CIPI: WHEN TO INTEGRATE PEER FEEDBACK IN MULTI-AGENT LLM REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Conventional round-based multi-agent LLM systems integrate peer feedback at predefined interaction rounds, without considering whether the receiving agent needs an immediate correction or whether the feedback can wait. This fixed timing may delay necessary corrections or interrupt reasoning that is already correcting itself. We introduce CIPI (Conditional Integration of Peer Information), a protocol that lets the receiving agent decide when to use feedback after it arrives. CIPI stores peer feedback and uses the agent's current reasoning state to decide when to review it. The agent reviews feedback early when a contradiction or misunderstanding of the question could affect the next reasoning step and is not already being corrected. Otherwise, feedback is saved for later review. The model checks proposed corrections before accepting them and records accepted changes to guide later reasoning. Early corrections do not prevent further review, and a final check takes place before the answer is generated. The key idea is to use peer feedback when the receiving agent needs it. We evaluate CIPI on MuSiQue, HotpotQA, MATH-500, OlymMATH, APPS, and HumanEval, covering multi-hop question answering, mathematical reasoning, and code generation. We compare CIPI with output-level message passing, adapted Exchange-of-Thought Report, and message passing augmented with Self-Refine, evaluating task performance and measured token usage. Under the evaluated role-specific thinking settings, CIPI achieves the highest reported scores across the six benchmarks, improving over message passing with Self-Refine by 0.84–9.00 percentage points, generally with higher token usage. These findings suggest the value of selectively integrating peer feedback during ongoing reasoning to support timely correction without unnecessary disruption.

1 INTRODUCTION

Multi-agent reasoning systems use large language models (LLMs) to propose solutions, review intermediate work, and exchange corrections. Round-based debate organizes these interactions into scheduled exchanges (Du et al., 2024; Liang et al., 2024; Yin et al., 2023). A correction may become useful before the next scheduled exchange. Another suggestion may require more reasoning before the receiving agent can assess it. The timing of an update therefore matters alongside the content of the feedback.

An interaction schedule determines when information is exchanged, but does not by itself provide a separate rule for acting on available feedback. Debate changes answers through successive rounds, while Self-Refine alternates criticism and revision (Madaan et al., 2023). Selective communication methods change agent participation or message routing (Zeng et al., 2025; Zhang et al., 2025; Wang et al., 2025). The decision considered here is whether a buffered comment warrants a change to the receiving agent's current reasoning state.

Conditional Integration of Peer Information (CIPI) separates receiving feedback from integrating it into ongoing reasoning. A main agent produces intermediate states in successive model calls. A peer reviews each state and sends its comments to a buffer. Between calls, a separate gate assesses the receiving agent's state and nominates comments about an active error that could affect its next step. An integrator proposes an update, and a verifier checks accepted peer corrections before com-

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.

mitment. These are distinct model calls orchestrated by the controller. Deferred comments remain available for terminal review before answer generation.

Feedback verification and selective communication have established precedents (Lin et al., 2024; Zeng et al., 2025; Shang et al., 2026). CIPI focuses on the receiving agent's decision to act on a pending correction within one reasoning trajectory. Inspection determines when feedback is reviewed, while commitment determines whether the resulting update changes the state.

Our contributions are twofold:

- **A receiver-side integration protocol.** We define state-dependent inspection and verified commitment over a reasoning state, a feedback buffer and a correction record. The controller supports repeated review and recovery from malformed proposals while preserving the last valid state.
- **An evaluation separating system performance from timing and selection.** We compare CIPI with output-level message passing (MP), adapted Exchange-of-Thought Report (EoT-Report), and MP with Self-Refine (MP+SR) on six benchmarks spanning multi-hop question answering, mathematics and code generation. Four-arm ablations on MuSiQue, OlymMATH and HumanEval use CIPI-All, Always-All and Random-All to distinguish the inspection schedule from comment selection.

2 Related Work

Correction needs reliable feedback. Self-Refine alternates a model's feedback and revision (Madaan et al., 2023), and Reflexion stores verbal feedback across trials (Shinn et al., 2023). Their feedback sources and task conditions matter. Huang et al. (2024) find that intrinsic self-correction can degrade reasoning without external feedback. Kamoi et al. (2024) distinguish this setting from correction supported by reliable external signals or training. CriticBench separately evaluates generation, criticism, and correction (Lin et al., 2024). These studies motivate distinguishing feedback availability from its reliability. CIPI addresses when available feedback should be inspected within an ongoing reasoning trajectory.

Communication topology and selection. Symmetric debate exposes solvers to one another's answers (Du et al., 2024). The affirmative/negative/moderator arrangement of Liang et al. (2024) instead assigns different roles. Exchange-of-Thought varies message routing and supplies confidence information (Yin et al., 2023). These communication topologies determine who receives which message. Receiver-side integration adds a decision about when a received comment should prompt a change to the solver's state.

S^2-MAD sparsifies exchanges and participation to remove redundant debate (Zeng et al., 2025). AgentPrune removes communication links, and AgentDropout removes redundant agents (Zhang et al., 2025; Wang et al., 2025). GPTSwarm optimizes the computation graphs used by language agents (Zhuge et al., 2024). Learned targeting and communication gates also appear in TarMAC and IC3Net (Das et al., 2019; Singh et al., 2019). CIPI operates at the level of a pending comment about an active intermediate claim, complementing selection over agents and communication links.

Recent gating and verification. Pre-commit gating of reusable skills addresses contamination across an agent's successive learning rounds (Shang et al., 2026). CIPI applies verification before commitment to a different object: a proposed change to the reasoning state within one task. DebateCoder gates continued code deliberation using a confidence threshold (Zhang et al., 2026). AgentDropoutV2 rectifies or rejects messages using error indicators distilled from past failures (Wang et al., 2026). These methods gate different events, including skill reuse, further deliberation and message acceptance. CIPI's focus is inspection timing for feedback already available to the receiver. A debate taxonomy likewise distinguishes participants, interaction rules and agreement rules as separate aspects of a multi-agent system (Motger et al., 2026).

3 Method

Figure 1 summarizes the controller's flow between separate model calls. The controller provides up to three checkpoints, one peer, a buffer of pending comments, and a verification step for accepted

peer corrections. A checkpoint is the boundary after a completed main agent call and its peer review, before the next main agent call or final answer. Integration changes the explicit state passed to a subsequent call. It does not interrupt token generation or modify a model's hidden chain of thought. Separate gate, integrator, and verifier calls assess whether to inspect feedback, propose a state update, and judge accepted peer corrections, respectively. They use the receiving agent's state but are not decisions emitted inside the main agent call. HotpotQA enables guarded stopping and skips integration with no feedback or active corrections.

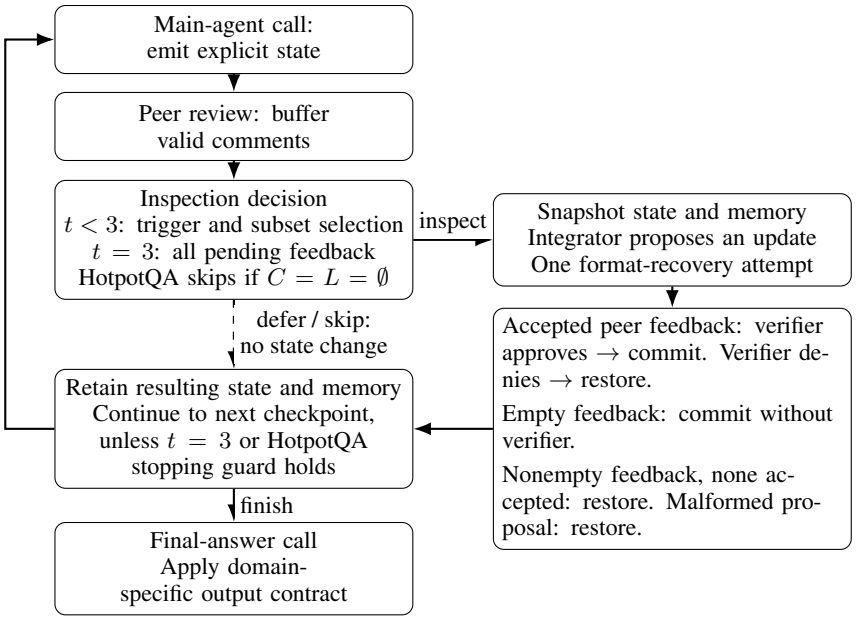

Figure 1: **Control flow between model calls, starting at $t = 1$.** The update branches distinguish verification of accepted peer feedback, unverified empty-buffer self-consolidation, and local rollback. A committed update also updates hypothesis history. Terminal rollback or a skip still leads to final answer generation. The HotpotQA guard can finish after checkpoint 2 without a third main/peer call. Operational admission or transport stops and exhausted main/verifier/final format recovery terminate with a recorded failure rather than following a continuation arrow.

3.1 State and Feedback Memory

For a task x, let X map public source identifiers to text. At checkpoints $t \in \{1, 2, 3\}$ the main agent emits a reasoning string and a state

$$z_t = (y_t, S_t, U_t, n_t), \tag{1}$$

where the conclusion y_t and next step n_t are strings, and supporting statements S_t and uncertainties U_t are lists of strings. In Eq. 1, S_t contains text written by the model and is not a validated subset of X. Schema checks establish field types, whereas source membership and semantic support require separate checks. Later calls receive the current state, a correction record L_t, and hypothesis history H_t.

One peer reviews each state and returns either silence or $c = (q, r, \sigma, s)$. The tuple contains, in order, the target, critique, list of source identifiers, and suggestion. Each text field has 1 to 280 characters after trimming surrounding whitespace. The source list is nonempty and refers to supplied evidence. The buffer C_t merges comments only when all four fields, including the ordered source list, are equal. A repeated issue receives a new identifier after its previous occurrence has been resolved. With one comment per checkpoint, at most three distinct comments accumulate in this configuration. Evidence selection and limits on request size are recorded separately (Appendix A).

The correction record stores accepted changes with their sources and checkpoints. A proposal assigns each selected comment a status (accepted, rejected, unresolved), and each existing correction

a retain/revoke decision. A committed proposal removes accepted and rejected comments from the pending buffer. Unresolved comments remain in the pending buffer for later review. Rejected proposals leave the buffer and correction record unchanged, allowing reconsideration at a later checkpoint.

3.2 Early-Inspection Rule

When the pending buffer is nonempty at $t = 1, 2$, the gate receives the question, current state, pending comments, and their cited source text. One call returns the assessment $(k_t, \varepsilon_t, \eta_t, \rho_t, D_t, E_t, \ell_t)$. Its fields are, in order, an issue class, three Boolean judgments, selected comment IDs, evidence IDs, and an explanation. The issue class is contradiction, answer-target mismatch, support gap, or none. The Booleans indicate an active error, predicted harm to the stated next step, and an issue already being addressed. These fields describe one joint assessment of the nominated comments, not independent assessments of every comment.

Let $V_t = 1$ when the response satisfies all structural conditions. These require exactly the prescribed fields and types, a permitted issue class, and unique IDs drawn from the supplied comments and evidence. The explanation must contain at most 16 words separated by whitespace. If $\varepsilon_t \eta_t (1-\rho_t) = 1$, validation additionally requires D_t, E_t, ℓ_t to be nonempty and $E_t \subseteq \bigcup_{c:\, \mathrm{id}(c) \in D_t} \sigma(c)$, where $\sigma(c)$ is the comment's cited set of source identifiers. This is a subset check, not a requirement to cite every source of every selected comment. Invalid output has $V_t = 0$, and the binary trigger is

$$I_t = V_t \, \mathbf{1}\{k_t \in \{\text{contradiction}, \text{answer-target}\}\} \varepsilon_t \eta_t (1 - \rho_t). \tag{2}$$

Equation 2 determines whether to request early inspection, not which comments to inspect. The feedback input passed to the integrator is defined separately as follows.

$$C_t^* = \begin{cases} \{c \in C_t : \mathrm{id}(c) \in D_t\}, & t < 3 \text{ and } I_t = 1, \\ \emptyset, & t < 3 \text{ and } I_t = 0, \\ C_t, & t = 3. \end{cases} \tag{3}$$

The selected subset retains buffer order and is not reclassified comment by comment. For an absent or invalid assessment, I_t is set to zero without evaluating missing fields. An empty buffer, support-gap/none classification, or valid wait decision also gives no early inspection. A requested inspection executes only after admission checks. Budget or transport stops count as operational failures and do not count as wait decisions. Harm to the next step is predicted from n_t without executing a counterfactual next step to measure that harm. All semantic fields are model judgments, not independent error labels. Mathematical contradictions include checkable arithmetic, algebraic, or constraint violations. QA target mismatches concern the requested entity, relation, or answer slot.

A wait decision is cached only for an identical serialized gate payload, including the state and cited evidence. A change to the state invalidates the cached payload match. Cache hits are an implementation optimization whose frequency requires measurement. When reached, checkpoint 3 bypasses the early gate and processes all pending feedback.

3.3 Proposal, Verification, and State Update

For selected comments C_t^*, the integrator samples

$$(\widetilde{z}_t, R_t, J_t) \sim p_\theta^{\mathrm{int}}(\cdot \mid x, X, z_t, L_t, C_t^*, H_t), \tag{4}$$

where θ denotes the model parameters, p_θ^{int} is the integrator response distribution, R_t contains comment decisions and J_t reviews existing corrections. Equation 4 is a stochastic model operation followed by deterministic schema checks. When at least one comment is accepted, a separate verifier receives the original and candidate states, source text, selected feedback, both decision records, and H_t. A positive verdict covers every accepted comment and its cited evidence. The controller commits state and memory together only after a positive verdict. A negative verdict restores the previous state and memory. With no accepted comments, the controller rolls back locally without a verifier call.

An early intervention leaves the next checkpoint eligible. A pending comment can be proposed at most once per checkpoint, apart from bounded format retries. At the terminal checkpoint, an

empty buffer permits self-consolidation without peer verification. This branch can revise the state and retain or revoke existing corrections after structural checks, but makes no separate verifier call. HotpotQA skips this call when the correction record is also empty. Thus verification covers accepted peer corrections, not every state transition or final answer. Always and Random retain the terminal behavior of their full-CIPI anchor (Table 2).

HotpotQA skips the third main and peer calls only when all stopping conditions hold. Checkpoints 1 and 2 must have identical nonempty conclusions after case and whitespace normalization, nonblank supporting text, no unresolved uncertainty, and valid silent peer reviews. The buffer and correction record must also be empty. An invalid peer response does not count as silence. The retained state is not logged as a new state at the third checkpoint. Final answer selection still runs after the guarded stop. All other adapters continue to execute all three checkpoints.

3.4 History, Final Answers, and Recovery

Both the QA and mathematical adapters initialize H to an empty list. After each main state and committed integration, a nonempty conclusion is compared by the hash of its full serialized string against retained entries. A new distinct conclusion is appended to the history. Duplicates already retained in the history do not move to the end. Three main states and at most three committed integrations produce at most six entries, so history eviction is not exercised in this configuration. Each stored excerpt is at most 1,200 characters. Main, integrator, verifier, and final answer calls receive the history. The prompts identify the retained history as unverified hypotheses.

QA final selection emits up to three candidates with quoted sources, requested/selected type labels, and one selected identifier. Validation checks quotation membership, field structure, and label consistency. The selected phrase provides the answer used for scoring. In contrast, the communication baselines emit an answer string without CIPI's candidate, quotation, or checks that match answer types. This difference affects answer generation and is not merely a common scoring transformation. The mathematical adapter retains the same controller and memory and returns a mathematical answer of at most 4,096 characters. This answer may use deductions from the problem without requiring a verbatim source quote.

Main, integrator, verifier, and final roles permit one format retry per call site. Diagnostics from integrator and final calls name the violated schema constraints. Two malformed integration proposals restore the snapshot and continue. Two malformed main, verifier, or final responses end the episode under its recorded failure category. Invalid peer comments are discarded, and invalid gate assessments defer. Recovery receives no reference answers, and every rollback preserves recorded costs and events. Algorithm 1 gives the ordering of the implemented controller operations.

3.5 Mechanism and Testable Contrasts

The inspection gate controls both the early trigger and the nominated feedback subset. The verifier controls commitment when a proposal accepts peer feedback. The current Always and Random specifications retain the same solver, peer, verifier, memory, and final answer contract. Always inspects every pending comment at each intermediate checkpoint. Random inspects the same pending comments on a frozen random draw independent of the state's content. Both policies retain the same final review as CIPI. These variants inspect all pending comments, whereas CIPI may nominate a subset. Their comparisons therefore combine changes in timing and comment selection and cannot isolate timing alone. The four-arm study therefore adds CIPI-All, which keeps the conditional trigger but inspects all pending comments. Comparing CIPI-All with Always-All and Random-All holds the inclusion rule fixed; comparing full CIPI with CIPI-All evaluates selection under the conditional trigger. A verifier using the same model can share the solver's errors, so a successful schema check or positive verdict is not a correctness guarantee.

Algorithm 1 CIPI with up to three checkpoints. The FORMAT operation permits one attempt to recover from a schema violation.

Require: Task x, source map X, role contracts, admission rules; $e = 1$ for the HotpotQA efficiency adapter, 0 otherwise

Initialize z as absent; $C, L, H \leftarrow \emptyset$. Every call requires admission
for $t = 1, 2, 3$ **do**
  If e and $t = 3$ and guarded stopping holds, exit loop with the retained state
  $z \leftarrow \text{FORMAT}(\text{Main}, x, X, z, L, H)$. Stop on failure
  $H \leftarrow \text{UPDATEHISTORY}(H, z)$
  Query peer and add valid nonduplicate comments to C
  Stop on peer transport/admission failure. Discard malformed peer comments
  If $t < 3$ and $C \neq \emptyset$, obtain a gate assessment (or cached wait). Stop on operational failure
  If $t < 3$, set I by Eq. 2. Empty buffer, invalid assessment, or cached wait gives zero
  Set C^* by Eq. 3. Checkpoint 3 bypasses the gate
  if $C^* \neq \emptyset$ or $t = 3$ **then**
    If e and $C^* = L = \emptyset$, skip integration and continue
    Snapshot (z, L, C, H). Sample by Eq. 4 with recovery. Stop on operational failure
    if proposal still malformed **then**
      Restore snapshot, retaining costs. Continue loop (then Final if $t = 3$)
    else if $C^* = \emptyset$ **then**
      Commit terminal self-consolidation and update H
    else if no comment accepted **then**
      Restore snapshot without querying the verifier
    else
      $v \leftarrow \text{FORMAT}(\text{Verify}, x, X, z, \widetilde{z}, L, C^*, R, J, H)$
      Stop on invalid v. On approval, commit and update H. Otherwise restore snapshot
    end if
  end if
end for
$a \leftarrow \text{FORMAT}(\text{Final}, x, X, z, L, H)$ using the domain contract
return a or the explicit failure status, retaining all attempt costs

4 EXPERIMENTS

4.1 COMPLETED COMPARISONS AND BENCHMARK SCOPE

The completed comparison covers MATH-500, MuSiQue, HotpotQA, OlymMATH and APPS with 100 tasks each, and all 164 HumanEval tasks. Four methods share each task list. The table summarizes 2,656 task–method outcomes, retaining invalid outputs in each denominator. The initial four-method batches randomize method order within tasks; batch provenance is specified in the accompanying Evaluation Notes. MATH-500 tests mathematical correctness (Hendrycks et al., 2021b; Lightman et al., 2024), while MuSiQue and HotpotQA test multi-hop QA using supplied contexts (Trivedi et al., 2022; Yang et al., 2018).

The reported batches use the diagnostic-recovery CIPI controller with adapters specific to each benchmark. HotpotQA's efficiency revision combines concise-answer guidance and reference-blind yes/no normalization for all methods with CIPI-specific answer targeting and guarded skips of empty integration or redundant review (Appendix A). It applies only to HotpotQA because it was introduced in a separate diagnostic rerun of that benchmark, leaving the other benchmark configurations unchanged rather than deploying a uniform method-version update. MATH-500 uses the level quotas 19/21/20/20/20 when selecting its tasks. MuSiQue uses 34 two-hop, 33 three-hop and 33 four-hop questions, with possible overlap between decomposition components.

OlymMATH covers competition mathematics and HumanEval evaluates code with its original tests (Sun et al., 2026; Chen et al., 2021). APPS is the sixth benchmark (Hendrycks et al., 2021a), using a fixed 100-task Interview-difficulty test subset. Final programs are evaluated against the provided test cases in isolated Python execution. The separate timing-ablation study covers MuSiQue, OlymMATH and HumanEval; its results do not replace the main comparison.

4.2 COMMUNICATION BASELINES

Output-level message passing. MP uses an affirmative solver, a negative solver and a moderator (Liang et al., 2024). In each of at most three rounds, the affirmative solver acts first and the negative solver sees its current response. From round 2, the affirmative solver sees the preceding negative response. A nonempty moderator answer ends the run, and an empty answer requests the next round. If all three moderator answers are empty, two judge calls extract candidates from the original responses from the first round and select the final answer. Usage from both judge calls is included in the reported total. This bounded implementation uses JavaScript Object Notation (JSON) outputs and adapts the baseline (Appendix D.2).

Exchange-of-Thought Report. EoT-Report uses three nodes in a synchronous star (Yin et al., 2023). Each round passes answers from the previous round, reasoning and modal-history confidence between the center and its two neighbors. Confidence is the size of the largest group of equivalent answers divided by valid history length. The recipient is prompted to consider it, without a numeric controller weight. The run stops when at least two current answers agree. It returns the first member of the largest group whose answers agree pairwise, using the fixed order from center to peripherals. Agreement between only the peripheral nodes therefore returns a peripheral answer. Only at the cap of six rounds without agreement does it return the center's answer with a cap flag.

Message passing with Self-Refine. MP+SR samples its own MP trajectory, then permits up to two critique/revision cycles (Madaan et al., 2023). A critic receives the task and answer and returns a Boolean revision request with a reason. A false request ends refinement with the current answer. A true request invokes one refiner whose output replaces the current answer. After a second revision, the revised answer is returned without a third critique. An invalid initial MP result is returned without refinement. This adaptation uses neither hidden answers nor nested debate inside the critic.

4.3 SCORING AND USAGE

QA reports token-overlap F1; the MuSiQue ablation additionally reports exact match (EM). Mathematics reports whether the final answer is correct. Code reports the fraction of tasks whose final programs pass the specified tests. The frozen output contracts determine whether a final answer is valid. Invalid outputs and unresolved mathematical answers receive zero primary score. CIPI's QA selector requires typed candidates with source quotations, whereas MP, EoT-Report and MP+SR return answer strings without those checks. HotpotQA shares guidance for concise answers and leading yes/no normalization across methods, but not the full selector.

The requested model identifier is `deepseek-flash`, with requested temperature 0.6. Thinking is enabled at low effort for CIPI's peer, integrator and verifier, for MP's moderator and judges, and for MP+SR's moderator, judges, critic and refiner. EoT-Report keeps thinking disabled for every node in the comparison. Other primary roles disable thinking, with recovery exceptions documented in Appendix A. Section 6 discusses how these configuration differences affect interpretation.

Measured usage sums input and output tokens reported by the provider from every persisted response, including retries and rejected proposals. Reasoning tokens already included in completion usage are counted once. Cached tokens remain included in the input token count. Missing usage is reported separately from the operational reservation used for request admission.

4.4 TASK PERFORMANCE AND COST

Table 1 shows the completed results. CIPI has the highest mean on the evaluated MATH-500, MuSiQue and HotpotQA samples. Relative to MP+SR, CIPI's accuracy is 98.00% versus 93.00% on MATH-500. Its F1 is 65.78 versus 64.94 on MuSiQue. On HotpotQA, CIPI reaches 83.37 F1 versus 79.90 for MP, 81.04 for EoT-Report and 81.44 for MP+SR. CIPI scores 87.00% on Olym-MATH versus MP's 83.00%, EoT-Report's 67.00% and MP+SR's 78.00%. On HumanEval, CIPI passes 163/164 tasks (99.39%), versus 160/164 for EoT-Report, 157/164 for MP and 159/164 for MP+SR. On APPS, CIPI passes 90/100 tasks, above MP (87/100), MP+SR (86/100) and EoT-Report (77/100).

Table 1: **Main comparison: 100 tasks per benchmark except HumanEval (164).** Thinking settings vary by role. HotpotQA uses the benchmark-specific efficiency revision described in Section 4 on previously evaluated tasks. Scores are percentages (Acc., accuracy). Tok. denotes mean measured input/output tokens per task, bold type the highest score without implying significance, with APPS restricted to Interview difficulty.

	MATH-500		MuSiQue		HotpotQA	
Method	Acc.	Tok.	F1	Tok.	F1	Tok.
MP	89.00	4,283	61.71	13,082	79.90	7,396
EoT-Report	95.00	2,132	61.53	12,008	81.04	5,080
MP+SR	93.00	4,944	64.94	18,224	81.44	9,481
CIPI	**98.00**	7,100	**65.78**	23,492	**83.37**	12,345

	OlymMATH		APPS Interview (100)		HumanEval	
Method	Acc.	Tok.	Pass rate	Tok.	Pass rate	Tok.
MP	83.00	29,431	87.00	32,661	95.73	6,164
EoT-Report	67.00	39,612	77.00	47,592	97.56	7,378
MP+SR	78.00	36,354	86.00	39,582	96.95	7,915
CIPI	**87.00**	37,662	**90.00**	40,066	**99.39**	10,677

The paired estimates are imprecise at this sample size. For CIPI minus MP+SR, the 95% bootstrap interval over tasks is $[-1.00, 11.00]$ percentage points on MATH-500. Intervals use 10,000 paired resamples and seed 20260919, without confirmatory multiplicity claims.

CIPI consumes more recorded tokens than every communication baseline in the first three batches. Relative to MP+SR, the measured token ratios are 1.44 on MATH-500, 1.29 on MuSiQue, 1.30 on HotpotQA, 1.04 on OlymMATH, 1.35 on HumanEval and 1.01 on APPS. On OlymMATH, the reported score is 9.00 points above MP+SR with approximately 4% more recorded tokens. The evidence does not support a general claim of token savings.

4.5 Timing and Selection Ablations

The four-arm study distinguishes feedback selection from timing. CIPI uses its normal gate and selected comments. CIPI-All keeps the same state-dependent trigger but inspects all pending comments when triggered. Always-All inspects all pending comments at every eligible intermediate checkpoint. Random-All uses a seeded Bernoulli draw at each nonempty-buffer checkpoint and, when triggered, inspects all pending comments. All arms retain the integrator, verifier, rollback and terminal review; inspection does not force acceptance.

The completed study contains 100 matched tasks per arm on MuSiQue, OlymMATH and HumanEval, totaling 1,200 task–arm outcomes. Random-All probabilities are frozen from development checkpoints before formal generation. Table 2 reports task scores and measured usage; Appendix C.1 reports paired contrasts.

Table 2: **Timing and selection ablations: 100 tasks per arm and dataset.** Scores are percentages. Tok. denotes mean measured tokens per scheduled task.

	MuSiQue			OlymMATH		HumanEval	
Arm	F1	EM	Tok.	Acc.	Tok.	Pass rate	Tok.
CIPI	**70.40**	**62.00**	42,479	**89.00**	30,637	**98.00**	12,078
CIPI-All	68.34	57.00	42,114	88.00	32,439	97.00	11,727
Always-All	64.66	54.00	43,372	88.00	33,649	96.00	11,747
Random-All	68.40	58.00	43,007	88.00	33,501	96.00	11,383

On MuSiQue, CIPI-All exceeds Always-All by 3.675 F1 points but trails Random-All by 0.066 points, while CIPI exceeds CIPI-All by 2.060 points. On OlymMATH, CIPI reaches 89.00%, one point above CIPI-All, Always-All and Random-All at 88.00%, with the lowest recorded token us-

age of the four arms. On HumanEval, CIPI reaches 98.00%, followed by CIPI-All at 97.00% and Always-All and Random-All at 96.00%. The CIPI-All timing contrasts hold the all-comment selection rule fixed; CIPI versus CIPI-All evaluates selective feedback under the conditional trigger. CIPI-All exceeds both timing controls by 1.00 point on HumanEval, and CIPI exceeds CIPI-All by 1.00 point. These point estimates do not by themselves establish a statistically significant timing advantage.

4.6 Matched Illustrative Trace

The trace uses the first task in the archived four-method MuSiQue order, `4hop1__15607_59747_211319_557671`. It asks for a bordering county reached through a chain of state, city and county relations. The selection of this case for presentation is retrospective. The case illustrates behavior on a common input rather than estimating how often that behavior occurs.

CIPI's first state names Georgia, Atlanta and Fulton County. The peer flags the county link as unsupported by the supplied passages. Both decisions at the intermediate gates defer feedback inspection. At checkpoint 3, the verifier approves a revision marking the unresolved link, and the final answer is "Unable to determine." EoT-Report and MP+SR return "DeKalb County," while MP returns "Cobb County or DeKalb County." All four receive zero exact match. Their F1 scores are 40.00, 40.00, 25.00 and 0.00 for EoT-Report, MP+SR, MP and CIPI, respectively. Measured usage is 9,980, 21,735, 17,050 and 48,206 tokens in the same order. This case shows that a verified revision related to evidence need not improve benchmark F1.

5 Conclusion

CIPI makes feedback integration a separate decision within an agent's ongoing reasoning. Its buffer allows a comment to remain available while the main agent continues, and its checkpoints permit inspection before the terminal review. Verification and recovery determine which proposed updates become committed states.

CIPI has the highest reported score among the four systems on all six evaluated benchmarks. Relative to MP+SR, the gains range from 0.84 F1 points on MuSiQue to 9.00 accuracy points on OlymMATH, with measured token ratios of 1.01 to 1.44 across benchmarks. The ablations give a more specific picture. Conditional all-comment inspection improves MuSiQue F1 over Always-All but is nearly tied with Random-All. Full CIPI exceeds all three OlymMATH controls by one accuracy point while using fewer recorded tokens, and has the highest HumanEval ablation pass rate. State-dependent integration therefore offers a configurable review mechanism whose score and cost should be evaluated together.

TypeSafe's Jev evaluates typed questions against a supplied state and returns structured answers with probability distributions (TypeSafe AI, n.d.). This interface suggests a possible decision layer for CIPI's feedback-inspection gate, while a generative model would continue to produce and revise the reasoning state. Such an integration would require an adapter to preserve CIPI's evidence identifiers and explanation fields. Whether it improves reasoning quality or computational cost remains an open empirical question.

6 Limitations

The study uses one hosted model family. Role-specific thinking settings, output-validation rules and unequal computation mean that the main comparison evaluates complete systems rather than isolating protocol effects. The evaluation does not establish superiority over selective communication methods that are not included as comparators. MuSiQue components can overlap across experiments, HumanEval has prior project exposure, and APPS covers 100 Interview tasks rather than the full benchmark. Harmful revisions have not been quantified, so reduced harmful revision is not an established result. Same-model verifiers can share solver errors, and provider identifiers do not guarantee a stable model snapshot.

Reproducibility Statement

Algorithm 1 and Appendix A specify control flow, role interfaces and request limits. Appendix A.5 describes the local source companion and outstanding requirements for public release. The accompanying numerical audit checks archived file hashes, recomputes task means, and reconciles available request usage. A public reproduction package with anonymous hosting remains pending.

Ethics Statement

The study uses public benchmark tasks and restricts reference answers to scoring. Generated programs require an isolated execution environment before testing, and dataset licenses govern redistribution of task content. These research outputs have not been validated for deployment decisions.

AI Use Statement

We used OpenAI Codex as an LLM-based assistant for manuscript drafting and revision, literature search and citation checking, discussion of experimental design, implementation and debugging of experimental code, and organization and interpretation of results. The authors directed the revisions and retain full responsibility for the final text, claims, code and reported results. The language models used as experimental agents and their settings are described in Section 4 and Appendix A.

References

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. doi: 10.48550/arXiv.2107.03374. URL https://arxiv.org/abs/2107.03374.

Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. TarMAC: Targeted multi-agent communication. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1538–1546. PMLR, 2019. URL https://proceedings.mlr.press/v97/das19a.html.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11733–11763. PMLR, 2024. URL https://proceedings.mlr.press/v235/du24e.html.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring coding challenge competence with APPS. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021a. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/c24cd76e1ce41366a4bbe8a49b02a028-Abstract-round2.html.

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. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021b. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 32808–32824, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/8b4add8b0aa8749d80a34ca5d941c355-Paper-Conference.pdf.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. When can LLMs actually correct their own mistakes? a critical survey of self-correction of LLMs. *Transactions of the Association for Computational Linguistics*, 12:1417–1440, 2024. doi: 10.1162/tacl_a_00713. URL https://aclanthology.org/2024.tacl-1.78/.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17889–17904, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.992. URL https://aclanthology.org/2024.emnlp-main.992/.

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. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Zicheng Lin, Zhibin Gou, Tian Liang, Ruilin Luo, Haowei Liu, and Yujiu Yang. CriticBench: Benchmarking LLMs for critique-correct reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 1552–1587, 2024. doi: 10.18653/v1/2024.findings-acl.91. URL https://aclanthology.org/2024.findings-acl.91/.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594. Curran Associates, Inc., 2023. doi: 10.52202/075280-2019. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/91edff07232fb1b55a505a9e9f6c0ff3-Paper-Conference.pdf.

Quim Motger, Marc Oriol, Jordi Marco, and Xavier Franch. Multi-agent debate strategies: Survey, taxonomy, and challenges. *arXiv preprint arXiv:2607.26212*, 2026. doi: 10.48550/arXiv.2607.26212. URL https://arxiv.org/abs/2607.26212.

Linfang Shang, Ming Xu, Yiding Sun, Tianle Xia, Lingxiang Hu, Lan Xu, and Ning Zheng. When self-evolution backfires: Pre-commit gating against skill contamination in LLM agents. *arXiv preprint arXiv:2608.05810*, 2026. doi: 10.48550/arXiv.2608.05810. URL https://arxiv.org/abs/2608.05810v2.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf.

Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. Learning when to communicate at scale in multi-agent cooperative and competitive tasks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rye7knCqK7.

Haoxiang Sun, Yingqian Min, Zhipeng Chen, Wayne Xin Zhao, and Ji-Rong Wen. Challenging the boundaries of reasoning: An olympiad-level math benchmark for large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17438–17457. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.792. URL https://aclanthology.org/2026.acl-long.792/.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10: 539–554, 2022. doi: 10.1162/tacl_a_00475. URL https://aclanthology.org/2022.tacl-1.31/.

TypeSafe AI. Introduction. Official documentation for Jev, n.d. URL https://docs.typesafe.ai/introduction. Accessed September 26, 2026.

Yutong Wang, Siyuan Xiong, Xuebo Liu, Wenkang Zhou, Liang Ding, Miao Zhang, and Min Zhang. AgentDropoutV2: Optimizing information flow in multi-agent systems via test-time rectify-or-reject pruning. *arXiv preprint arXiv:2602.23258*, 2026. doi: 10.48550/arXiv.2602.23258. URL https://arxiv.org/abs/2602.23258v2.

Zhexuan Wang, Yutong Wang, Xuebo Liu, Liang Ding, Miao Zhang, Jie Liu, and Min Zhang. AgentDropout: Dynamic agent elimination for token-efficient and high-performance LLM-based multi-agent collaboration. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings*

of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 24013–24035, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1170. URL https://aclanthology.org/2025.acl-long.1170/.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuanjing Huang, and Xipeng Qiu. Exchange-of-Thought: Enhancing large language model capabilities through cross-model communication. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15135–15153, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.936. URL https://aclanthology.org/2023.emnlp-main.936/.

Yuting Zeng, Weizhe Huang, Lei Jiang, Tongxuan Liu, XiTai Jin, Chen Tianying Tiana, Jing Li, and Xiaohua Xu. S^2-MAD: Breaking the token barrier to enhance multi-agent debate efficiency. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 9393–9408, 2025. doi: 10.18653/v1/2025.naacl-long.475. URL https://aclanthology.org/2025.naacl-long.475/.

Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Xu Yu, and Tianlong Chen. Cut the Crap: An economical communication pipeline for LLM-based multi-agent systems. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=LkzuPorQ5L.

Haoji Zhang, Yuzhe Li, Zhenqiang Liu, Chenyang Liu, Shenyang Zhang, and Yi Zhou. Adaptive confidence gating in multi-agent collaboration for efficient and optimized code generation. *arXiv preprint arXiv:2601.21469*, 2026. doi: 10.48550/arXiv.2601.21469. URL https://arxiv.org/abs/2601.21469.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. GPTSwarm: Language agents as optimizable graphs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62743–62767. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhuge24a.html.

A Implementation and Reproducibility

A.1 Implementation Boundary and Notation

The inspected QA controller is `diagnostic-recovery-musique-v1`, and the mathematical adapter is `diagnostic-recovery-math-v1`. Both use the same checkpoints, history, and rollback operations but differ in task guidance and final answer validation. The runner for the four methods is `abstract-stage1-v1`. The APPS CIPI adapter adds `apps-final-code-consistency-v1`: after final-code generation, one audit compares the program with the public problem, committed state and correction record. An explicitly anchored contradiction permits at most one semantic repair. Uncertain or invalid audits, and malformed or syntactically invalid repairs, retain the original program. No test execution or test feedback enters generation. Audit and repair costs are included in reported usage. These names identify local source revisions whose public release remains outstanding. The MuSiQue, OlymMATH and HumanEval four-arm ablations contain 1,200 scored outcomes.

HotpotQA uses a derived policy identified as `hotpot-efficient-answer-v1`. All four methods receive guidance for concise answers and reference-blind normalization of explicit leading yes/no answers. CIPI's reasoning and final selection prompts distinguish the requested relation or property from intermediate entities. Its diagnostic-recovery gate, verifier and rollback are retained. Empty integration is skipped only when both the feedback buffer and correction record are empty. The third reasoning/review slot is skipped only after two supported identical conclusions, two valid silent peer reviews, no unresolved uncertainty, and empty buffer and correction record. Final answer selection still runs after the guarded stop. This is a joint development revision, not an ablation isolating any one change. Existing role controls and request limits are inherited.

Table 3: **Controller notation.** The table defines the state, feedback, memory, gate, and decision quantities used by the controller.

Symbol	Meaning
x, X	Question and public source-ID-to-text map
$z_t = (y_t, S_t, U_t, n_t)$	Conclusion string, supporting-text list, uncertainty list, next-step string
C_t, C_t^*	Pending comments and the subset selected for inspection
L_t, H_t	Accepted-correction record and bounded unverified hypothesis history
$k_t, \varepsilon_t, \eta_t, \rho_t$	Joint issue class and active-error, predicted next-step-harm, already-addressed judgments
D_t, E_t, ℓ_t	Gate-nominated comment IDs, cited source IDs, and explanation
V_t, I_t	Structural-validity indicator and binary early-inspection decision
R_t, J_t	Proposed comment decisions and review of existing corrections

Each run stores the task list and order, exact request payloads, role controls, responses, and usage for each attempt. A configuration manifest records code, prompt, and input hashes. Checksums distinguish the outputs of completed generation from later scoring. Resumption preserves all completed records and the previously recorded expenditure. Execution identifiers and their seeding behavior are documented in the accompanying Evaluation Notes.

A.2 Role Limits and Request Controls

The later OlymMATH, HumanEval and APPS adapters raise main, peer, integrator and final output ceilings to 32,768 tokens, retaining the 1,024-token gate and 4,096-token verifier ceilings. OlymMATH retains a 768,000-token trajectory admission limit; HumanEval and APPS use 3,072,000. APPS uses a 131,072-token input ceiling; its final consistency audit uses the verifier ceiling and its one permitted repair uses the final-role ceiling. These are maximum admission limits, not observed consumption. The later bounded recovery and code interfaces are benchmark-specific, so the earlier limits below must not be read as a uniform cap across all datasets.

Integrator and verifier calls for recovery from formatting errors disable thinking in the inspected implementation. Effective request records identify these exceptions to the thinking settings. Baseline calls allow two total schema attempts and at most two network attempts within each schema

Table 4: **Output limits for the initial MATH-500 and QA batches.** All listed input limits, including the gate in the controller derived from QA and the mathematical adapter, are 32,768. These caps differ from typical consumption, and the gate also faces the smaller available allocation described below.

Method	Role	Output cap
Output-level MP	Affirmative / negative	8,192
	Moderator / extraction judge / selection judge	12,288
EoT-Report (adapted)	Each of the three nodes	8,192
MP+SR (adapted)	Initial MP roles	As above
	Critic / refiner	8,192 / 12,288
CIPI	Main / peer	8,192 / 8,192
	Gate / integrator	1,024 / 12,288
	Verifier / final	4,096 / 16,384

Table 5: **Thinking allocation in the main comparison.** Table 1 uses the allocation shown for the four methods.

Method	Disabled	Enabled (low)
Output-level MP	Debaters	Moderator, both judges
EoT-Report (adapted)	All nodes	None
MP+SR (adapted)	Debaters	Moderator/judges, critic, refiner
CIPI	Main, gate, final	Peer, integrator, verifier

attempt for the specified disconnection errors. CIPI's wrapper also records each attempt to transport a request. These retries differ from a new candidate solution sampled for quality.

For every dispatched call, the reservation for missing usage includes locally counted input and a framing margin of 32 tokens. The reservation also includes the role's output cap (Table 4). The local counter is itself calibrated, so this margin is not the only adjustment to raw tokenizer counts. Let B be the CIPI controller's configured allowance, initially 768,000 tokens, and $G = \lfloor 0.03B \rfloor$. The gate allowance is G minus gate expenditure so far, further capped at $\lfloor G/2 \rfloor$ at checkpoint 1. Thus a fresh episode allocates 23,040 tokens to gate calls, with at most 11,520 available initially. These limits do not represent amounts that are automatically spent. When an operational resume starts a fresh controller with $B = 768{,}000 - T_{\text{prior}}$, the gate allocation is recomputed from the remaining allowance. Previously persisted expenditure remains part of the recorded usage.

Write $b_r = i_r^{\max} + o_r^{\max}$ for the sum of a role's input and output caps. The controller reserves $(3-t)b_{\text{main}} + b_{\text{final}} + b_{\text{int}}$ for main, peer and gate calls at checkpoint t. Integration reserves $(3-t)b_{\text{main}} + b_{\text{final}} + \mathbf{1}\{t<3\}b_{\text{int}}$, adding b_{verifier} when feedback is nonempty. The verifier uses the same reserve without its own term. Final delivery reserves zero tokens for any subsequent calls. Future peers, all retries and every possible future verifier are not reserved in advance, so this is not a guarantee that every path will fit. The HotpotQA skip rules avoid calls but do not replace these inherited admission bounds.

A call is admitted only when its counted input plus margin fits its input cap. Expenditure plus the call reservation and reserve for future calls must also fit B. A gate call must additionally fit its available gate allocation. Otherwise the controller records an input, stop due to the total budget or gate budget rather than silently converting it to a semantic wait. Physical retry guards also check and enforce the remaining allowance. Reserves for future calls affect admission but are excluded from measured consumption. Only attempted calls incur usage or reservations for missing usage, and all costs survive rollback. The generic history routine retains the first and five subsequent distinct hypotheses. On overflow, it removes the second oldest entry. Overflow cannot occur in the configuration with three checkpoints described in Section 3.

A.3 Schemas and Effective Prompts

Table 6 summarizes the inspected interfaces. The same supplied source map is the reference for identifier checks. Support fields written by the model do not become externally verified evidence simply by satisfying the schema.

Table 6: **Role interfaces and deterministic checks.** The source companion retains the effective prompt constructors for these interfaces.

Role	Output and validation
Main	`reasoning` string and `state` with the four fields of Eq. 1.
Peer	`comment:null`, or target/critique/suggestion strings and nonempty supplied `evidence_ids`.
Gate	Typed `issue_kind`. Boolean `state_error`, `next_step_harm`, `already_addressed`. Selected comment IDs, evidence IDs, short explanation.
Integrator	State. One decision per selected comment, with correction text and source IDs. Retain/revoke review of active corrections. Correction summaries have 1 to 280 characters.
Verifier	Exactly `commit`, `verified_comment_ids`, `evidence_ids`, `reason`. Positive verdicts cover every accepted comment and accepted source. Negative verdicts have an empty list of verified comments.
QA final	Requested answer type. At most three typed candidates with source quotations. A selected ID or explicit unresolved selection. The selected candidate's type matches the requested type.
Math final	Nonempty answer of at most 4,096 characters, without JSON/code-fence packaging or truncation.

The effective prompts combine base instructions for each role with domain, additions for candidate history, type, and recovery. The local source companion preserves these constructors rather than replacing them with approximate prose prompts. Prompts for the main agent, integrator, and verifier identify history as hypotheses. The peer uses public evidence, and the early gate distinguishes unsupported claims from contradictions. The verifier compares original and candidate states, and the final prompt applies the domain contract. Format recovery names violated constraints and asks for a complete response. All prompts exclude hidden answers and hidden tests from their inputs.

A.4 Dataset Identity and Selection

The available MATH-500 source is `https://huggingface.co/datasets/HuggingFaceH4/MATH-500`, test split, revision `6e4ed1a2a79af7d8630a6b768ec859cb5af4d3be`. Its JSONL SHA-256 is `35dc41080a3680858b27fa7e0533d2d547825316fc5dafe5d316f4ccc5a06132`. MuSiQue uses `https://huggingface.co/datasets/dgslibisey/MuSiQue`, revision `c8f4f8c9465fb69d31a8eae894c3fd509c4ca321`, file `musique_ans_v1.0_dev.jsonl`, SHA-256 `15fa63794d18a94ce12411aca6e2327e65b6e83b0b1490efab3f1962e48abf3b`. HotpotQA's available selection uses `hotpot_dev_distractor_v1.json`, SHA-256 `4e9ecb5c8d3b719f624d66b60f8d56bf227f03914f5f0753d6fa1b359d7104ea`.

The initial MATH-500 and QA selection process uses the salt `abstract-baseline-20260919-v1` for ranking tasks. Ranking uses SHA-256 of the UTF-8 string `salt|dataset:source-id`. Stratum quotas and exclusions are applied before ordering. Selected and excluded IDs, source hashes, and component metadata are provided in the local companion's sanitized selection records. Ablation batches receive separate manifests and explicit exposure classifications. Frozen lists are not reconstructed by rerunning exclusion scans after further experiments.

OlymMATH uses the English EASY/HARD files at `RUC-AIBOX/OlymMATH` revision `2c6532ea2cf929ac1c421532af5951553eaee727`, selecting 50 tasks from each difficulty with salt `olymmath-english-four-arm-20260922-v1`. HumanEval uses all 164 original tasks; the compressed source SHA-256 is `b796127e635a67f93fb35c04f4cb03cf06f38c8072ee`

7cee8833d7bee06979ef. Generation receives public prompts and entry points, never hidden tests.

APPS uses the test split of https://huggingface.co/datasets/codeparrot/apps, revision 21e74ddf8de1a21436da12e3e653065c5213e9d1, with source SHA-256 5b003a65ac40feb47dd5eaec267a767a6fc435bdcfa68ff715fe869f948e760c. A fixed salted ranking selects 100 Interview-difficulty tasks (99 standard-input/output tasks and one function-call task), using salt apps-interview-formal-candidates-20260924-v1. Eight training-split development tasks are separate from the test sample. Generation receives the public problem, not reference solutions or evaluation test cases. Scoring runs the final program against all provided evaluation cases in isolated Python 3.12, with a six-second startup-inclusive limit per case, 512 MiB memory and a 1 MiB output limit. The native runner uses an APPS-derived comparator without implicit imports or upstream input rewriting; equivalence to the official harness is not claimed.

A.5 Artifact Availability

The manuscript has a local source companion containing controller and baseline source, effective prompt constructors, relevant tests, and sanitized available selection records. The source snapshot excludes credentials, raw model responses and benchmark answers. A separate numerical audit accompanies the manuscript. Validation as a turnkey reproduction package remains pending. Public anonymous hosting, complete environment locking, complete scoring environments are outstanding. No public release uniform resource locator (URL) is asserted here.

The public package is planned to include environment and dependency versions, effective role prompts and schemas, and dataset manifests, salts, selected/excluded IDs and licenses. It is also planned to include scoring, equivalence and sandbox code, controls and usage metadata for each request, and the paired bootstrap analysis. An offline rebuild check will verify that the selected IDs and effective prompt hashes match the archived manifests before release. Run provenance for Table 1 is documented in the accompanying Evaluation Notes. The numerical audit records source hashes and recomputes task means from archived scores.

B Completed Results

B.1 Token Accounting

Mean combined input and output usage in Table 1 is 23,492 tokens per task for MuSiQue CIPI and 12,345 for HotpotQA CIPI. Operational reservations are excluded when measured usage is reported. HotpotQA MP has a measured mean of 7,396.08 tokens per task.

B.2 Trace Provenance

The matched case in Section 4 uses task musique:4hop1__15607_59747_211319_557671. Episodes 00000 to 00003 of abstract_musique100_r1_20260920 contain EoT-Report, MP, MP+SR and CIPI in that order. CIPI records one peer comment, two subsequent events with peer silence, two wait decisions and one verified terminal commitment. Scores for each row and recorded usage provide the displayed values. This retrospective illustration does not meet a claim of case selection without knowledge of its outcome.

C Timing and Selection Ablations

C.1 Timing and Selection Ablation Records

Table 2 reports four arms: CIPI, CIPI-All, Always-All and Random-All. CIPI-All retains the conditional trigger but inspects all pending comments, providing an all-comment anchor for the timing controls. Always-All and Random-All change only the early-inspection policy relative to that anchor. All retain verification, rollback and terminal review.

Random-All uses the fraction of triggered inspections among eligible, nonempty-buffer development checkpoints with completed decisions. The frozen probabilities are $12/32 = 0.375$ for MuSiQue, $7/12$ for OlymMATH and $2/3 \approx 0.666667$ for HumanEval. Each batch freezes its task IDs and full-CIPI anchor before generation. Ablation trajectories are separate from the main comparison. The accompanying Evaluation Notes provide the scoring dispositions, usage coverage and execution history.

Table 7 reports the three prespecified contrasts for each dataset. Intervals use 10,000 paired task-level percentile bootstrap resamples with seed 20260924 and are unadjusted exploratory intervals. MuSiQue differences use unrounded scores; accuracy and pass-rate differences follow the percentages in Table 2.

Table 7: **Paired ablation contrasts.** Differences and 95% intervals are in percentage points of the dataset's primary score.

Dataset	Contrast	Difference	95% interval
MuSiQue	CIPI-All − Always-All	3.68	$[-3.07, 10.56]$
	CIPI-All − Random-All	−0.07	$[-7.40, 7.26]$
	CIPI − CIPI-All	2.06	$[-2.96, 7.24]$
OlymMATH	CIPI-All − Always-All	0.00	$[-6.00, 6.00]$
	CIPI-All − Random-All	0.00	$[-7.00, 7.00]$
	CIPI − CIPI-All	1.00	$[-5.00, 8.00]$
HumanEval	CIPI-All − Always-All	1.00	$[-3.00, 6.00]$
	CIPI-All − Random-All	1.00	$[-2.00, 5.00]$
	CIPI − CIPI-All	1.00	$[0.00, 3.00]$

D Six-Benchmark Evaluation Scope

D.1 Benchmarks and Study Boundary

The program across six benchmarks covers multi-hop QA, mathematics and executable code (Table 8). The main table comprises 100 tasks each for MATH-500, MuSiQue, HotpotQA, OlymMATH and APPS, and 164 for HumanEval: 664 tasks and 2,656 task–method outcomes. Batch identities and execution history are recorded in the accompanying Evaluation Notes. These sample sizes do not constitute a completed power plan.

The main comparison contains CIPI, MP, EoT-Report and MP+SR only, using the documented role settings. Its completed MATH-500, MuSiQue, HotpotQA, OlymMATH, APPS and HumanEval comparisons appear in Table 1. The four-arm timing/selection comparisons remain separate ablations outside the main comparison.

Table 8: **Six datasets in the main comparison.** Counts give the evaluated tasks, and Section D.4 describes precision and multiplicity. Appendix A.4 lists source identifiers and subset exclusions.

Dataset	Tasks	Split/selection and primary outcome
MuSiQue	100	MuSiQue-Ans v1.0 dev. 34 two-hop, 33 three-hop, 33 four-hop. Answer F1
HotpotQA	100	Distractor dev v1. Answer F1
MATH-500	100	Test. Level quotas 19/21/20/20/20. Correctness
OlymMATH	100	English EASY/HARD, 50 each. Correctness
APPS	100	Interview test subset. 99 standard-I/O, one function-call task. Task pass rate
HumanEval	164	Original tasks/tests. Task pass rate

MuSiQue and HotpotQA provide contexts from multiple documents (Trivedi et al., 2022; Yang et al., 2018). MATH-500 is the subset associated with Lightman et al. (2024), drawn from MATH (Hendrycks et al., 2021b). OlymMATH uses English problems with numerical answers from EASY/HARD (Sun et al., 2026), not the LEAN subset added in the formal publication. The

exact source revision is given in Appendix A.4. APPS and HumanEval use execution-based scoring (Hendrycks et al., 2021a; Chen et al., 2021). The code study uses the original tests, and any analysis with an augmented test suite is separately named.

Selection ranks eligible source IDs by a fixed salted hash and records exclusions before generation. The earlier MuSiQue amendment excludes prior full questions but allows shared decomposition components, including at higher hop counts. Such samples support holdout analyses at the full question level. Analyses of unseen components require a separately identified eligible subset. HumanEval has also been used previously in this project. Future samples distinguish previously inspected tasks from fresh tasks, and all subsets retain their exposure labels. None of these controls at the project level determines what a hosted model saw during training.

D.2 Comparators and Controls

Output-level message passing (MP). Following the topology of Liang et al. (2024), an affirmative solver proposes an answer, a negative solver responds, and a moderator follows both. This adaptation uses at most three rounds and structured JSON responses. In round 1 the affirmative solver has no opponent response. In subsequent rounds, it receives the negative solver's response from the preceding round. The negative solver receives the current affirmative response. Each solver also receives its own interaction history, while the moderator receives its prior rounds and both current responses. A nonempty trimmed `debate_answer` ends the run independently of the moderator's preference and supported-side fields. An empty answer requests another round of the debate. After three empty moderator answers, an extraction judge produces 1 to 16 candidates from the original responses in the first round. A selection judge receives those responses, the extracted candidates and the public task. Its nonempty answer is returned with a fallback flag. Usage from both judge calls is included in accounting. Exhausted schema recovery yields an invalid result, not an alternative answer selected by its reference score.

Exchange-of-Thought Report (EoT-Report, adapted). Three nodes A, B, C exchange messages from the previous round synchronously in a star (Yin et al., 2023). The center A receives both peripheral messages, and B and C receive only A's message. No node receives peer messages during the first round, round 1. Although requests are dispatched in A, B, C order, each reads a frozen snapshot from the previous round, never another node's output from the current round. Every node also receives its own answer/reasoning history.

For each node, confidence is the size of the largest group of pairwise equivalent answers divided by the valid history length. An answer is valid when the comparator accepts it against itself. Confidence accompanies the answer and reasoning in the next round's peer payload, and the prompt asks the recipient to consider it. The controller applies no numeric confidence weight or threshold. After each complete round, it enumerates groups whose answers agree pairwise by decreasing size, breaking ties lexicographically in A, B, C order. If the largest group has at least two members, the run returns its first member's answer. In particular, a $\{B, C\}$ agreement returns B, not the disagreeing center. A lack of agreement after six rounds returns A's answer with `round_cap_no_consensus`. The implementation has no separate rule that stops each node when its answer becomes stable. Exhausted schema recovery ends the run as invalid.

QA agreement uses equality of normalized strings, code uses normalization of line endings and outer whitespace, and mathematics requires the local `math_equivalence_v1` evaluator to accept both answer orderings. Pairwise agreement is not replaced by transitive closure. The agreement checks exclude reference answers and hidden tests. This bounded Report adaptation uses the same model across nodes and retains the star topology and messages containing confidence. The six-round cap, exact agreement/output rules, and role settings follow this implementation's specifications. These choices do not claim to reproduce every original EoT setting.

Message passing plus Self-Refine (MP+SR, adapted). This arm samples its own MP run rather than reusing the main-table MP response. An invalid MP result ends the arm immediately. Otherwise a critic receives the public task and current answer and emits `needs_revision` plus a reason (Madaan et al., 2023). A false request returns the current answer without refinement. A true request invokes one refiner with the task, answer and critique. The refiner emits reasoning and a replacement answer that becomes current. At most two such cycles run, with the answer returned

after the second revision without a third critique. No nested debate or separate acceptance verifier is used. Exhausted schema recovery yields an invalid result.

Ablation boundary. CIPI-All supplies an all-comment conditional anchor; Always-All and Random-All replace its early-inspection policy as defined in Table 2. These arms retain verification and the rest of the controller and are not main-table communication baselines.

D.3 Thinking, Budget, and Failure Controls

The main setting retains the thinking allocation specified for each role in Table 5, with requested temperature 0.6. Always/Random ablations match the role settings of their full-CIPI anchor.

The existing batches use the requested model identifier `deepseek-flash`. Request logs record returned identity, provider fingerprint, role settings, and code/prompt hashes. In thinking mode, the provider ignores the requested temperature setting.[1] The current evidence therefore concerns one hosted model configuration. A second model family would test generalization but is outside this comparison across six datasets and its four-arm ablation study.

All attempted requests count toward usage, including retries and rejected proposals. Let $\mathcal{O}$ and $\mathcal{U}$ denote calls with complete and missing usage. For role r_j, let m_j be the locally counted serialized input plus the recorded framing margin, and $o_{r_j}^{\max}$ its output limit. Define

$$T_{\text{in}} = \sum_{j \in \mathcal{O}} u_j^{\text{in}}, \quad T_{\text{out}} = \sum_{j \in \mathcal{O}} u_j^{\text{out}}, \quad T_{\text{reserved}} = \sum_{j \in \mathcal{U}} (m_j + o_{r_j}^{\max}). \tag{5}$$

The total accounted quantity for usage is $T_{\text{in}} + T_{\text{out}} + T_{\text{reserved}}$. Equation 5 separates measured tokens from operational reservations. Reasoning tokens already in completion usage are counted once. Cached input remains included in measured input usage. Token counts measure workload and do not represent monetary cost.

Some retry paths at their maximum length exceed the batch-specific admission ceiling, including the 768,000-token ceiling used in the initial batches. The ceiling can therefore prevent these paths from completing. Existing code suspends episodes that reach budget or input limits as operational halts. These operational halts remain in the denominator of planned tasks. An incomplete campaign therefore reports completed/invalid/suspended counts, not an accuracy for the full sample. Service outages remain suspended with the pair unresolved. Schema failures after allowed retries count as invalid, with zero primary score. Raising a ceiling defines a new batch and cannot silently replace the original configuration. Quality, actual usage, call counts and rates of reaching caps are reported together. No performance claim under equal token usage follows from the present comparison.

D.4 Outcomes and Statistical Plan

QA uses answer F1 as its primary outcome; exact match is additionally reported for the MuSiQue ablation. Mathematics uses answer correctness with unresolved equivalence checks listed separately. Code uses the task pass rate from final programs on the stated tests in isolation. The output schema is fixed before each batch. A supplementary extraction tolerant of formatting differences uses a common, reference-blind rule.

The planned main family contains 18 contrasts between CIPI and the communication baselines. These cover three baselines on each of the six benchmarks, with one primary outcome per dataset. Any confirmatory significance assessment must apply Holm correction across this full family at 0.05. Results from an incomplete comparison family remain descriptive. EM, cost, strata and Always/Random ablations are separately reported exploratory outcomes.

Main-comparison intervals use 10,000 paired percentile bootstrap draws, seed 20260919, over matched task scores. MuSiQue's additional analysis of connected components resamples entire groups sharing decomposition IDs. This accounts for dependence within the sample but not historical exposure. For a paired binary outcome with discordance q, the approximate 80%-power detectable difference is

$$\delta_{80} \simeq (z_{1-\alpha/2} + z_{0.8})\sqrt{q/n}. \tag{6}$$

[1] `https://api-docs.deepseek.com/guides/thinking_mode/`, checked 2026-09-20.

At $n = 100, q = 0.25$, Eq. 6 gives about 14.0 percentage points at $\alpha = .05$, or 19.2 at $.05/18$. These are sensitivity calculations, not F1 power estimates. The current sample sizes therefore do not establish power for small gains. Results remain descriptive, and a nonsignificant contrast is not evidence of equivalence.