top-$k$ | 60% | 39.9 |
| Bottom-$k$ | 20% | 36.2 |
| Bottom-$k$ | 40% | 36.9 |

| **(c) Weighting** | Avg. ↑ |
|---|---|
| Upscale 1.2 | 39.4 |
| Upscale 1.5† | **40.9** |
| Upscale 1.8 | 39.6 |
| Raw flow | 33.8 |
| Sigmoid (raw) | 37.1 |

### 5.3 Ablation Studies

**Small response groups already capture most of the cross-trajectory gain.** Group size affects both the peers available for attribution and, under a fixed response budget, the number of distinct problems per update. Across the tested sizes, FlowWeaver retains a 5.3–5.7-point advantage over GRPO (Table 7), suggesting that its benefit does not require many peers. Increasing $G$ beyond eight does not improve performance under a fixed response budget, where it also reduces prompt diversity. With prompts fixed, doubling or quadrupling the response count adds at most 0.3 points to FlowWeaver's average. The limited return from larger groups makes $G = 8$ a practical balance.

**Moderate exchange and selective weighting favor the full method.** Table 8 favors balanced native and peer routing ($\beta = 0.5$); weaker or stronger exchange reduces performance. The clearest contrast concerns which tokens receive extra emphasis: at the same 40% budget, emphasizing high-flow rather than low-flow positions improves the average by 4.0 points. Changing the fraction of top-ranked tokens has a smaller effect, suggesting that where credit is concentrated matters more than the exact selection fraction. Moderate amplification also performs best among the tested factors, while raw-flow and sigmoid weighting trail the selective rule. Together, these results favor moderate exchange and selective amplification of high-flow tokens in this setting.

Appendix D.1 further examines this attribution design across exchange strengths and correspondence rules, using unconditioned shared flow as a control for outcome targeting. The full method outperforms this control at every tested exchange strength. Within the full method at the default configuration, cumulative-flow also outperforms token progress, step progress, and embedding cosine similarity, with a 2.0-point advantage over the strongest alternative correspondence, token progress. Additional studies cover **group sampling and outcome composition**, **credit normalization**, **layer selection**, and **exploratory outcome-affinity adjustment** (Appendices D.2, D.3, D.6, and D.7).

## 6 Conclusion

We presented FlowWeaver, which constructs a shared attribution graph across sampled responses for propagation from the prompt to different outcomes. Conditioning this propagation on each response's observed outcome yields token-importance scores based on the attention structure of both successful and unsuccessful responses. Our analyses link flow correspondence to similar prefix outcome certainty and token rankings to outcome sensitivity under masking. In RL experiments, using this shared outcome-conditioned attribution to weight GRPO supervision improves performance across mathematical reasoning, general reasoning, and agentic tasks.

### AI USE STATEMENT

We used generative AI tools in three parts of this work: drafting and editing the text of this paper, implementing code, and running and analyzing experiments. We did not use generative AI tools for research ideation. The problem formulation, the outcome-conditioned information-flow framework, the credit-assignment method, and the claims made are the authors' own. All AI-assisted text was carefully checked and revised as necessary by the authors. We take full responsibility for the final content of this work, including any text or other artifacts produced with the aid of generative AI.

### ETHICS STATEMENT

Our work studies credit assignment in reinforcement learning for large language models using attention-derived attribution signals. The proposed method is intended to improve the efficiency and effectiveness of training reasoning models, while providing a structured way to allocate outcome supervision across tokens using sampled trajectories. The scores may also support analysis of token selection during training. At the same time, improvements in reasoning capability are general-purpose and could potentially strengthen models used in undesirable or harmful applications. We therefore view our method as complementary to, rather than a replacement for, existing safety and alignment practices. Our experiments do not involve human subjects or private user data, and we use standard research benchmarks and models under their respective licenses and usage policies.

### REPRODUCIBILITY STATEMENT

To support reproducibility, we provide detailed method definitions, document the experimental setup and analysis protocols, and report additional evaluations of robustness. Our method description specifies flow correspondence, cross-trajectory coupling, outcome conditioning, and GRPO token weighting, with supporting derivations and computational conventions (Section 4 and Appendix B). For the experiments, we identify the model checkpoints, datasets, and environments with public reference links (Appendix A). We document optimization and sampling settings, training schedules, hyperparameter selection, reward definitions, attention extraction, and flow computation, together with baseline settings and protocols for the prefix-continuation and token-masking analyses (Appendix C). We also report ablations and sensitivity analyses to assess the effects of design choices (Section 5.3 and Appendix D), including means and standard deviations over five training seeds (Appendix D.5). We intend to release the code and experimental configurations following the review process to further facilitate reproduction and future research.

## REFERENCES

[1] Syeda Nahida Akter, Shrimai Prabhumoye, Matvei Novikov, Seungju Han, Ying Lin, Evelina Bakhturina, Eric Nyberg, Yejin Choi, Mostofa Patwary, Mohammad Shoeybi, et al. Nemotron-crossthink: Scaling self-learning beyond math reasoning. *arXiv preprint arXiv:2504.13941*, 2025.

[2] Paul C Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought anchors: Which llm reasoning steps matter? *arXiv preprint arXiv:2506.19143*, 2025.

[3] Vivien Cabannes, Charles Arnal, Wassim Bouaziz, Xingyu Yang, Francois Charton, and Julia Kempe. Iteration head: A mechanistic study of chain-of-thought. *Advances in Neural Information Processing Systems*, 37:109101–109122, 2024.

[4] Sanjiban Choudhury. Process reward models for LLM agents: Practical framework and directions. *arXiv preprint arXiv:2502.10325*, 2025.

[5] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[6] Zhichen Dong, Yang Li, Yuhan Sun, Weixun Wang, Yijia Luo, Zinian Peng, Taiheng Ye, Chao Yang, Wenbo Su, Yu Cheng, Bo Zheng, and Junchi Yan. How does reasoning flow? tracing attention-induced information flow for targeted RL in LLMs. *arXiv preprint arXiv:2606.10646*, 2026.

[7] Subhabrata Dutta, Joykirat Singh, Soumen Chakrabarti, and Tanmoy Chakraborty. How to think step-by-step: A mechanistic understanding of chain-of-thought reasoning. *arXiv preprint arXiv:2402.18312*, 2024.

[8] Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. *arXiv preprint arXiv:2505.10978*, 2025.

[9] Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12216–12235, 2023.

[10] Annabel Green, Andrew Delta, Carter Wilson, Eleanor Wragge, Andrew Scolto, and Benedict Wickersham. Contextual gradient recomposition for sequential coherence preservation in large language model token generation. ResearchGate preprint, August 2025. URL https://www.researchgate.net/publication/394442653_Contextual_Gradient_Recomposition_for_Sequential_Coherence_Preservation_in_Large_Language_Model_Token_Generation.

[11] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

[12] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. *arXiv preprint arXiv:2103.03874*, 2021.

[13] Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In Trevor Cohn, Yulan He, and Yang Liu (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1627–1643, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.147. URL https://aclanthology.org/2020.findings-emnlp.147/.

[14] Mehran Kazemi, Bahare Fatemi, Hritik Bansal, John Palowitch, Chrysovalantis Anastasiou, Sanket Vaibhav Mehta, Lalit K. Jain, Virginia Aglietti, Disha Jindal, Peter Chen, et al. BIG-Bench Extra Hard. *arXiv preprint arXiv:2502.19187*, 2025.

[15] Muhammad Khalifa, Rishabh Agarwal, Lajanugen Logeswaran, Jaekyeom Kim, Hao Peng, Moontae Lee, Honglak Lee, and Lu Wang. Process reward models that think. *arXiv preprint arXiv:2504.16828*, 2025.

[16] Yang Li, Zhichen Dong, Yuhan Sun, Weixun Wang, Shaopan Xiong, Yijia Luo, Jiashun Liu, Han Lu, Jiamang Wang, Wenbo Su, Bo Zheng, and Junchi Yan. Attention illuminates LLM reasoning: The preplan-and-anchor rhythm enables fine-grained policy optimization. *arXiv preprint arXiv:2510.13554*, 2025.

[17] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

[18] Zicheng Lin, Tian Liang, Jiahao Xu, Qiuzhi Lin, Xing Wang, Ruilin Luo, Chufan Shi, Siheng Li, Yujiu Yang, and Zhaopeng Tu. Critical tokens matter: Token-level contrastive estimation enhances llm's reasoning capability. *arXiv preprint arXiv:2411.19943*, 2024.

[19] Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

[20] Jiashun Liu, Johan Obando-Ceron, Han Lu, Yancheng He, Weixun Wang, Wenbo Su, Bo Zheng, Pablo Samuel Castro, Aaron Courville, and Ling Pan. Asymmetric proximal policy optimization: Mini-critics boost LLM reasoning. *arXiv preprint arXiv:2510.01656*, 2025.

[21] MAA. American mathematics competitions. Mathematical Association of America. https://maa.org/student-programs/amc/, n.d.

[22] Tianyi Men, Pengfei Cao, Zhuoran Jin, Yubo Chen, Kang Liu, and Jun Zhao. Unlocking the future: Exploring look-ahead planning mechanistic interpretability in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7713–7724, 2024.

[23] Hosein Mohebbi, Willem Zuidema, Grzegorz Chrupała, and Afra Alishahi. Quantifying context mixing in transformers. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 3378–3400, 2023.

[24] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[25] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

[26] Jie Ren, Qipeng Guo, Hang Yan, Dongrui Liu, Quanshi Zhang, Xipeng Qiu, and Dahua Lin. Identifying semantic induction heads to understand in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 6916–6932, 2024.

[27] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

[29] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[30] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021.

[31] Hongze Tan and Jianfei Pan. GTPO and GRPO-S: Token and sequence-level reward shaping with policy entropy. *arXiv preprint arXiv:2508.04349*, 2025. URL https://arxiv.org/abs/2508.04349.

[32] Hieu Tran, Zonghai Yao, and Hong Yu. Exploiting tree structure for credit assignment in RL training of LLMs. *arXiv preprint arXiv:2509.18314*, 2025.

[33] Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

[34] Jean Vassoyan, Nathanaël Beau, and Roman Plaud. Ignore the kl penalty! boosting exploration on critical tokens to enhance rl fine-tuning. *arXiv preprint arXiv:2502.06533*, 2025.

[35] Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. Understanding reasoning in thinking language models via steering vectors. *arXiv preprint arXiv:2506.18167*, 2025.

[36] Jiawei Wang, Jiacai Liu, Yuqian Fu, Yingru Li, Xintao Wang, Yuan Lin, Yu Yue, Lin Zhang, Yang Wang, and Ke Wang. Harnessing uncertainty: Entropy-modulated policy gradients for long-horizon llm agents. *arXiv preprint arXiv:2509.09265*, 2025.

[37] Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024.

[38] Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. *arXiv preprint arXiv:2506.01939*, 2025.

[39] Weixun Wang, Shaopan Xiong, Gengru Chen, Wei Gao, Sheng Guo, Yancheng He, Ju Huang, Jiaheng Liu, Zhendong Li, Xiaoyang Li, et al. Reinforcement learning optimization for large-scale learning: An efficient and user-friendly scaling library. *arXiv preprint arXiv:2506.06122*, 2025.

[40] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024.

[41] Guofu Xie, Yunsheng Shi, Hongtao Tian, Ting Yao, and Xiao Zhang. CAPO: Towards enhancing LLM reasoning through verifiable generative credit assignment. *arXiv preprint arXiv:2508.02298*, 2025.

[42] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[43] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, 2018.

[44] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, pp. 20744–20757, 2022.

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

[46] Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

[47] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1904.09675.

[48] Chengfeng Zhao, Shizhu He, Shanshan Jiang, Bin Dong, Jun Zhao, and Kang Liu. Why and how LLMs benefit from knowledge introspection in commonsense reasoning. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 6905–6920, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.352. URL https://aclanthology.org/2025.emnlp-main.352/.

## APPENDIX ROADMAP

Appendix A lists model checkpoints, datasets, and environments. Appendix B gives the construction and mathematical properties of outcome-conditioned batch flow, including its optimization interpretation (Appendix B.6) and use with scalar training rewards (Appendix B.7). Appendix C documents training, sampling, hyperparameter selection, agentic rewards, memory management, baselines, and intervention protocols. Appendix D provides the additional experiments summarized below. The design ablations use Qwen3-4B-Base in the specified settings; numerical gains below are in percentage points.

- **Outcome conditioning and flow correspondence** (Appendix D.1, Table 11). Outcome-conditioned flow outperforms the unconditioned shared-flow control at every tested exchange strength. Cumulative-flow correspondence leads the tested correspondence rules, exceeding the strongest alternative, token progress, by 2.0 points.
- **Credit normalization and uniform scaling** (Appendix D.3, Table 13). Normalizing mean credit to one retains 5.1 of the main method's 5.6-point gain over GRPO. Uniformly scaling GRPO does not improve its score; flow-guided allocation is effective at both tested credit scales.
- **Group sampling and outcome composition** (Appendix D.2, Table 12). At group size eight, same-outcome peer replacement displaces 0.70–0.98% of all focal-response tokens from the selected set; opposite-outcome replacement displaces 1.01–3.02%. Selection changes remain small, with larger shifts when the outcome composition changes.
- **Five training seeds across all main-result settings** (Appendix D.5, Table 15). Five-seed comparisons of GRPO, Flow, and FlowWeaver cover all 11 main-result settings. FlowWeaver retains the highest mean on every reported reasoning benchmark and agentic outcome metric, with standard deviations comparable to Flow.
- **Attention layer selection** (Appendix D.6, Table 16). Middle-layer attention gives the highest average score among the four tested choices (40.9%, versus 40.6% for all layers), supporting the default middle-third selection.
- **Lexical diversity** (Appendix D.4, Table 14). Pairwise BLEU-4 on MATH-500 remains comparable to GRPO (69.57% versus 69.56%), indicating similar surface-form diversity in this setting.
- **Runtime and memory overhead** (Appendices D.8 and C.1, Table 18). The complete attribution pipeline, including attention extraction, accounts for 1.45–2.17% of training-step time on mathematical reasoning and 0.33–0.68% on agentic tasks in the measured configurations. A fixed-length 8K execution benchmark peaks at 8.848 GiB GPU allocated memory and 5.722 GiB sampled CPU process PSS with two groups in flight; dense graph storage remains quadratic in sequence length.
- **Exploratory outcome-affinity adjustment** (Appendix D.7, Table 17). Limited attenuation or sign reversal adds 0.3–0.6 points in the tested setting, while broader or stronger adjustments can degrade performance. All main experiments use the positive-weighting rule.

Related main-text ablations appear in Section 5.3: Table 7 shows a 5.3–5.7-point gain over GRPO across group sizes 4–32 under fixed-response and fixed-prompt budgets; Table 8 favors moderate exchange and selective amplification of high-flow tokens among the tested choices.

# A Dataset and Model Reference

We summarize the policy checkpoints, training data, and evaluation resources used in Section 5. Tables 9 and 10 provide the names and public reference links.

## A.1 Models

The mathematical reasoning experiments use Qwen3-4B-Base and Qwen3-8B-Base [42] at context lengths of 1K and 8K. Qwen3-8B-Base is also used for general reasoning. For the main Qwen agentic experiments, HotpotQA and WebShop use Qwen3-4B-Base; only $\text{ALFWorld}_{\text{sub}}$ and $\text{ALFWorld}_{\text{full}}$ use the post-trained Qwen3-4B checkpoint. In our ALFWorld setup, action-format validity with the base checkpoint remained at 0%, so the post-trained initialization is needed to enable policy training. To evaluate a different model family and initialization, we use DeepSeek-R1-Distill-Llama-8B [5], which was obtained by fine-tuning Llama-3.1-8B-Base on reasoning data generated by DeepSeek-R1. This model is evaluated on the five mathematical reasoning benchmarks at 1K and on HotpotQA. We additionally use post-trained Qwen3-8B for the $\text{ALFWorld}_{\text{full}}$ runtime measurement in Appendix D.8.

Table 9: Policy models used in our experiments. Links display the Hugging Face repository identifiers and point to the public checkpoint releases.

| Model | Variant | Experiments | Link |
|---|---|---|---|
| Qwen3-4B-Base | Base | Math reasoning, HotpotQA, and WebShop | `Qwen/Qwen3-4B-Base` |
| Qwen3-8B-Base | Base | Math and general reasoning | `Qwen/Qwen3-8B-Base` |
| Qwen3-4B | Post-trained | ALFWorld (sub/full) | `Qwen/Qwen3-4B` |
| Qwen3-8B | Post-trained | ALFWorld (full), runtime analysis | `Qwen/Qwen3-8B` |
| DeepSeek-R1-Distill-Llama-8B | Distilled | Math reasoning and HotpotQA | `deepseek-ai/DeepSeek-R1-Distill-Llama-8B` |

## A.2 Datasets and Environments

**Training corpora.** DAPO-Math-17K [45] is a curated collection of approximately 17,000 mathematical problems with reference answers. We use it for mathematical reasoning training with answer-based outcome rewards. For general reasoning, we use CrossThinkQA [1], publicly released as *CrossThink-QA*, training on its training split and evaluating on its held-out test split. It brings together question-answer data across STEM, the humanities, and the social sciences, including multiple-choice and open-ended formats with verifiable answers. This setting extends outcome-supervised training beyond mathematics.

**Mathematical reasoning benchmarks.** AIME24 and AIME25 contain problems from the corresponding years of the American Invitational Mathematics Examination, whose answers are integers; they require several steps of mathematical reasoning. AMC23 provides competition problems from the 2023 American Mathematics Competitions. MATH500 is a 500-problem subset of the MATH test set [12], covering topics such as algebra, geometry, number theory, and counting and probability. OlympiadBench [11], abbreviated as *Olympiad* or *Olymp.* in our result tables, adds challenging Olympiad-level problems to the mathematical evaluation. The full public benchmark includes mathematics and physics in multiple languages and modalities; our results concern mathematical reasoning.

**General reasoning benchmarks.** Alongside the CrossThinkQA test split, we evaluate on MMLU-Pro, GPQA-Diamond, and BIG-Bench Extra Hard (BBEH). MMLU-Pro [40] contains reasoning-focused multiple-choice questions across 14 domains, with up to ten answer options. GPQA-Diamond is the high-quality subset of GPQA [25], which contains graduate-level questions in biology, chemistry, and physics. BBEH [14] extends the scope of evaluation to diverse reasoning

Table 10: Datasets and interactive environments used in our experiments. Links display the repository identifiers on Hugging Face (data) or GitHub (benchmark projects and environments).

| Dataset / environment | Use in this work | Link |
|---|---|---|
| *Training data and in-domain evaluation* | | |
| DAPO-Math-17K | Training; mathematical reasoning | `BytedTsinghua-SIA/DAPO-Math-17k` |
| CrossThinkQA | Training and test splits; general reasoning | `nvidia/CrossThink-QA` |
| *Mathematical reasoning evaluation* | | |
| AIME24 | 2024 American Invitational Mathematics Examination | `math-ai/aime24` |
| AIME25 | 2025 American Invitational Mathematics Examination | `math-ai/aime25` |
| AMC23 | 2023 American Mathematics Competitions problems | `math-ai/amc23` |
| MATH500 | 500-problem subset of the MATH test set | `HuggingFaceH4/MATH-500` |
| OlympiadBench | Olympiad-level mathematical reasoning | `OpenBMB/OlympiadBench` |
| *General reasoning evaluation* | | |
| MMLU-Pro | Broad-domain knowledge and reasoning | `TIGER-Lab/MMLU-Pro` |
| GPQA-Diamond | Expert-level scientific question answering | `idavidrein/gpqa` |
| BBEH | Diverse challenging reasoning tasks | `google-deepmind/bbeh` |
| *Agentic tasks* | | |
| HotpotQA | Multi-hop question answering | `hotpotqa/hotpot` |
| WebShop | Shopping from natural-language instructions | `princeton-nlp/WebShop` |
| ALFWorld | Household tasks; sub/full settings | `alfworld/alfworld` |

abilities by replacing the tasks in BIG-Bench Hard with more demanding counterparts. Together, these benchmarks assess both domain knowledge and reasoning beyond competition mathematics.

**Agentic tasks.** HotpotQA [43] requires combining information from multiple Wikipedia passages to answer a question and provides supporting-fact annotations. We report answer exact match and token F1. WebShop [44] is a simulated shopping environment in which an agent searches, browses, and selects products that satisfy a natural-language request. It tests instruction understanding and sequential decisions under product constraints. ALFWorld [30] provides text-based household environments in which an agent navigates and manipulates objects to accomplish a goal. Our ALFWorld-Lite setting, denoted $\text{ALFWorld}_{\text{sub}}$, selects shorter single-object games whose solutions fit the action horizon; $\text{ALFWorld}_{\text{full}}$ retains the full task mixture, including longer-horizon tasks. Both use the same underlying environment. For WebShop and ALFWorld, we report task success and the corresponding task score on held-out tasks.

## B Details and Properties of Outcome-Conditioned Batch Flow

This appendix gives the detailed construction and properties of the coupled routing and outcome-conditioned flow in Section 4.1. Appendix B.1 establishes the coverage, monotonicity, and sparsity of the exchange correspondence. Appendix B.2 proves that the coupled routing conserves mass and remains acyclic, ensuring finite absorption at answer boundaries. Appendix B.3 derives conditioning on terminal outcomes and shows why aggregating boundaries by outcome preserves token flows. Appendix B.4 defines token flow with a fixed prompt source and gives its backward and forward computation. Appendix B.5 clarifies limiting cases and the scope of the flow interpretation. Appendix B.6 connects token credits to a weighted optimization surrogate and characterizes the gra-

dient shift relative to GRPO. Appendix B.7 explains how binary outcome attribution combines with scalar training rewards. Finally, Appendix B.8 introduces selective downscaling and sign reversal as an exploratory extension of the main credit rule.

Each trajectory has distinct token occurrences, including its prompt tokens, and its own absorbing answer boundary $z_k$. Let $V$ denote the set of nonterminal nodes across trajectories. Native transitions between these nodes are strictly forward in token order, and each retained native graph reaches its answer boundary. The source $\sigma_k$ has unit mass on prompt tokens. All statements concern the auxiliary information-flow graph.

### B.1 Properties of the Exchange Correspondence

**Intervals and their overlap.** For the response tokens $\mathcal{R}_k$ of trajectory $k$, let $S_k = \sum_{u\in\mathcal{R}_k} f_k^0(u) > 0$ and $\mu_k(u) = f_k^0(u)/S_k$. Accumulating these normalized reference masses in token order gives intervals whose overlaps define the exchange weights:

$$I_k(u) = \left( \sum_{\substack{w\in\mathcal{R}_k \\ w\prec u}} \mu_k(w), \sum_{\substack{w\in\mathcal{R}_k \\ w\preceq u}} \mu_k(w) \right], \qquad C_{k\ell}(u,v) = \frac{|I_k(u)\cap I_\ell(v)|}{\mu_k(u)}, \qquad \mu_k(u) > 0. \tag{10}$$

Here $\prec$ denotes token order and $|I|$ is interval length. For convenience, write the left endpoint as $a_k(u) = \sum_{w\prec u} \mu_k(w)$, with the sum over response tokens. Then $I_k(u) = (a_k(u), a_k(u) + \mu_k(u)]$, and the intervals of each trajectory partition $(0, 1]$, up to empty intervals. The overlap mass is the finite matrix

$$\Gamma_{k\ell}(u,v) = |I_k(u)\cap I_\ell(v)|. \tag{11}$$

Thus $C_{k\ell}$ is the row-normalized overlap, rather than an additional matching rule to be chosen. Equivalently, draw a position uniformly from $I_k(u)$ and select the token whose interval contains that position in trajectory $\ell$. This gives the same conditional correspondence weights.

**Full coverage and preserved marginals.** The overlap masses satisfy

$$\sum_v \Gamma_{k\ell}(u,v) = \mu_k(u), \qquad \sum_u \Gamma_{k\ell}(u,v) = \mu_\ell(v). \tag{12}$$

*Proof.* Intersecting a fixed interval in one trajectory with the partition of the other divides it into disjoint pieces whose lengths sum to its own length. Applying this argument in each direction proves both identities. □

Consequently, every positive-flow source row is fully covered, and transporting normalized reference mass recovers the target reference mass:

$$\sum_v C_{k\ell}(u,v) = 1, \qquad \sum_u \mu_k(u) C_{k\ell}(u,v) = \mu_\ell(v). \tag{13}$$

The two directions also satisfy $\mu_k(u)C_{k\ell}(u,v) = \mu_\ell(v)C_{\ell k}(v,u)$, although $C_{k\ell}$ itself need not be symmetric. Normalization makes the correspondence invariant to multiplying all reference flows within a trajectory by the same positive constant. It does not imply that different trajectories have equal unnormalized total visit mass.

**Noncrossing and canonical correspondence.** If $u \prec u'$ in trajectory $k$, positive overlaps cannot simultaneously connect $u$ to $v$ and $u'$ to $v'$ with $v' \prec v$ in trajectory $\ell$. Indeed, the first intersection would lie before the second according to trajectory $k$, but after it according to trajectory $\ell$, a contradiction. This is the monotone coupling of the two ordered token-mass distributions. For any token prefixes ending at $u$ and $v$, its cumulative joint mass is

$$\sum_{w\preceq u}\sum_{t\preceq v} \Gamma_{k\ell}(w,t) = \min\{ \sum_{w\preceq u} \mu_k(w), \sum_{t\preceq v} \mu_\ell(t)\}. \tag{14}$$

For any noncrossing coupling, the two opposite off-diagonal prefix rectangles cannot both carry mass, which forces the same minimum identity. These cumulative masses determine all entries by successive differences, so the ordered marginals determine the monotone coupling uniquely. No semantic matching objective, distance threshold, or anchor selector is required. The correspondence is determined by cumulative reference-flow order.

**One-to-many matching and flow-weighted exchange mass.** A wide interval can overlap several narrower intervals in another trajectory. More generally, splitting an interval into adjacent subintervals preserves its aggregate overlap with every target interval, by additivity of length. This is a property of the coupling at fixed reference intervals, not an invariance claim for the full method under arbitrary retokenization. Under the normalized reference mass, the amount allocated from $u$ to a particular other trajectory is

$$\sum_v \frac{\beta}{B-1}\mu_k(u)C_{k\ell}(u,v) = \frac{\beta}{B-1}\mu_k(u). \tag{15}$$

Thus all-token coupling does not give equal coupling mass to every token: higher-flow tokens account for more of this reference mass. The exchange *fraction per visit* is nevertheless the same $\beta$ at every coupled token. For an actual visit mass $f(u)$, the pre-conditioning exchange component has mass $\beta f(u)$; the eventual coupled or conditioned visit mass need not be proportional to $\mu_k(u)$.

**Sparse construction without thresholding.** If trajectories $k$ and $\ell$ contain $n_k$ and $n_\ell$ positive-flow tokens, their common interval refinement has at most $n_k + n_\ell - 1$ pieces. Each nonempty overlap is one such piece, so

$$\mathrm{nnz}(C_{k\ell}) \le n_k + n_\ell - 1. \tag{16}$$

A two-pointer sweep over the ordered interval endpoints constructs these entries in $O(n_k + n_\ell)$ time without forming a dense token-pair matrix. All-token coverage is therefore distinct from dense all-to-all connectivity. This bound applies to the correspondence matrices; the composed native routing kernel may have more nonzero entries.

**Exact zero-flow cases.** A token with $f_k^0(u) = 0$ has an empty interval and contributes no overlap. Its exchange row is bypassed rather than divided by zero; its native routing can be retained. This is exact zero-mass handling, not a positive-flow threshold. Such nodes cannot be reached from the prescribed source through positive native-flow nodes: the reference-flow recursion implies that a positive-flow node cannot have a positive-probability native transition into a zero-flow node. Nor can overlap exchange select an empty interval. They may therefore be omitted from the source-reachable flow computation. Normalization applies to nonempty source-reachable responses with $S_k > 0$; no artificial interval is assigned when the response has no flow mass.

### B.2 Conservation and Global Acyclicity

**Boundary conventions and local conservation.** Use Eq. equation 4 on positive-flow response nodes. Prompt nodes and exact zero-flow response nodes retain their native rows, and $P_{\mathcal{B}}(z_k, z_k) = 1$. Native rows are understood as distributions over the disjoint batch state space, with zero entries outside their own trajectories. For $B = 1$, set $P_{\mathcal{B}} = P_1$. For $B > 1$, Eq. equation 13 gives

$$\sum_w P_{\mathcal{B}}(u,w) = (1-\beta) + \frac{\beta}{B-1}\sum_{\ell \ne k} 1 = 1. \tag{17}$$

All entries are nonnegative for $\beta \in [0,1]$, so no residual unmatched-mass term is needed.

**Proposition: overlap followed by native propagation is acyclic.** On positive-flow response nodes, every nonterminal transition of $P_{\mathcal{B}}$ strictly increases the interval left endpoint. Consequently, its source-reachable transient graph is a DAG for every $\beta \in [0,1]$, including $\beta = 1$.

*Proof.* A native transition from $u$ to a later response token $w$ satisfies $a_k(w) \ge a_k(u) + \mu_k(u) > a_k(u)$. For a borrowed continuation, suppose $C_{k\ell}(u,v)P_\ell(v,w) > 0$, where $w$ is nonterminal. Positive overlap implies that the right endpoint of $I_\ell(v)$ exceeds $a_k(u)$. Since native propagation is strictly forward, $w$ lies after $v$, and hence

$$a_\ell(w) \ge a_\ell(v) + \mu_\ell(v) > a_k(u). \tag{18}$$

Thus all response-to-response transitions strictly increase the same scalar coordinate. Prompt nodes can be placed before response nodes, in their native token order, because they do not exchange and response nodes cannot return to them. Answer boundaries are placed last. This provides a topological order for the source-reachable graph. Exact zero-flow nodes are unreachable from the source, as established above. □

In a finite row-stochastic graph with no transient cycles or nonterminal sinks, every path reaches an answer boundary in finitely many transitions. Thus no additional spectral-radius assumption or restriction $\beta < 1$ is needed for this particular construction. If $Q_{\mathcal{B}}$ is its transient block on the retained states, it is strictly upper triangular in the shared order and therefore nilpotent. The result relies on *exchange followed by a strictly forward native step*: overlap exchange alone need not increase the left endpoint. It does not automatically apply to implementations with native token self-loops, backward native edges, or exchange treated as a separately repeatable transition.

### B.3 Outcome Conditioning and Terminal Aggregation

**Outcome-hitting probabilities.** Keep the trajectory-specific boundaries $Z = \{z_1, \dots, z_B\}$ and write $Z_b = \{z_k : o_k = b\}$ for $b \in \{0, 1\}$. Let $T_Z$ be the first time the coupled process reaches $Z$. The outcome-hitting probability is

$$h_b(u) = \Pr_{P_{\mathcal{B}}}(X_{T_Z} \in Z_b \mid X_0 = u). \tag{19}$$

The terminal labels set the boundary values, and conditioning on the next transition gives the nonterminal recursion:

$$h_b(z_k) = \mathbf{1}\{o_k = b\}, \qquad h_b(u) = \sum_v P_{\mathcal{B}}(u, v) h_b(v). \tag{20}$$

The DAG property makes this boundary-value problem uniquely solvable by backward propagation. Absorption at a binary-labeled boundary gives $h_0 + h_1 = 1$, so one backward solve suffices for both potentials.

**Aggregating boundaries is optional and exact.** The $B$ terminals can equivalently be merged into the two outcome classes, with incoming transition probabilities

$$d_b(u) = \sum_{k : o_k = b} P_{\mathcal{B}}(u, z_k), \qquad b \in \{0, 1\}. \tag{21}$$

The transient recursion becomes $h_b(u) = \sum_{v \in V} P_{\mathcal{B}}(u, v) h_b(v) + d_b(u)$. Merging terminals leaves this recursion, the conditioned transient kernels, and the token flows unchanged. It discards only the identity of the individual terminal eventually reached. Here $\mathcal{R}_k$ denotes response tokens, not a terminal class.

**Conservation on the reachable support.** Let $V_b = \{u \in V : h_b(u) > 0\}$. For $u \in V_b$, Eq. equation 5 gives

$$\sum_{v \in V \cup Z} P_{\mathcal{B}}^{(b)}(u, v) = \frac{\sum_v P_{\mathcal{B}}(u, v) h_b(v)}{h_b(u)} = 1. \tag{22}$$

All transitions into states with $h_b = 0$ have zero conditioned probability. The transform is therefore a conservative kernel on $V_b \cup Z_b$, with the target terminals absorbing. It retains a subset of the base edges and hence preserves acyclicity. No transform row is needed outside this support when the injected sources all lie in $V_b$.

**Path interpretation.** Consider a base-supported path $\omega = (s, \dots, z_k)$ stopped at its first terminal visit, with $h_b(s) > 0$. If $z_k \in Z_b$, all intermediate potentials are positive and the potential ratios telescope. If $z_k \notin Z_b$, the conditioned process assigns zero probability to the path. Together these give

$$\overset{(b)}{\Pr}(\omega \mid X_0 = s) = \Pr(\omega \mid X_0 = s) \frac{\mathbf{1}\{o_k = b\}}{h_b(s)}. \tag{23}$$

This is exact conditioning on terminal outcome $b$ for each fixed start. The target remains fixed as the path moves across trajectories, including those whose original outcome differs from $b$.

### B.4 Fixed-Source Token Flow and Computation

**Holding the injection fixed.** Set $\sigma_{\mathcal{B}}(u) = \sigma_k(u)/B$ on each trajectory's prompt nodes and zero elsewhere. For each target $b$, require $\mathrm{supp}(\sigma_{\mathcal{B}}) \subseteq V_b$. Forward propagation through the conditioned kernel accumulates the visit mass

$$f_{\mathcal{B}}(u \mid b) = \sigma_{\mathcal{B}}(u) + \sum_{v \in V_b} f_{\mathcal{B}}(v \mid b) P_{\mathcal{B}}^{(b)}(v, u), \quad u \in V_b. \tag{24}$$

Nodes outside $V_b$ have zero conditioned visit mass under this source. We use this same unit-mass source for both conditioned flows. In particular, neither the total mass assigned to a prompt copy nor its allocation among prompt tokens is reweighted by outcome. For a token $u$, the path interpretation gives

$$f_{\mathcal{B}}(u \mid b) = \sum_s \sigma_{\mathcal{B}}(s) \Pr_{P_{\mathcal{B}}}(u \text{ is visited} \mid X_0 = s,\ X_{T_Z} \in Z_b). \tag{25}$$

Thus conditioning is normalized separately at each source. By contrast, if $f_{\mathcal{B}}$ denotes the unconditioned base flow, $f_{\mathcal{B}}(u)h_b(u)$ is the joint mass of visiting $u$ and ending in $Z_b$ under the original source mixture. It is generally different from Eq. equation 25, as is the globally conditioned flow obtained by dividing that joint mass by $\sum_s \sigma_{\mathcal{B}}(s)h_b(s)$. In particular, $h_0 + h_1 = 1$ does not imply that the two fixed-source conditioned flows sum to the base flow.

**Source reachability.** A sufficient condition for both targets to be reachable in a mixed group is $0 < \beta < 1$, together with a native path of positive probability from each injected prompt node to at least one response token. To see this, take such a token $u$ in trajectory $k$. Native transitions retain positive weight, so its own boundary remains reachable. Every peer row of $C_{k\ell}(u, \cdot)$ sums to one, allowing exchange to at least one peer token followed by a native step. Retaining native routing thereafter gives a positive-probability path to the peer's boundary. Since the group contains both labels, both outcomes are reachable from the source. This is a support condition; the resulting probabilities can still be small. Sources that bypass all response tokens, and endpoint choices of $\beta$, require their own reachability check. Conditioning a source on an unreachable event is undefined; dropping that source or changing its weight would change the specified process.

**Token and edge flow.** Forward propagation uniquely solves Eq. equation 24. Because a transient node cannot be revisited, its flow is both the expected number of visits before absorption and its visit probability:

$$f_{\mathcal{B}}(u \mid b) = \mathbb{E}_{\sigma_{\mathcal{B}}}^{(b)}\left[\sum_{t=0}^{T_Z - 1} \mathbf{1}\{X_t = u\}\right] = \Pr_{\sigma_{\mathcal{B}}}^{(b)}(\exists t < T_Z : X_t = u). \tag{26}$$

Visits are defined at the states of the composed routing kernel. A token whose native row is borrowed inside a transition is not counted as an additional intermediate visit. Equivalently, each borrowed transition can be expanded as $u \to \widetilde{v} \to w$, where $\widetilde{v}$ is a distinct donor state that must take a native step according to $P_\ell(v, \cdot)$. Marginalizing these donor states recovers Eq. (4); conditioning the expanded process on the same outcome gives the same flow at the original token states. For $u \in V_b$ and $v \in V_b \cup Z_b$, define $F^{(b)}(u, v) = f_{\mathcal{B}}(u \mid b)P_{\mathcal{B}}^{(b)}(u, v)$. Then

$$\begin{aligned} \sum_{v \in V_b \cup Z_b} F^{(b)}(u, v) &= f_{\mathcal{B}}(u \mid b), \\ f_{\mathcal{B}}(u \mid b) &= \sigma_{\mathcal{B}}(u) + \sum_{v \in V_b} F^{(b)}(v, u). \end{aligned} \tag{27}$$

Summing over transient nodes cancels internal flow and gives

$$\sum_{u \in V_b} \sum_{z \in Z_b} F^{(b)}(u, z) = \sum_{u \in V_b} \sigma_{\mathcal{B}}(u) = 1. \tag{28}$$

This conserves terminal absorption mass. The sum of token visit masses can exceed one because a path can pass through multiple tokens.

**Backward and forward passes.** First compute trajectory-local reference flows and construct pairwise interval overlaps using two-pointer sweeps. Order prompt nodes before responses and merge the response lists by their interval left endpoints. Equal left endpoints can be ordered arbitrarily, as there are no positive transitions between them. Set terminal values to $o_k$ and compute $h_1$ in reverse order; then obtain $h_0 = 1 - h_1$. Initialize two flow channels with $\sigma_{\mathcal{B}}$ and propagate each channel through its corresponding $P_{\mathcal{B}}^{(b)}$ in forward order on $V_b$. Read out $g_{\tau,i}$ from the channel $b = o_\tau$ only after propagation; changing channels at intermediate nodes would define a different process. These are finite passes on a DAG, with no matrix inversion or iterative convergence tolerance. The base mixture can be evaluated from native rows and sparse overlap weights. Numerical interval endpoints should be accumulated consistently, with the final endpoint fixed at one to avoid roundoff-induced coverage gaps.

### B.5 Limiting Cases and Scope of Interpretation

**All outcomes agree.** If $o_k = c$ for every trajectory, then $h_c \equiv 1$ and $h_{1-c} \equiv 0$. Conditioning on the observed class leaves the base process unchanged:

$$P_{\mathcal{B}}^{(c)} = P_{\mathcal{B}}, \qquad f_{\mathcal{B}}(u \mid c) = f_{\mathcal{B}}(u). \tag{29}$$

The observed-outcome flow is therefore well defined; the unreachable opposite-outcome channel is not needed for credit assignment. For binary outcome-only GRPO, these homogeneous groups have zero advantage, so any finite token credits yield the same zero contribution to the advantage-weighted policy term (Eq. 9). Bypassing credit computation with unit weights as specified in Section 4.3 preserves this term exactly. If reward shaping produces nonzero advantages in a homogeneous-outcome group, its observed-outcome flow above remains usable (Appendix B.7).

**No cross-trajectory exchange.** For $\beta = 0$ or $B = 1$, each trajectory reaches only its own boundary. On trajectory $k$, $h_{o_k} \equiv 1$, so conditioning native routing on its own observed outcome leaves $P_k$ unchanged. With its unit-mass native source, the resulting flow is $f_k^0$. For a mixed group at $\beta = 0$, however, a fixed target $b$ is unreachable from prompt copies belonging to the other outcome class. Thus the two conditioned flows from the full batch source are not defined at this endpoint. A trajectory-local reference is computed directly from the native flows; it is not obtained by silently dropping unreachable sources from the fixed-source construction.

**Interpretation.** Interval overlap supplies a correspondence in cumulative reference-flow mass. The potential $h_b$ measures outcome-hitting probabilities under the coupled graph kernel, and the conditioned visits quantify token participation in its outcome-directed paths. Flow supplies the importance ranking for weighting the observed trajectory-level supervision, while the trajectory advantage supplies its sign. The two outcome channels share a definition and a probability interpretation; their numerical values need not agree. Selecting a channel according to each trajectory's label produces the scores $g_{\tau,i}$ used in Section 4.3, not a third globally conservative flow. Appendix B.6 characterizes how this importance is used to shape token-level credit in policy optimization.

### B.6 Optimization Interpretation of Token Credit

The flow score $g_{\tau,i}$ measures outcome-conditioned token importance in the auxiliary graph. Section 4.3 uses this importance to construct credit weights $c_{\tau,i}$ and allocate trajectory-level supervision through $\widetilde{A}_{\tau,i}$. We formalize the resulting weighted optimization surrogate for the positive-credit rule and its normalized variant.

**Frozen rollout quantities.** Let $\bar{\theta}$ denote the rollout policy parameters and $\mathsf{D}_{\bar{\theta}}$ the distribution of sampled prompt groups under the training sampling protocol. For each group $\mathcal{B}$, compute $c_{\tau,i} = c_{\tau,i}^{\bar{\theta}}(\mathcal{B})$ from its attention, completed trajectories, and observed outcomes. These weights and the GRPO advantages are held fixed throughout policy optimization on that rollout batch; no gradient is propagated through their computation. For the sampled token $x_{\tau,i}$ with preceding context $s_{\tau,i}$,

define the usual likelihood ratio and clipped policy term

$$r_{\tau,i}(\theta) = \frac{\pi_\theta(x_{\tau,i} \mid s_{\tau,i})}{\pi_{\bar{\theta}}(x_{\tau,i} \mid s_{\tau,i})}, \qquad \ell(r,a) = \min\left\{ra,\ \mathrm{clip}(r, 1-\epsilon, 1+\epsilon)a\right\}, \qquad \epsilon > 0. \tag{30}$$

Let $w_{\tau,i} \geq 0$ encode the original loss aggregation, fixed with the sampled data. The weighted surrogate to maximize is

$$L_c(\theta;\bar{\theta}) = \mathbb{E}_{\mathcal{B}\sim \mathsf{D}_{\bar{\theta}}}\left[\sum_{\tau\in\mathcal{B}}\sum_{i=1}^{T_\tau} w_{\tau,i}c_{\tau,i}\,\ell\left(r_{\tau,i}(\theta), \widehat{A}_{\tau,i}\right)\right]. \tag{31}$$

The reference $L_0$ sets every $c_{\tau,i} = 1$, with the same rollout distribution, advantages, clipping, and aggregation.

**Proposition: equivalence to a weighted surrogate.** For fixed positive credits, inserting $c_{\tau,i}\widehat{A}_{\tau,i}$ into the clipped policy term gives exactly the integrand in Eq. (31). Indeed, $\ell(r, ca) = c\ell(r,a)$ for $c > 0$, since a positive common factor can be taken outside the minimum. Differentiating with the rollout quantities fixed therefore gives the gradient of $L_c$. At a fixed evaluation parameter $\theta$, where the terms are differentiable and differentiation can be exchanged with expectation, a sampled-group gradient is unbiased for $\nabla_\theta L_c$. No independence between a credit and its group's tokens, outcomes, or gradients is required for this statement. On a fixed collected batch, the same identity applies to its empirical surrogate.

**Proposition: gradient shift relative to GRPO.** Write $u_{\tau,i}(\theta) = \nabla_\theta \ell(r_{\tau,i}(\theta), \widehat{A}_{\tau,i})$. Under the same conditions,

$$\begin{aligned}\Delta(\theta;\bar{\theta}) &:= \nabla_\theta L_c - \nabla_\theta L_0 \\ &= \mathbb{E}_{\mathcal{B}}\left[\sum_{\tau,i} w_{\tau,i}(c_{\tau,i}-1)u_{\tau,i}(\theta)\right].\end{aligned} \tag{32}$$

This follows by subtracting the two surrogate gradients. For the main rule in Eq. (7), $c_{\tau,i} - 1$ equals $\gamma - 1$ on $\mathcal{H}_\tau$ and zero elsewhere; groups with zero advantages contribute zero under the bypass convention. Thus unit credits recover $L_0$, whereas selective amplification generally changes its expected gradient. At $\theta = \bar{\theta}$, clipping is inactive and

$$u_{\tau,i}(\bar{\theta}) = \widehat{A}_{\tau,i}\ \nabla_\theta \log \pi_\theta(x_{\tau,i} \mid s_{\tau,i})|_{\theta=\bar{\theta}}. \tag{33}$$

Stop-gradient removes derivatives through $c$; it does not remove the statistical dependence between $c-1$ and these gradient contributions. Consequently, $\Delta$ is the additional bias if the weighted update is regarded as an estimator of the unweighted surrogate gradient. This comparison does not assume that GRPO itself is an unbiased estimator of the gradient of expected task reward.

**Effect of mean-credit normalization.** Set $\bar{c}_\tau = T_\tau^{-1}\sum_i c_{\tau,i}$ and $c_{\tau,i}^{\mathrm{norm}} = c_{\tau,i}/\bar{c}_\tau$. Although $\sum_i (c_{\tau,i}^{\mathrm{norm}} - 1) = 0$, the shift in Eq. (32) need not vanish: the token gradient vectors differ, so zero-sum coefficient changes need not cancel after weighting those vectors. Normalization fixes the mean credit while preserving relative emphasis across tokens. Appendix D.3 evaluates learning with this allocation at matched mean credit.

**Scope of the optimization interpretation.** The update optimizes the weighted surrogate for a fixed reference policy and fixed credits. Refreshing rollouts and credits yields a sequence of such surrogates; the analysis does not identify a single fixed expected-reward objective for the full procedure or guarantee reward improvement. Likewise, the graph interpretation of $g_{\tau,i}$ does not establish that $c_{\tau,i}\widehat{A}_{\tau,i}$ estimates a local action advantage or a causal reward contribution. Positive credits preserve advantage signs, while their unequal weighting can change the direction of the aggregate parameter gradient.

### B.7 Outcome Attribution with Scalar Training Rewards

In tasks with a binary success criterion, the same outcome-conditioned flow supports scalar training rewards. The task outcome selects the attribution channel, and the full training reward determines the policy advantage. Our agentic experiments on HotpotQA, WebShop, and ALFWorld use this setting (Table 4); Appendix C.1 gives their reward definitions and evaluation metrics.

For a trajectory with observed task outcome $o_\tau$, token importance remains $g_{\tau,i} = f_{\mathcal{B}}(u_{\tau,i} \mid o_\tau)$, with credits defined by Eq. (7). Let $\widehat{A}^{\mathrm{sh}}_{\tau,i}$ be the advantage computed by the underlying RL algorithm from the full scalar reward, including any auxiliary or intermediate components. Apply

$$\widetilde{A}^{\mathrm{sh}}_{\tau,i} = c_{\tau,i}\widehat{A}^{\mathrm{sh}}_{\tau,i}. \tag{34}$$

The advantage may vary across tokens, and its sign need not agree with the binary outcome. Since $c_{\tau,i} > 0$, the main credit rule preserves each advantage's sign. Flow measures participation in paths toward the observed task outcome, while the full shaped reward supplies the supervision to be emphasized. Homogeneous-outcome groups may have nonzero advantages. For such groups, the observed-outcome channel remains well defined and equals the base coupled flow by Eq. (29); only that channel is needed. The zero-advantage shortcut applies only when the actual advantages vanish.

### B.8 Exploratory Extension: Selective Downscaling and Sign Reversal

The main rule amplifies high-flow positions and assigns unit weight elsewhere. We explore a further credit rule that retains high-flow amplification and uses relative outcome affinity to attenuate or reverse the original advantages at a limited number of positions. Appendix D.7 evaluates this alternative.

For trajectory $\tau$, let $\mathcal{Q}_\tau$ be its prompt nodes and $\sigma_\tau$ its unit-mass native source. Using the base-kernel potentials, define

$$\begin{aligned} \bar{h}_\tau &= \sum_{s\in\mathcal{Q}_\tau} \sigma_\tau(s)h_1(s), \\ m_{\tau,i} &= (2o_\tau - 1)\big(h_1(u_{\tau,i}) - \bar{h}_\tau\big). \end{aligned} \tag{35}$$

Equivalently, $m_{\tau,i}$ is the difference between $h_{o_\tau}(u_{\tau,i})$ and its average over this trajectory's prompt source. A negative margin indicates below-source affinity for the observed outcome under the base graph.

Among positions with $m_{\tau,i} < 0$, select up to $\lfloor \rho_{\mathrm{down}} T_\tau \rfloor$ with the smallest margins into $\mathcal{D}_\tau$, where $0 \le \rho_{\mathrm{down}} \le 1$, resolving ties by token order. This set is drawn from the whole response and may be empty. An exploratory credit rule is

$$c^{\mathrm{down}}_{\tau,i} = \begin{cases} \gamma_{\mathrm{down}}, & i \in \mathcal{D}_\tau, \\ \gamma, & i \in \mathcal{H}_\tau \setminus \mathcal{D}_\tau, \\ 1, & \text{otherwise}, \end{cases} \qquad \gamma_{\mathrm{down}} \in \mathbb{R}. \tag{36}$$

Here $\mathcal{H}_\tau$ is still selected by the outcome-conditioned score of the main method; the assignment on $\mathcal{D}_\tau$ takes precedence at overlapping positions. The credits directly replace $c_{\tau,i}$ in Eq. (8). On selected positions, $0 < \gamma_{\mathrm{down}} < 1$ attenuates the original advantage, $\gamma_{\mathrm{down}} = 0$ zeros it, and $\gamma_{\mathrm{down}} < 0$ reverses its sign. The empirical sweep also includes $\gamma_{\mathrm{down}} \ge 1$ as controls. In particular, $\gamma_{\mathrm{down}} = 1$ removes amplification at positions in $\mathcal{D}_\tau \cap \mathcal{H}_\tau$, so it need not recover the main rule; $\rho_{\mathrm{down}} = 0$ does. Groups with zero advantages retain the unit-credit convention. This extension adds a fraction cap and a multiplier; all main results for FlowWeaver use Eq. (7) without this adjustment.

## C Implementation Details

### C.1 Reinforcement Learning Implementation

**Training framework.** We implement RL training with ROLL [39] (Reinforcement Learning Optimization for Large-Scale Learning), an open-source library for distributed LLM reinforcement learning. Its multi-role architecture uses Ray to coordinate rollout generation, reward evaluation, and policy optimization, with support for both reasoning and multi-turn agentic training. We integrate our flow-based credit assignment between rollout collection and the GRPO policy update.

**Optimization and sampling.** The optimizer uses a global batch of 512, a micro-batch of 32, and 16 gradient-accumulation steps, with learning rate $1 \times 10^{-6}$. The loss keeps the clipped policy term and includes neither a KL penalty nor an entropy bonus. Trajectories are sampled with temperature 0.99, top-$p = 1$, and top-$k = 100$. All GRPO-based methods share the same dynamic-sampling protocol: we discard groups with zero within-group reward variance and continue sampling until the target update batch is filled. Under binary outcome-only rewards, all retained groups are mixed-outcome, so the mixed-group fraction among raw rollouts does not reduce the number of groups with nonzero advantages per update. Training budgets refer to optimization steps and batch sizes after this filtering. Replenishment cost belongs to this shared GRPO sampling protocol; attention extraction and flow computation incur the separate attribution overhead reported in Appendix D.8.

**Training schedules.** The 4B mathematical reasoning model trains for 500 steps on 8 GPUs, and the 8B model for 600 steps on 16 GPUs. Agentic training runs for 1,000 steps on 16 GPUs.

**Hyperparameter selection.** We select the defaults $\beta = 0.5$, $\rho = 0.4$, and $\gamma = 1.5$ on a held-out validation subset of MATH [12] and keep them fixed across tasks. The Qwen3-4B-Base sweeps in Table 8 illustrate the effects of varying these parameters.

**Agentic rewards and evaluation.** Agentic training uses scalar rewards to estimate advantages and binary task outcomes to condition flow. The binary labels are strict answer EM for HotpotQA, full task success for WebShop, and official success for ALFWorld.

For HotpotQA, an answer in the required format receives $0.05 + 0.70\,\mathrm{EM} + 0.25\,\mathrm{F1}$, where F1 is token F1. Invalid format receives $-0.1$, and search steps receive zero reward. Evaluation uses strict EM as the main metric and reports token F1 separately.

For WebShop, a valid purchase receives $0.05 + 0.95s$, where $s \in [0, 1]$ is the environment's raw score for matching product type, attributes, options, and price. Invalid format, invalid actions, and invalid operations incur rewards of $-0.1$, $-0.05$, and $-0.02$, respectively. Evaluation disables the completion bonus and reports the raw score $s$, with success defined by $s = 1$.

For ALFWorld, training uses potential shaping derived from the environment's hidden walkthrough with progress weight 0.2, together with penalties for invalid format, invalid actions, idling, and repeated states. Without penalties, a successful trajectory has total return 1, while an unsuccessful trajectory can receive up to approximately 0.2 in progress reward. Evaluation sets the progress weight to zero and uses official binary success as the main metric.

**Attention extraction after sampling.** After sampling a group of trajectories, we replay the completed sequences with their prompts in one additional forward pass through the same inference engine used for sampling. The sampled tokens remain fixed during this pass. Following FlowTracer [6], we retain the mean attention over the middle third of the model's layers, from $L/3$ to $2L/3$ for a model with $L$ layers, averaging over the selected layers and attention heads. The retained mean attention maps supply the influence weights $W$ in Section 3.1. Keeping these aggregates reduces the attention data retained for attribution. Appendix D.6 compares layer choices within the full method.

**From local flow to the joint DAG.** For each trajectory, we first compute the backward potential and native kernel $P_k$, then propagate the prompt source forward to obtain the reference flow $f_k^0$. We normalize the response-token flows and accumulate them in token order to define flow-progress intervals. Their overlaps match positions across trajectories sampled for the same prompt. We then combine the native kernels and these correspondences into the joint cross-trajectory DAG. On this graph, a reverse topological pass computes the outcome potential $h_1$, with $h_0 = 1 - h_1$; a forward pass propagates both outcome-conditioned flows in two channels. Each response reads its token scores from the channel corresponding to its observed outcome. Appendix B.4 gives the recurrence and source convention.

**Batched computation and policy integration.** We use batched tensor operations for attention aggregation and flow computation, batching independent trajectories and prompt groups while preserving the dependency order within each graph. The two outcome channels are propagated together, and the native kernels and overlap weights are reused throughout the coupled computation.

The DAG structure permits finite backward and forward passes without matrix inversion or iterative convergence checks. These graph operations use the extracted attention maps and require no further model forward passes. The resulting token weights multiply the original GRPO advantages as in Eq. (8) before the policy update. The weights are detached and cached for policy optimization on that rollout batch, with no gradient through attention extraction, graph computation, or token selection; new rollouts receive newly computed weights. This is the frozen-credit convention in Appendix B.6. Homogeneous-outcome groups under binary outcome-only GRPO follow the zero-variance convention in Section 4.3. Appendix D.8 reports the measured runtime.

**Memory management.** We reconstruct attention in blocks from the selected layers' query/key tensors and reuse GPU workspace, exporting each completed trajectory graph to CPU for groupwise solving. Before replaying a prompt group, we reserve memory for all its graphs; the budget covers groups undergoing replay, waiting, or being solved. Once a group has been solved, we retain only its token weights and release its graphs and reservation. For group size $G$, sequence length $T$ including the prompt, and at most $C$ groups in flight, resident dense graph storage is bounded by $O(CGT^2)$. This bound depends on the configured group limit; storage remains quadratic in sequence length within each group.

**8K execution measurements.** A separate forward-DAG execution benchmark uses Qwen3-4B-Base on one H200, four $G = 8$ groups of frozen responses repeated or truncated to total length 8K, Q/K from layers 10/15/20/25, and $\beta = 1$. Limiting residency to $C = 2$ reduces peak sampled CPU process proportional set size (PSS) from 9.848 to 5.722 GiB (41.9%). Peak GPU allocated memory is 8.817 and 8.848 GiB for the accumulated and bounded schedules, respectively, including the resident model. These maxima are taken over two paired runs after warmup, covering decoder replay, graph export, and CPU solving; generation and policy optimization are excluded. CPU PSS is sampled approximately once per second and may miss shorter peaks. Both schedules produce identical token weights.

For fixed $C$, $G$, and storage precision, increasing sequence length from 8K to 16K or 32K increases the dense graph payload by factors of four or sixteen, respectively. These factors describe graph storage; complete GPU and CPU memory peaks also depend on the model, replay microbatch, and execution workspace.

### C.2 Baseline Methods

Reasoning and agentic tasks use different baseline sets, as specified in their result tables. The DeepSeek-R1-Distill-Llama-8B comparison uses GRPO, Random, Entropy, Attention, and Flow.

Within each model–task setting, all methods share the policy initialization, training data, context limit, rollout sampling settings, effective batch size, and optimization schedule specified in Appendix C.1. Baseline-specific hyperparameters follow the choices reported in the respective papers.

**Token-level weighting.** GRPO [29] assigns the outcome advantage uniformly to every token. Token-level baselines reweight positions with signals from the policy itself: Random applies the same budget to randomly selected positions, Entropy [38] uses predictive entropy, Gradient [10] uses gradient-based token-importance scores, Attention [16] uses attention structure, and Flow [6] uses trajectory-local information flow.

**Advantage estimation and reward shaping.** AsyPPO [20] uses asymmetric mini-critics, TEMPO [32] adds prefix-tree value corrections, GTPO [31] reshapes token rewards by entropy, and GiGPO [8] adds step-level advantages at repeated agent states.

**External process supervision.** CAPO [41] and ThinkPRM [15] score reasoning steps, and Agent-PRM [4] scores agent actions using an external process judge.

### C.3 Protocols for Exchange Correspondence and Token-Masking Interventions

**Success probability, outcome certainty, and alignment mismatch.** The analyses in Section 4.2 use the same mixed-outcome groups sampled from a frozen policy. We compare random correspondence, normalized token progress, normalized step progress, and cumulative flow progress. Normalized token progress assigns equal mass to each response token before constructing interval overlaps. Normalized step progress assigns equal mass to each reasoning step, distributed uniformly over its tokens; flow progress uses the reference masses in Eq. equation 10. For a prefix ending at position $u$ in trajectory $k$, let $\widehat{p}_k(u)$ be the success rate of fresh continuations. We quantify outcome certainty by the probability of the more likely correctness outcome:

$$\widehat{c}_k(u) = \max\{\widehat{p}_k(u),\, 1 - \widehat{p}_k(u)\}. \tag{37}$$

The preferred outcome is determined by the fresh continuations at that prefix, independently of the original trajectory's outcome label $o_k$. Certainty is invariant to exchanging correct and incorrect labels: prefixes with success probabilities 0.80 and 0.21 have certainties 0.80 and 0.79, respectively, and hence a certainty gap of 0.01. Their success-probability gap is 0.59, retaining the distinction between prefixes favoring different outcomes. We report both gaps. Source prefixes are chosen at fixed interior token fractions, independently of the alignment method. For each source $u$ and peer trajectory $\ell$ from the same prompt, all four methods match this same source to the peer. Writing $C^A_{k\ell}$ for the correspondence under alignment $A$, we use the correspondence-weighted absolute success-probability gap

$$\widehat{d}^{\,p}_A(k, \ell, u) = \sum_v C^A_{k\ell}(u, v)\big|\widehat{p}_k(u) - \widehat{p}_\ell(v)\big|, \tag{38}$$

and the corresponding certainty gap

$$\widehat{d}^{\,c}_A(k, \ell, u) = \sum_v C^A_{k\ell}(u, v)\big|\widehat{c}_k(u) - \widehat{c}_\ell(v)\big|. \tag{39}$$

All methods use the same eligible source–peer comparisons, excluding zero-flow source positions and comparisons that include explicit final-answer positions under any alignment. Continuation counts, decoding settings, and budgets of newly generated tokens are shared; estimates for shared prefixes are reused. Fresh outcomes are reserved for evaluation. Each absolute gap is averaged first within each prompt and then across prompts. Table 1 reports these means multiplied by 100, in percentage points, abbreviating the matched-prefix estimates as $p, p'$ and $c, c'$.

**Token selection and information blocking.** We first sample complete trajectory groups and compute trajectory-local flow $f^0_\tau$ and observed-outcome flow $g_{\tau,i}$ from these original samples. Local flow uses the focal trajectory, while observed-outcome flow uses routing and observed outcomes from its group. We then randomly choose a breakpoint in each trajectory. Eligible intervention positions precede this breakpoint and exclude an adjacent gap, so the masked tokens are separated from the point where generation resumes. The original scores determine the masks; regenerated continuations are used only for evaluation. This is a retrospective attribution experiment using complete sampled trajectories.

Table 2(a) compares four selection rules on the eligible positions: random, highest trajectory-local flow, lowest observed-outcome flow, and highest observed-outcome flow. Each rule uses the same 20% token budget within a response. We block forward information propagation from the selected tokens and sample a fresh continuation from the breakpoint in the trajectory's own context. An unmasked control samples a continuation from the same breakpoint without information blocking. Both masked and unmasked continuations use sampling, and the breakpoint and eligible support are shared across selectors.

**Interventions on entering and leaving positions.** Let $H^0_\tau$ and $H^g_\tau$ denote the top-20% sets under local and observed-outcome flow, respectively, using the same eligible positions and token count. Define

$$E_\tau = H^g_\tau \setminus H^0_\tau, \qquad L_\tau = H^0_\tau \setminus H^g_\tau. \tag{40}$$

Entering positions $E_\tau$ are selected only by observed-outcome flow; leaving positions $L_\tau$ are selected only by local flow. Since $|H^0_\tau| = |H^g_\tau|$, their counts satisfy $|E_\tau| = |L_\tau|$ within each response.

Table 2(b) blocks each of these sets separately and compares them with random masks of the same per-response size. Each set accounts for 7.23% of tokens overall; the actual mask count follows the set size in each response. This comparison tests the positions whose selection changes when group information is incorporated.

**Answer changes and correctness flips.** Let $a_\tau^{\text{orig}}$ and $a_\tau^M$ denote the extracted answers from the original sampled trajectory and the continuation under mask rule $M$, and let $o_\tau$ and $o_\tau^M$ be their binary correctness labels. The two indicators are

$$A_\tau^M = \mathbf{1}\{a_\tau^M \neq a_\tau^{\text{orig}}\}, \qquad F_\tau^M = \mathbf{1}\{o_\tau^M \neq o_\tau\}. \tag{41}$$

Table 2 reports their aggregate rates as percentages. An answer can change while remaining incorrect. Correctness flips count both correct-to-incorrect and incorrect-to-correct transitions, and therefore measure outcome sensitivity rather than a signed change in accuracy. The same indicators are evaluated for unmasked resampling, yielding an answer-change rate of 20.8% and a correctness-flip rate of 10.8% in each intervention panel. These controls quantify changes introduced by regeneration alone, with every condition compared to the original sampled answer.

For each metric, the table also reports $\Delta$, the masked rate minus the corresponding unmasked rate in percentage points. This contrast measures the aggregate increase in changes relative to unmasked resampling. It does not count how many individual responses would change under masking but remain unchanged under unmasked resampling; that joint quantity requires paired outcomes. The contrasts between selection rules assess which token sets produce larger downstream changes under this intervention and continuation protocol. The downstream training experiments separately evaluate the utility of the credit rule.

# D Additional Experiments

We examine the design and behavior of shared outcome-conditioned credit. The graph and weighting comparisons illustrate the effects of parameter and design choices, and peer-replacement experiments assess sensitivity to group sampling and outcome composition. Measurements of lexical diversity, variation across seeds, and runtime characterize its use in training. An exploratory credit rule further examines selective adjustment based on relative outcome affinity.

## D.1 Design of Outcome-Conditioned Attribution

**Outcome-specific importance on shared paths.** Coupling makes the group's alternative answer boundaries reachable in a common graph, providing the path distribution on which outcome conditioning operates. Conditioning this distribution on a response's observed outcome defines its token-importance scores. Both successful and unsuccessful trajectories can contribute intermediate routes toward the target outcome. On an isolated native graph, $h_{o_k} = 1$, so conditioning on its own outcome leaves routing unchanged (Appendix B.5). The ablations below examine how the shared paths and outcome targeting shape token selection within this attribution framework.

**Setup.** We evaluate the attribution design on Qwen3-4B-Base at context length 1K. Table 11(a) varies exchange strength with cumulative-flow correspondence, comparing the full method with unconditioned propagation under the same shared kernel $P_{\mathcal{B}}$. This control ranks tokens by shared flow without targeting an outcome; local Flow provides the trajectory-local reference. All variants amplify the top 40% of response tokens by $\gamma = 1.5$, keeping the total credit fixed for a given response while changing the selected positions.

**Outcome-conditioned attribution across exchange strengths.** In Table 11(a), the full method reaches its highest average score of 40.9% at $\beta = 0.5$, compared with 39.4% for local Flow and 40.1% for the unconditioned shared-flow control. It outperforms the unconditioned control at every tested exchange strength. These comparisons support using the shared graph to measure token participation in paths toward the observed outcome. The exchange sweep also shows that the native–peer routing balance matters for this attribution: increasing exchange beyond $\beta = 0.5$ lowers the full method's score.

Table 11: Design of outcome-conditioned attribution on Qwen3-4B-Base at context length 1K. (a) Exchange strength in the full method and an unconditioned shared-flow control, both using cumulative-flow correspondence. (b) Correspondence rules within the full method at $\beta = 0.5$. Avg. is the unweighted mean over the five mathematical reasoning benchmarks (%). All variants use the same top-$\rho$ selection fraction ($\rho = 40\%$) and amplification ($\gamma = 1.5$).

Local flow (no exchange, $\beta = 0$): Avg. = 39.4

**(a) Exchange strength**

| $\beta$ | Unconditioned shared flow | Outcome-conditioned flow (FlowWeaver) |
|---|---|---|
| 0.1 | 38.8 | 39.6 |
| 0.3 | 38.9 | 40.1 |
| **0.5**† | **40.1** | **40.9** |
| 0.7 | 39.0 | 39.4 |
| 0.9 | 38.4 | 39.7 |

**(b) Correspondence**

| Rule | Avg. ↑ | $\Delta_{\text{local}}$ |
|---|---|---|
| Token progress | 38.9 | $-0.5$ |
| Step progress | 38.4 | $-1.0$ |
| Embedding similarity | 38.1 | $-1.3$ |
| Flow progress | **40.9** | $+1.5$ |

Local Flow is from Table 3; full-method scores in (a) are from Table 8(a). Panel (a) reports Avg. (%); $\Delta_{\text{local}}$ in (b) is Avg. minus local Flow (pp). Blue shading and † mark defaults; bold marks the best Avg. within each variant in (a) and across rules in (b). Embedding similarity uses cosine DTW on final-layer states with a shared reference.

**Correspondence baselines.** Table 11(b) compares four correspondence rules used throughout RL training within the full outcome-conditioned method, with $\beta = 0.5$, $\rho = 0.4$, and $\gamma = 1.5$ held fixed. All four rules construct ordered token intervals and use their normalized overlaps as correspondence weights. Normalized token progress assigns equal mass to tokens, normalized step progress assigns equal mass to reasoning steps and distributes it uniformly over their tokens, and cumulative flow progress uses native reference-flow mass. Contextual token matching in BERTScore [47] and SimAlign [13] motivates the embedding-similarity baseline. It uses cosine similarity between the policy's final-layer token states, drawing on Zhao et al. [48], who use cosine similarity between mean-pooled final-layer states to assess knowledge relevance.

**Monotone correspondence from embedding similarity.** We take the first sampled response in each group as a shared reference, independently of its outcome. Each other response is aligned to it by dynamic time warping (DTW), minimizing the sum of $1 - \cos(e_{k,i}, e_{\text{ref},j})$ along a path that covers both sequences using moves $(1, 0)$, $(0, 1)$, or $(1, 1)$. Here $e$ denotes a final-layer token state. If the reference has $m$ tokens, each contributes mass $1/m$, divided equally among the response tokens aligned to it. Writing $\mathcal{A}_k$ for the alignment path and $d_{k,j} = |\{i : (i, j) \in \mathcal{A}_k\}|$, the assigned token mass is

$$\mu^{\text{sim}}_{k,i} = \sum_{j:(i,j)\in\mathcal{A}_k} \frac{1}{m\, d_{k,j}}. \tag{42}$$

The reference retains equal token masses. For every response, we accumulate these positive masses in token order to obtain intervals on $[0, 1]$, then apply the interval-overlap construction in Appendix B.1. The resulting correspondence is monotone. Because all responses share this interval coordinate, exchange followed by native propagation strictly advances it, preserving the joint DAG as in Appendix B.2.

**Flow correspondence performs best among the tested rules.** Flow progress reaches 40.9%, exceeding token progress (38.9%), step progress (38.4%), and embedding similarity (38.1%) by 2.0, 2.5, and 2.8 percentage points, respectively. It is the only tested correspondence that improves on local Flow (39.4%) at this exchange strength. This comparison evaluates how correspondence choices affect the complete outcome-conditioned attribution, supporting reference flow as its shared path coordinate.

### D.2 Sensitivity to Group Sampling and Outcome Composition

The group-size ablation in Table 7 shows that small groups already capture most of the cross-trajectory gain. Here we fix the group size at eight and examine how sampling different peers from the same policy affects token selection, both with and without changing the group's outcome composition.

**Setup and peer replacements.** We sample 32 responses per problem on MATH-500 using a fixed Qwen3-4B-Base policy. From a problem's sample pool, we select a reference group containing one incorrect and seven correct trajectories, denoted by the *incorrect:correct* ratio 1:7. We hold the sole incorrect trajectory fixed as the focal response and compute its observed-outcome flow, conditioned on failure as in Eq. (6). For each replacement count $k \in \{1, \dots, 6\}$, we replace $k$ of the seven correct peers with other trajectories from the same pool. *Opposite-outcome replacement* introduces $k$ incorrect trajectories, yielding a ratio of $(1+k)$:$(7-k)$. *Same-outcome replacement* introduces $k$ other correct trajectories, preserving 1:7. The focal response and the remaining peers stay fixed. Each perturbed group is compared directly with the original 1:7 group.

**Change in the selected tokens.** We recompute the focal response's flow in each group and select its top 40% of tokens using the rule in Section 4.3. Let $H_\tau^{\mathrm{ref}}$ and $H_\tau^{\mathrm{pert}}$ be the selected sets under the reference and perturbed groups. Since the sets have equal size, the numbers of entering and leaving tokens are equal. We report either count as a fraction of all $T_\tau$ response tokens,

$$d_\tau = \frac{|H_\tau^{\mathrm{ref}} \setminus H_\tau^{\mathrm{pert}}|}{T_\tau} = \frac{|H_\tau^{\mathrm{pert}} \setminus H_\tau^{\mathrm{ref}}|}{T_\tau}. \tag{43}$$

Table 12 reports the mean change as a percentage for each replacement type and count.

Table 12: Sensitivity of the focal trajectory's top-40% token selection to peer replacement on Qwen3-4B-Base and MATH-500. All groups contain eight trajectories drawn from a pool of 32 responses per problem. Entries report the mean percentage of all focal-response tokens entering (equivalently, leaving) the top-40% set relative to the original 1:7 group (Eq. (43)). Target ratios are incorrect:correct and apply to opposite-outcome replacement; same-outcome replacement always preserves 1:7.

| Target ratio from 1:7 | Replaced peers $k$ | Opposite-outcome change (%) | Same-outcome change (%) |
|---|---|---|---|
| 2:6 | 1 | 1.01 | 0.71 |
| 3:5 | 2 | 1.41 | 0.89 |
| 4:4 | 3 | 1.80 | 0.87 |
| 5:3 | 4 | 2.29 | 0.70 |
| 6:2 | 5 | 2.68 | 0.95 |
| 7:1 | 6 | 3.02 | 0.98 |

**Token selection is stable under peer resampling.** Same-outcome replacement displaces only 0.70–0.98% of all response tokens from the selected set across all six replacement counts. Opposite-outcome replacement produces a larger change at every matched count, increasing monotonically from 1.01% for one replacement to 3.02% for six. At group size eight, token selection is therefore stable under peer resampling, with larger shifts when replacements alter the outcome composition.

### D.3 Credit Normalization and Uniform Scaling

This ablation examines how mean credit scale affects learning with flow-guided token allocation. We compare four weighting configurations on Qwen3-4B-Base at context length 8K, following the mathematical reasoning setup in Table 3: uniform and flow-guided weights, each at unit and amplified mean credit. Let $c_{\tau,i}$ be the credit in Eq. (7) and define its mean over the $T_\tau$ valid response tokens as

$$\bar{c}_\tau = \frac{1}{T_\tau} \sum_{i=1}^{T_\tau} c_{\tau,i} = 1 + (\gamma - 1) \frac{\lceil \rho T_\tau \rceil}{T_\tau}. \tag{44}$$

At the default $\rho = 0.4$ and $\gamma = 1.5$, $\bar{c}_\tau \approx 1.2$; the exact value accounts for rounding the selected-token count. The four rules multiply the original GRPO advantages by 1 (GRPO), $\bar{c}_\tau$ (uniformly scaled GRPO), $c_{\tau,i}/\bar{c}_\tau$ (normalized FlowWeaver), and $c_{\tau,i}$ (FlowWeaver), respectively. Normalization preserves the relative token weights and sets the mean credit to one within each response. The uniform control matches the main method's mean credit within each response. Homogeneous groups with zero advantage retain the unit-credit convention.

All four conditions use the same number of optimization steps, response batch size after dynamic filtering, optimization and evaluation settings, and three-seed averaging protocol, with $\beta = 0.5$ for both FlowWeaver variants. Table 13 reports the five mathematical reasoning benchmarks and their unweighted average, allowing token allocation to be compared at each matched mean credit and scale sensitivity to be assessed within each method.

Table 13: Credit normalization and uniform scaling on Qwen3-4B-Base at context length 8K. Mean credit is computed per response; $\bar{c}_\tau$ is defined in Eq. (44). Scores are in %, and Avg. is the unweighted mean over the five mathematical reasoning benchmarks. All results are averaged over three training seeds. GRPO and FlowWeaver reuse the results from Table 3. Best scores in each column are **bold**.

| Method | Mean credit | AIME24 | AIME25 | AMC23 | MATH500 | Olymp. | Avg. ↑ |
|---|---|---|---|---|---|---|---|
| GRPO | 1 | 19.5 | 16.1 | 57.6 | 81.0 | 49.9 | 44.8 |
| GRPO (uniform upscale) | $\bar{c}_\tau \approx 1.2$ | 19.7 | 16.4 | 57.0 | 80.2 | 49.6 | 44.6 |
| FlowWeaver (normalized) | 1 | 24.4 | 23.0 | 64.2 | 84.4 | 53.5 | 49.9 |
| FlowWeaver | $\bar{c}_\tau \approx 1.2$ | **24.6** | **23.1** | **65.0** | **84.6** | **54.6** | **50.4** |

**Flow-guided allocation is effective at both credit scales.** Uniformly scaling GRPO yields an Avg. of 44.6, compared with 44.8 for the original GRPO. Normalized FlowWeaver reaches 49.9, improving over GRPO by 5.1 percentage points at mean credit one and falling 0.5 points below the main method (50.4). At the amplified mean credit, FlowWeaver exceeds uniformly scaled GRPO by 5.8 points (50.4 versus 44.6). Flow-guided allocation performs well at both tested scales, with the main weighting configuration yielding the highest score and normalization retaining most of its gain.

### D.4 Lexical Diversity of Generated Responses

**Setup.** We characterize the lexical diversity of responses generated after RL with different credit rules. We compare Qwen3-4B-Base with models trained from it using GRPO, Random, Entropy, Attention, Flow, and FlowWeaver. For each problem in MATH-500, we sample $n = 32$ responses from each model at context length 1K, with temperature 1.0, top-$p = 0.7$, and top-$k = 10$. We compute pairwise BLEU-4 between responses to the same problem, using up to 4-grams. We average the pairwise scores within each problem and then across the benchmark. Lower scores indicate less surface-form overlap and hence greater lexical diversity.

Table 14: Lexical diversity on MATH-500 using Qwen3-4B-Base at context length 1K, with $n = 32$ sampled responses per problem. Lower pairwise BLEU-4 indicates greater lexical diversity. $\Delta$: model $-$ GRPO in percentage points; negative values indicate greater diversity than GRPO.

| Model | Pairwise BLEU-4 (%) ↓ | Δ vs. GRPO (pp) |
|---|---|---|
| Base | 43.67 | — |
| GRPO | 69.56 | — |
| Random | 69.21 | –0.35 |
| Entropy | 70.82 | +1.26 |
| Attention | 68.23 | –1.33 |
| Flow | 69.62 | +0.06 |
| **FlowWeaver** | 69.57 | +0.01 |

**Lexical diversity remains comparable to GRPO.** Table 14 shows that FlowWeaver has a pairwise BLEU-4 of 69.57%, 0.01 percentage points above GRPO (69.56%). It also lies within the range of the other token-level methods (68.23–70.82%). All trained models exhibit greater lexical overlap than the base model (43.67%). In this setting, learning with shared outcome-conditioned credit yields lexical diversity comparable to GRPO.

### D.5 Main Results over Five Training Seeds

The main results in Section 5.2 are averaged over three training seeds. To further assess the effectiveness of FlowWeaver, we extend this evaluation to five training seeds for GRPO, trajectory-local Flow, and FlowWeaver. These comparisons cover every experimental setting reported in the Main Results subsection (Section 5.2).

Table 15 reports mean ± sample standard deviation for each reasoning benchmark and the primary agentic outcome metrics. For reasoning, each seed's Avg. is the unweighted benchmark mean, and its standard deviation is computed across the five seed-level averages. Distill-Llama-8B denotes DeepSeek-R1-Distill-Llama-8B; the agentic settings use the checkpoints and EM/success metrics specified in the main results.

Table 15: Results over five training seeds for GRPO, trajectory-local Flow, and FlowWeaver across all 11 main-result settings (mean ± sample standard deviation). Scores are in %; standard deviations are in percentage points. Avg. is the unweighted mean across reasoning benchmarks; agentic metrics are HotpotQA EM and task success.

| *Mathematical reasoning* | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Model / ctx.** | **Method** | **AIME24** | **AIME25** | **AMC23** | **MATH500** | **Olymp.** | **Avg.** |
| Qwen3-4B-Base 1K | GRPO | $8.4 \pm 0.2$ | $5.1 \pm 0.4$ | $55.3 \pm 0.4$ | $74.4 \pm 0.5$ | $42.9 \pm 0.3$ | $37.2 \pm 0.3$ |
| | Flow | $10.6 \pm 0.4$ | $6.8 \pm 0.3$ | $58.9 \pm 0.4$ | $75.9 \pm 0.4$ | $44.0 \pm 0.4$ | $39.2 \pm 0.3$ |
| | **FlowWeaver** | $11.7 \pm 0.5$ | $7.8 \pm 0.3$ | $61.7 \pm 0.5$ | $77.3 \pm 0.4$ | $45.6 \pm 0.4$ | $40.8 \pm 0.3$ |
| Qwen3-4B-Base 8K | GRPO | $19.6 \pm 0.4$ | $16.0 \pm 0.5$ | $57.5 \pm 0.7$ | $81.0 \pm 0.3$ | $49.9 \pm 0.4$ | $44.8 \pm 0.4$ |
| | Flow | $22.5 \pm 0.4$ | $21.7 \pm 0.4$ | $62.5 \pm 0.5$ | $83.1 \pm 0.3$ | $53.0 \pm 0.3$ | $48.6 \pm 0.3$ |
| | **FlowWeaver** | $24.5 \pm 0.4$ | $23.0 \pm 0.4$ | $65.0 \pm 0.6$ | $84.5 \pm 0.3$ | $54.5 \pm 0.4$ | $50.3 \pm 0.4$ |
| Qwen3-8B-Base 1K | GRPO | $9.2 \pm 0.4$ | $7.2 \pm 0.6$ | $58.9 \pm 0.7$ | $77.2 \pm 0.3$ | $44.2 \pm 0.5$ | $39.3 \pm 0.4$ |
| | Flow | $12.8 \pm 0.5$ | $11.4 \pm 0.6$ | $65.0 \pm 0.6$ | $79.2 \pm 0.7$ | $46.5 \pm 0.3$ | $43.0 \pm 0.4$ |
| | **FlowWeaver** | $13.8 \pm 0.5$ | $12.9 \pm 0.5$ | $67.8 \pm 0.8$ | $80.2 \pm 0.6$ | $47.4 \pm 0.5$ | $44.4 \pm 0.4$ |
| Qwen3-8B-Base 8K | GRPO | $25.0 \pm 0.4$ | $20.7 \pm 0.4$ | $66.5 \pm 0.5$ | $85.1 \pm 0.3$ | $54.4 \pm 0.3$ | $50.3 \pm 0.3$ |
| | Flow | $26.0 \pm 0.5$ | $22.2 \pm 0.4$ | $70.4 \pm 0.6$ | $86.6 \pm 0.4$ | $56.7 \pm 0.4$ | $52.4 \pm 0.3$ |
| | **FlowWeaver** | $27.5 \pm 0.6$ | $23.0 \pm 0.4$ | $73.6 \pm 0.5$ | $88.7 \pm 0.4$ | $58.1 \pm 0.4$ | $54.2 \pm 0.3$ |
| Distill-Llama-8B 1K | GRPO | $7.5 \pm 0.4$ | $7.5 \pm 0.3$ | $42.4 \pm 0.4$ | $62.7 \pm 0.4$ | $29.0 \pm 0.3$ | $29.8 \pm 0.3$ |
| | Flow | $8.9 \pm 0.3$ | $7.7 \pm 0.4$ | $45.5 \pm 0.5$ | $64.2 \pm 0.4$ | $31.8 \pm 0.4$ | $31.6 \pm 0.3$ |
| | **FlowWeaver** | $10.4 \pm 0.4$ | $9.1 \pm 0.4$ | $49.0 \pm 0.5$ | $67.9 \pm 0.4$ | $34.2 \pm 0.3$ | $34.1 \pm 0.3$ |

| *General reasoning: Qwen3-8B-Base* | | | | | |
|---|---|---|---|---|---|
| **Method** | **CrossThinkQA** | **MMLU-Pro** | **GPQA-Diamond** | **BBEH** | **Avg.** |
| GRPO | $48.1 \pm 0.4$ | $57.1 \pm 0.6$ | $42.0 \pm 0.7$ | $11.4 \pm 0.4$ | $39.7 \pm 0.6$ |
| Flow | $50.0 \pm 0.4$ | $61.4 \pm 0.7$ | $43.4 \pm 0.8$ | $11.5 \pm 0.4$ | $41.6 \pm 0.6$ |
| **FlowWeaver** | $52.4 \pm 0.4$ | $65.6 \pm 1.0$ | $45.5 \pm 0.7$ | $13.0 \pm 0.2$ | $44.1 \pm 0.6$ |

| *Agentic tasks* | *Qwen3-4B* | | | | *Distill-Llama-8B* |
|---|---|---|---|---|---|
| **Method** | **HotpotQA** | **WebShop** | **ALFWorld$_{\text{sub}}$** | **ALFWorld$_{\text{full}}$** | **HotpotQA** |
| GRPO | $52.0 \pm 0.6$ | $89.6 \pm 0.9$ | $60.0 \pm 1.7$ | $71.6 \pm 1.8$ | $54.1 \pm 0.8$ |
| Flow | $59.5 \pm 0.8$ | $92.4 \pm 0.9$ | $65.6 \pm 1.0$ | $76.1 \pm 1.4$ | $56.8 \pm 0.6$ |
| **FlowWeaver** | $61.0 \pm 0.7$ | $94.0 \pm 0.8$ | $68.6 \pm 1.0$ | $82.6 \pm 1.2$ | $59.4 \pm 0.6$ |

**Performance gains persist across five seeds.** FlowWeaver achieves the highest five-seed mean on every reported reasoning benchmark and agentic outcome metric. Across the five mathematical model–context settings, its average gains are 3.6–5.5 percentage points over GRPO and 1.4–2.5 over Flow. General reasoning improves by 4.4 points over GRPO and 2.5 over Flow. Across the five agentic settings, gains range from 4.4 to 11.0 points over GRPO and from 1.5 to 6.5 over Flow. The standard deviations of FlowWeaver are 0.3–0.6 points for reasoning averages and 0.6–1.2 points for agentic outcomes, comparable to those of local Flow. These consistent mean gains across all main-result settings provide further empirical support for the effectiveness of outcome-conditioned cross-trajectory credit.

### D.6 Attention Layer Selection

We compare four attention layer selection strategies for flow computation in FlowWeaver on Qwen3-4B-Base at context length 1K. Top and Bottom average attention over the final and initial thirds of transformer layers, respectively; Middle uses the middle third, and All uses every layer. Head averaging and the training setup are held fixed, with $\beta = 0.5$, $\rho = 0.4$, and $\gamma = 1.5$.

Table 16: Attention layer selection for FlowWeaver on Qwen3-4B-Base at context length 1K. Avg. is the unweighted mean over the five mathematical reasoning benchmarks (%).

| **Layers** | Top | Bottom | Middle† | All |
|---|---|---|---|---|
| Avg. (%) ↑ | 39.4 | 39.7 | 40.9 | 40.6 |

†: default; Middle reuses Table 3.

Table 16 shows that Middle achieves the highest average score (40.9%), followed by All (40.6%), Bottom (39.7%), and Top (39.4%). We therefore adopt Middle as the default layer selection for FlowWeaver.

### D.7 Exploratory Outcome-Affinity Credit Adjustment

**Setup.** We examine whether the additional outcome-affinity criterion in Appendix B.8 can provide incremental gains over the main positive-weighting rule. We evaluate the extension on Qwen3-4B-Base at context length 8K. Scores are averaged over three seeds; Table 17 reports their unweighted mean over the five mathematical reasoning benchmarks. We vary the fraction cap $\rho_{\text{down}}$ at fixed $\gamma_{\text{down}} = 0.5$, and the multiplier $\gamma_{\text{down}}$ at fixed $\rho_{\text{down}} = 5\%$. Both sweeps share the $(5\%, 0.5)$ configuration.

Table 17: Selective downscaling and sign reversal on Qwen3-4B-Base at context length 8K, averaged over three seeds. Avg. is the unweighted mean of the five mathematical reasoning benchmark scores (%); $\Delta$ is the difference from the main rule in percentage points. Gray columns mark the shared $(5\%, 0.5)$ configuration; the blue column is the main rule without downscaling. **Bold** marks the best score in each panel.

| **Reference Avg.** | GRPO: 44.8 | Main rule: 50.4 |
|---|---|---|

**(a) Fraction cap** *Fixed* $\gamma_{\text{down}} = 0.5$

| $\rho_{\text{down}}$ | 0% | 5% | 10% | 15% | 20% |
|---|---|---|---|---|---|
| Avg. ↑ | 50.4 | **50.7** | 48.1 | 46.6 | 44.6 |
| $\Delta$ (pp) | 0.0 | +0.3 | −2.3 | −3.8 | −5.8 |

**(b) Downscale** *Fixed* $\rho_{\text{down}} = 5\%$

| $\gamma_{\text{down}}$ | −1.0 | −0.5 | 0 | 0.25 | 0.5 | 0.75 | 1.0 | 1.5 |
|---|---|---|---|---|---|---|---|---|
| Avg. ↑ | 49.2 | **51.0** | 48.4 | 48.0 | 50.7 | 48.6 | 47.3 | 45.5 |
| $\Delta$ (pp) | −1.2 | +0.6 | −2.0 | −2.4 | +0.3 | −1.8 | −3.1 | −4.9 |

$\rho_{\text{down}}$ caps the selected fraction; only negative-margin positions are eligible. On selected positions, $\gamma_{\text{down}} < 0$ reverses the advantage sign, $\gamma_{\text{down}} = 0$ zeros the advantage, and $0 < \gamma_{\text{down}} < 1$ attenuates it. $\gamma_{\text{down}} = 1$ sets unit weight, including at positions that would otherwise be amplified; it need not recover the main rule. $\gamma_{\text{down}} > 1$ amplifies the selected advantages.

**Limited adjustments yield modest additional gains.** The main positive-weighting rule achieves 50.4, improving over GRPO (44.8) by 5.6 points. With a 5% cap, attenuation at $\gamma_{\text{down}} = 0.5$ reaches 50.7, while sign reversal at $\gamma_{\text{down}} = -0.5$ reaches 51.0, the highest mean in the multiplier sweep. These additional gains of 0.3 and 0.6 points are sensitive to the adjustment settings: increasing the attenuation cap to 10% and 20% reduces the score to 48.1 and 44.6, respectively, while a reversal multiplier of $-1$ yields 49.2. The sweep therefore supports exploring limited adjustments, with broader or stronger adjustments capable of degrading performance.

**Interpretation.** The extension uses a source-relative outcome-affinity margin as an additional selection criterion. At selected negative-margin positions, a positive multiplier below one attenuates the inherited advantage, while a negative multiplier reverses its sign; the rule applies to both outcome classes. The margin compares graph-hitting probabilities with a prompt-source reference and supplies an additional heuristic for adjusting supervision. The aggregate training scores suggest potential benefits from this heuristic, without establishing the correctness of individual token-level sign decisions. We retain the positive weighting rule in Eq. (7) for all main experiments, which demonstrate the effectiveness of outcome-conditioned supervision allocation across models and tasks. Affinity-based adjustment remains an exploratory extension.

### D.8 Runtime Overhead

**Measurement.** We measure the complete flow-computation overhead $T_{\text{flow}}$, including the additional inference pass for attention extraction and all subsequent graph computations, alongside the full training-step time $T_{\text{step}}$ for FlowWeaver. Both are in seconds and exclude evaluation. Table 18 reports the overhead as a share of the complete step, $100T_{\text{flow}}/T_{\text{step}}$. Mathematical settings specify the context length; both ALFWorld measurements use the full task mixture, including an additional measurement with post-trained Qwen3-8B.

Table 18: Per-step runtime of FlowWeaver, excluding evaluation. $T_{\text{flow}}$ includes attention-extraction inference and all subsequent flow computations; $T_{\text{step}}$ is the complete training-step time. Times are in seconds; Share is $100T_{\text{flow}}/T_{\text{step}}$.

| **Model** | **Setting** | $T_{\text{flow}}$ (s) | $T_{\text{step}}$ (s) | **Share (%)** |
|---|---|---|---|---|
| *Mathematical reasoning* | | | | |
| Qwen3-4B-Base | 1K | 1.06 | 58.51 | 1.81 |
| Qwen3-4B-Base | 8K | 3.99 | 183.61 | 2.17 |
| Qwen3-8B-Base | 1K | 0.94 | 64.87 | 1.45 |
| Qwen3-8B-Base | 8K | 10.19 | 615.88 | 1.65 |
| DeepSeek-R1-Distill-Llama-8B | 1K | 0.88 | 49.60 | 1.77 |
| *Agentic tasks* | | | | |
| Qwen3-4B-Base | HotpotQA | 0.34 | 50.28 | 0.68 |
| Qwen3-4B-Base | WebShop | 0.32 | 96.23 | 0.33 |
| Qwen3-4B | $\text{ALFWorld}_{\text{full}}$ | 0.33 | 52.53 | 0.63 |
| Qwen3-8B | $\text{ALFWorld}_{\text{full}}$ | 0.39 | 66.74 | 0.58 |

**Low overhead, especially in agentic training.** On mathematical reasoning, the complete flow pipeline takes 0.88–10.19 seconds and accounts for 1.45–2.17% of the training-step time. Its share remains below 2.2% in both 8K settings. For agentic tasks, the total overhead is only 0.32–0.39 seconds per step, or 0.33–0.68% of step time. Thus, the added computation has a negligible relative cost in the measured agentic configurations.

**Why the computation is inexpensive.** The implementation in Appendix C.1 helps explain these small shares. A single additional inference pass replays completed trajectories through the sampling engine, processing the already available token positions in parallel. We retain middle-layer attention averages, then compute local flows, flow-progress matches, and joint outcome-conditioned flows using batched tensor operations. Independent trajectories and prompt groups run in parallel, while native kernels and overlap weights are reused. The joint DAG requires finite backward and forward passes, with both outcome channels propagated together. In agentic training, multi-turn generation and environment or tool interactions also contribute to step time, while this attribution pipeline remains fast.