ROUTE TEAMS, NOT AGENTS: TEAM-VALUE LEARNING FOR MULTI-AGENT REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Effective multi-agent reasoning requires selecting agents that work well together rather than simply assembling strong individuals. Complementarity, redundancy, and team-size effects make individual suitability an incomplete guide to collective performance. We introduce **TeamRoute**, which separates offline outcome learning from online constraint-aware routing. Its Experience Brain combines context-dependent role, pairwise, and cardinality factors in a latent score space and decodes them into quality and resource estimates. The Rational Scheduler uses skill- and capacity-aware optimal transport to generate candidates, then applies the frozen predictor to screen resource budgets and minimize predicted cost within a quality tolerance. This separation allows runtime budgets and preferences to change without retraining the predictor. Our analysis characterizes a limitation of additive latent scores and derives a conditional quality-loss bound involving prediction error, quality tolerance, and candidate restriction. Experiments across diverse reasoning benchmarks show improved average accuracy and favorable accuracy–budget trade-offs. Ablations and diagnostic experiments further support structured set-level prediction, selection among unseen agent combinations, and transport-guided candidate generation.

1 INTRODUCTION

Large language model (LLM)-based multi-agent systems improve reasoning by coordinating agents with different roles and capabilities. Through response exchange, task decomposition, and verification, collaborating agents can address problems that are difficult for an individual agent. Role-based workflows and multi-agent debate illustrate the potential of this approach (Hong et al., 2024; Du et al., 2024). However, effective collaboration requires more than assembling capable agents. The system must determine which agents should work together for the current context and available resources. Selecting agents solely for their individual competence can miss complementary contributions or introduce redundant work. This motivates our central premise, *route teams, not agents*. Figure 1 illustrates the distinction between individual scoring and team-level outcome prediction.

Existing approaches organize collaboration through structured workflows, optimized communication topologies, and adaptive routing. GPTSwarm (Zhuge et al., 2024) optimizes agent computations and connectivity, while G-Designer (Zhang et al., 2025a) constructs task-conditioned communication graphs. DyLAN (Liu et al., 2024) selects agents using contribution-based importance scores. Related work also studies expert coordination and agent initialization (Zhang et al., 2025c; Tian et al., 2025). These approaches demonstrate the importance of adapting collaboration to the task. Nevertheless, individual importance and communication suitability do not directly specify the collective quality and resource consumption of an alternative agent combination. Our focus is to predict these outcomes explicitly and use them to compare candidate sets under the current deployment conditions.

Two challenges motivate this design. First, **collective performance is context-dependent and non-additive**. An agent's contribution depends on its teammates and the current reasoning requirements. Two capable solution generators may produce overlapping arguments, whereas a verifier with weaker standalone performance may identify errors that neither generator detects. Activation

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.

size introduces further variation because additional agents can improve capability coverage but also increase redundancy and execution costs. Second, **the preferred combination depends on resource constraints and quality–cost preferences**. A useful set may become unavailable or exceed the current token or latency budget. These challenges call for separating the estimation of collective outcomes from the criterion used to select an executable set.

We introduce **TeamRoute**, a team-value-guided framework that decouples offline outcome learning from online agent routing. The **Experience Brain** learns from sampled agent-set executions to predict correctness, answer quality, latency, and token consumption. It combines context-dependent role contributions, pairwise synergies, and cardinality effects in a latent score space, then decodes these scores into outcome estimates. This structured parameterization captures composition-dependent relationships without treating individual contributions as sufficient statistics for collective performance. The **Rational Scheduler** maps the current context to skill demands and uses capacity-aware optimal transport to construct candidate sets. The frozen predictor then guides sequential budget screening, quality-tolerance filtering, and predicted-cost minimization.

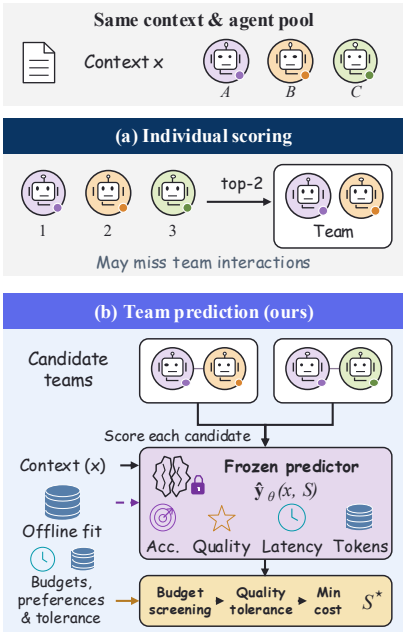


Figure 1: **From individual scoring to team prediction.** TeamRoute predicts collective outcomes to guide team selection.

The two components serve complementary purposes. Transport restricts the candidate search, while set-level prediction compares collective quality and resource requirements. Runtime constraints and preferences enter the decision layer, allowing the same learned predictor to be reused without parameter updates. Our theoretical analysis establishes a representation limitation of additive latent scores and provides a conditional quality-loss bound for the selection rule. Experiments on nine benchmarks show improved average accuracy and higher accuracy under the evaluated token and latency budgets. Component ablations, unseen-combination evaluation, and candidate coverage analysis examine the mechanisms behind these results. Our contributions are as follows:

- **Structured collective-outcome learning.** We develop the Experience Brain to model agent-set outcomes through context-dependent role, pairwise, and cardinality factors in a latent score space.
- **Constraint-aware agent selection.** We combine skill- and capacity-aware OT candidate generation with budget screening, quality-tolerance filtering, and cost minimization using a frozen outcome predictor.
- **Theoretical and empirical analysis.** We characterize an additive-score limitation and derive a conditional selection-quality bound. Experiments assess overall performance, component effects, unseen-combination selection, candidate quality, and accuracy under different resource budgets.

2 Related Work

2.1 LLM-Based Multi-Agent Collaboration

LLM-based multi-agent systems organize collaboration through response exchange, role specialization, and aggregation. AutoGen (Wu et al., 2023) provides programmable multi-agent conversations, while MoA (Wang et al., 2025) aggregates responses through a layered architecture. To automate collaboration design, GPTSwarm (Zhuge et al., 2024) optimizes agent computations and connectivity, G-Designer (Zhang et al., 2025a) learns task-conditioned communication graphs, and ARG-Designer (Li et al., 2026) jointly generates agent roles and communication structures.

AFlow (Zhang et al., 2025b) instead searches over executable workflows using execution feedback. These approaches emphasize how collaboration is organized. TeamRoute focuses on evaluating alternative agent sets under a fixed execution protocol, predicting their collective outcomes rather than introducing a new communication topology.

2.2 Adaptive Model and Agent Routing

Adaptive routing allocates computation according to the query or intermediate reasoning state. RouteLLM (Ong et al., 2025) learns model selection from preference data to balance response quality and cost. SpecReason (Pan et al., 2025) combines small-model reasoning with larger-model verification, while STEER (Lee et al., 2026) uses model confidence for step-level routing. For multi-agent systems, MasRouter (Yue et al., 2025) coordinates collaboration modes, role allocation, and model assignment. DMoA (Wu et al., 2026) dynamically activates agents through recurrent routing and entropy-based test-time adaptation. We study the same context-aware agent-routing setting as DMoA but adopt a different decision mechanism: a frozen predictor evaluates complete candidate sets, and the scheduler applies budget screening, quality filtering, and cost minimization. This separates learned execution outcomes from changing runtime constraints and preferences.

3 Problem Formulation

We consider an agent pool $\mathcal{A} = \{a_1, \ldots, a_N\}$, where $a_i = (\mathcal{M}_i, \mathcal{P}_i, \mathcal{T}_i)$ comprises a language model, a role profile, and available tools. Given query u, let $\mathcal{H}_t$ denote the shared context at reasoning step t, with $\mathcal{H}_1 = u$. A fixed encoder produces $x_t = \phi(\mathcal{H}_t) \in \mathbb{R}^d$. The router activates a set of agent indices $S_t \subseteq [N]$, where $[N] = \{1, \ldots, N\}$ and $1 \le |S_t| \le K \le N$.

Selected agents execute in parallel using the shared context, their role instructions, and their available tools. Their responses construct the context for a subsequent step:

$$\begin{aligned} o_{t,i} &= a_i(\mathcal{H}_t), \qquad i \in S_t, \\ \mathcal{H}_{t+1} &= \mathrm{Concat}\big(u, \{(i, o_{t,i}) : i \in S_t\}\big). \end{aligned} \tag{1}$$

To evaluate a selected set, a fixed task-specific protocol Π produces an answer through synthesis or answer selection:

$$\widehat{a}_t = \mathrm{Readout}_{\Pi}\big(\mathcal{H}_t, \{(i, o_{t,i}) : i \in S_t\}\big). \tag{2}$$

Correctness and quality labels evaluate this answer, and resource measurements cover its associated execution under Π. The readout defines the output being scored, whereas Eq. (1) defines information passed to a subsequent reasoning step. Agent configurations and the task protocol are held fixed when comparing alternative sets. The routing objective is to select agents suited to the current context while balancing collective answer quality and execution cost.

4 Method

TeamRoute separates offline learning of collective outcomes from online agent routing. The **Experience Brain** predicts the quality and execution costs of an agent set. The **Rational Scheduler** uses skill- and capacity-aware optimal transport to construct candidates, then applies the frozen predictor through budget screening, quality-tolerance filtering, and cost minimization (Figure 2).

4.1 Experience Brain: Learning Collective Outcomes

Execution-level supervision. For each training context $\mathcal{H}$, represented by $x = \phi(\mathcal{H})$, we execute sampled agent sets under a fixed answer-production and evaluation protocol. Each observation evaluates the selected agents' responses through the task-specific synthesis or answer-selection procedure. This yields $\mathcal{D} = \{(x, S, \mathbf{y}^{\mathrm{obs}})\}$, with

$$\mathbf{y}(x, S) = \mathbb{E}[\mathbf{y}^{\mathrm{obs}} \mid x, S] = \big[y^{\mathrm{acc}}, y^{\mathrm{qual}}, y^{\mathrm{lat}}, y^{\mathrm{tok}}\big]^{\top}. \tag{3}$$

The components describe correctness probability, normalized answer quality, latency, and token consumption. Quality labels and costs refer to the same execution protocol; offline grading is excluded from execution costs. They do not implicitly represent the return of an unspecified future routing policy.

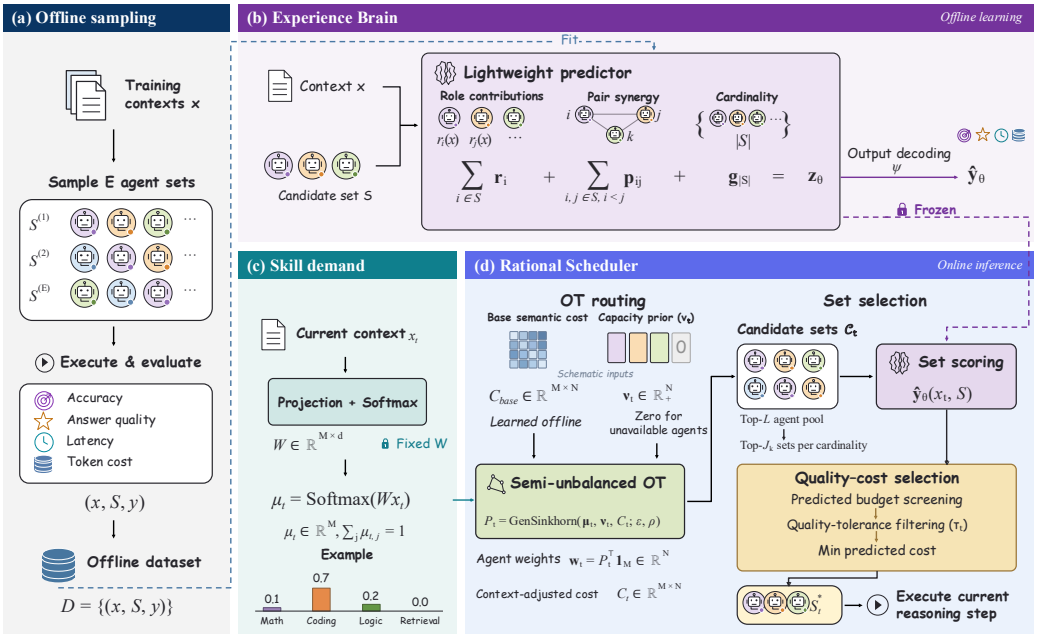


Figure 2: **Overview of TeamRoute.** Offline executions of sampled agent sets provide supervision for the Experience Brain. During online routing, skill-demand estimation and capacity-aware transport guide candidate generation. The frozen predictor then supports budget screening, quality-tolerance filtering, and predicted-cost minimization before the selected agents execute.

Structured prediction. We decompose *pre-link scores*, rather than decoded probabilities and costs, into role, pairwise, and cardinality terms:

$$\mathbf{z}_\theta(x, S) = \sum_{i \in S} \mathbf{r}_i(x) + \sum_{\substack{i,j \in S \\ i<j}} \mathbf{p}_{ij}(x) + \mathbf{g}_{|S|}(x), \qquad \hat{\mathbf{y}}_\theta(x, S) = \psi(\mathbf{z}_\theta(x, S)). \tag{4}$$

All factors are vector-valued outputs of indexed heads sharing a feature network over x. Role terms capture context-dependent agent contributions, pair terms capture complementarity or redundancy, and cardinality terms capture activation-size effects. Unordered pair indexing makes the prediction invariant to the enumeration order of S. The coordinate-wise decoder ψ uses sigmoid links for correctness and quality, and nonnegative log-cost decoding for resource estimates.

Training combines correctness cross-entropy, quality regression, and regression on log-transformed execution costs:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{acc}} + \mathcal{L}_{\text{qual}} + \lambda_{\text{cost}} \mathcal{L}_{\text{cost}}. \tag{5}$$

Here, $\lambda_{\text{cost}} \geq 0$ weights resource prediction. Appendix A.1 specifies the heads, links, and losses. The predictor is frozen after training, separating outcome estimation from runtime preferences. In particular, decoded log-cost estimates are not assumed to be unbiased estimates of expected resource consumption.

4.2 Rational Scheduler: Transport-Guided Candidates

Skill demand and semantic costs. An offline-fitted projection maps the current context to demands over M skills:

$$\boldsymbol{\mu}_t = \text{Softmax}(W x_t), \qquad W \in \mathbb{R}^{M \times d}. \tag{6}$$

For normalized skill embeddings $\mathbf{e}_m$ and offline-adapted agent embeddings $\widetilde{\mathbf{e}}_i$, define

$$C_{\text{base},mi} = 1 - \mathbf{e}_m^\top \widetilde{\mathbf{e}}_i, \qquad C_t = \text{diag}(\mathbf{1}_M + \boldsymbol{\mu}_t) C_{\text{base}}. \tag{7}$$

Highly demanded skills receive both more transport mass and a larger mismatch penalty. The projection and semantic embeddings are constructed from training data only and remain frozen; their construction is described in Appendix A.2.

Semi-unbalanced transport. Let $b_{t,i} \geq 0$ be agent i's available execution capacity and $\mathcal{I}_t = \{i : b_{t,i} > 0\}$. For nonempty $\mathcal{I}_t$, normalize positive capacities as $\nu_{t,i} = b_{t,i} / \sum_{j \in \mathcal{I}_t} b_{t,j}$ on $\mathcal{I}_t$, with zero mass elsewhere. We solve

$$\begin{aligned} P_t^\star \in \arg\min_{P \geq 0} \quad & \langle C_t, P \rangle + \varepsilon \sum_{m=1}^{M} \sum_{i \in \mathcal{I}_t} P_{mi}(\log P_{mi} - 1) \\ & + \rho \, \mathrm{KL}_{\mathcal{I}_t}(P^\top \mathbf{1}_M \| \boldsymbol{\nu}_t) \\ \text{subject to} \quad & P\mathbf{1}_N = \boldsymbol{\mu}_t, \qquad P_{mi} = 0 \quad (i \notin \mathcal{I}_t), \end{aligned} \tag{8}$$

where $\varepsilon, \rho > 0$ and $\mathrm{KL}_{\mathcal{I}_t}(\mathbf{a} \| \mathbf{b}) = \sum_{i \in \mathcal{I}_t} [a_i \log(a_i / b_i) - a_i + b_i]$. The skill marginal is exact, while agent capacities provide a soft prior rather than hard column bounds. Generalized Sinkhorn scaling yields a numerical plan P_t; updates are given in Appendix A.3.

Discrete candidate construction. Column masses define agent weights $\mathbf{w}_t = P_t^\top \mathbf{1}_M$. We form a candidate agent pool

$$\mathcal{R}_t = \mathrm{TopL}_{i \in \mathcal{I}_t}(w_{t,i}) \cup \mathcal{R}_{\mathrm{req}}, \tag{9}$$

where $\mathcal{R}_{\mathrm{req}}$ contains mandatory agent indices. Routing is deferred if no agents are available or a mandatory agent is unavailable. For admissible cardinalities $\mathcal{K} \subseteq \{1, \ldots, K\}$, let $\mathcal{G}_{t,k} = \{S \subseteq \mathcal{R}_t : |S| = k,\ \mathcal{R}_{\mathrm{req}} \subseteq S\}$. Retaining up to J_k sets per cardinality gives

$$s_{\mathrm{OT},t}(S) = \frac{1}{|S|} \sum_{i \in S} \log\big(\max(w_{t,i}, \eta)\big), \tag{10}$$

$$\mathcal{C}_t = \bigcup_{k \in \mathcal{K}} \mathrm{TopJ}_{\mathrm{k}\, S \in \mathcal{G}_{t,k}} \, s_{\mathrm{OT},t}(S), \tag{11}$$

where $\eta > 0$ is a numerical safeguard. Candidate-ranking ties use deterministic agent-index ordering. This bounds predictor evaluations by $|\mathcal{C}_t| \leq \sum_k J_k$ without executing the candidates online. Transport determines the shortlist, whereas the Experience Brain compares collective outcomes.

4.3 Sequential Quality–Cost Selection

Let $\mathcal{E}_t$ contain all available sets satisfying the cardinality and mandatory-agent constraints. The true expected-budget feasible family is

$$\mathcal{F}_t = \left\{ S \in \mathcal{E}_t : y^{\mathrm{lat}}(x_t, S) \leq B_t^{\mathrm{lat}},\ y^{\mathrm{tok}}(x_t, S) \leq B_t^{\mathrm{tok}} \right\}. \tag{12}$$

Online screening uses predicted costs instead, giving $\mathcal{V}_t = \mathcal{C}_t \cap \widehat{\mathcal{F}}_t$. An empty collection leads to deferral.

For nonnegative quality weights $\boldsymbol{\alpha}_t$ with $\|\boldsymbol{\alpha}_t\|_1 = 1$ and cost weights $\boldsymbol{\lambda}_t \geq 0$, define

$$\begin{aligned} Q_t(S) &= \alpha_{t,\mathrm{acc}} y^{\mathrm{acc}}(x_t, S) + \alpha_{t,\mathrm{qual}} y^{\mathrm{qual}}(x_t, S), \\ c_t(S) &= \lambda_{t,\mathrm{lat}} \frac{y^{\mathrm{lat}}(x_t, S)}{s_{\mathrm{lat}}} + \lambda_{t,\mathrm{tok}} \frac{y^{\mathrm{tok}}(x_t, S)}{s_{\mathrm{tok}}}. \end{aligned} \tag{13}$$

Here, $s_{\mathrm{lat}}, s_{\mathrm{tok}} > 0$ are fixed training-derived reference scales. Predicted quantities $\widehat{Q}_t$ and $\widehat{c}_t$ use decoded outcomes $\widehat{\mathbf{y}}_\theta$. For nonempty $\mathcal{V}_t$, the scheduler retains near-best-quality candidates and then minimizes cost:

$$\mathcal{V}_t(\tau_t) = \left\{ S \in \mathcal{V}_t : \widehat{Q}_t(S) \geq \max_{S' \in \mathcal{V}_t} \widehat{Q}_t(S') - \tau_t \right\}, \tag{14}$$

$$S_t^\star \in \arg\min_{S \in \mathcal{V}_t(\tau_t)} \widehat{c}_t(S), \tag{15}$$

where $\tau_t \geq 0$. Ties favor higher predicted quality, followed by deterministic agent-index ordering. Zero tolerance prioritizes maximum predicted quality; a positive tolerance permits lower-cost alternatives.

Before dispatch, execution slots are reserved and routing is retried if availability has changed. The selected agents execute in parallel under the fixed protocol, with $S_t = S_t^\star$ and the context update defined in Eq. (1). Slot reservation controls execution capacity, but predicted cost screening alone does not ensure per-execution budget compliance.

4.4 Theoretical Analysis

Limitations of additive scores. Because prediction includes nonlinear decoding, we analyze additivity in the pre-link score space. Fix context x, a background set B, and distinct indices i, j, ℓ, h outside B. Write $S_{ab} = B \cup \{a, b\}$ and $\mathcal{Q}_B = \{S_{ij}, S_{\ell h}, S_{ih}, S_{\ell j}\}$, assuming all these sets are in the prediction domain. For correctness probabilities in $(0, 1)$, define $f_x(S) = \mathrm{logit}(y^{\mathrm{acc}}(x, S))$ and

$$\Xi_x = f_x(S_{ij}) + f_x(S_{\ell h}) - f_x(S_{ih}) - f_x(S_{\ell j}). \tag{16}$$

Proposition 1 (Additive-score limitation)**.** *Every score model* $A_x(S) = \sum_{a \in S} f_a(x) + h_{|S|}(x)$ *satisfies*

$$\max_{S \in \mathcal{Q}_B} |f_x(S) - A_x(S)| \geq |\Xi_x|/4. \tag{17}$$

Pair factors permit nonzero score contrasts that individual and cardinality terms cannot express. This is a restriction on additive scores, not on arbitrary permutation-invariant networks or nonlinear output probabilities.

Selection quality. The next result first compares against the screened candidates actually used by the scheduler, then states the additional condition needed for comparison with a truly feasible oracle.

Theorem 1 (Decision error decomposition)**.** *Suppose* $\mathcal{V}_t \neq \varnothing$ *and*

$$\max_{S \in \mathcal{V}_t} |\widehat{Q}_t(S) - Q_t(S)| \leq \epsilon_t. \tag{18}$$

Then

$$0 \leq \max_{S \in \mathcal{V}_t} Q_t(S) - Q_t(S_t^\star) \leq 2\epsilon_t + \tau_t. \tag{19}$$

If additionally

$$\mathcal{V}_t = \mathcal{C}_t \cap \mathcal{F}_t \neq \varnothing, \tag{20}$$

define

$$S_t^{\mathrm{oracle}} \in \arg\max_{S \in \mathcal{F}_t} Q_t(S), \tag{21}$$

$$\Delta_t^{\mathrm{cand}} = Q_t(S_t^{\mathrm{oracle}}) - \max_{S \in \mathcal{C}_t \cap \mathcal{F}_t} Q_t(S). \tag{22}$$

Then

$$0 \leq Q_t(S_t^{\mathrm{oracle}}) - Q_t(S_t^\star) \leq 2\epsilon_t + \tau_t + \Delta_t^{\mathrm{cand}}. \tag{23}$$

The bound separates scoring error, quality tolerance, and candidate restriction. It is a conditional one-step result and does not imply exact budget screening, optimal true cost, or multi-step guarantees. Proofs are given in Appendices B.1 and B.2.

5 Experiments

5.1 Experimental Setup

5.1.1 Datasets.

We evaluate our method on 9 representative benchmarks spanning diverse reasoning settings. Specifically, we use MMLU (Hendrycks et al., 2020) to evaluate broad-domain reasoning; GSM8K (Cobbe et al., 2021), MultiArith (Roy & Roth, 2015), SVAMP (Patel et al., 2021), and AQuA (Ling et al., 2017) for mathematical reasoning; HumanEval (Chen et al., 2021) and DS-1000 (Lai et al., 2023) for coding ability; HotpotQA (Yang et al., 2018) for multi-hop question answering; and DDX-Plus (Fansi Tchango et al., 2022) for medical diagnosis reasoning.

5.1.2 Baselines.

We compare our method against a broad spectrum of multi-agent systems, including static methods such as CoT (Wei et al., 2022), Self-Consistency (Wang et al., 2023), Chain, Tree, Star, Complete Graph, Random Graph, AutoGen (Wu et al., 2023), MoA (Wang et al., 2025), and LLM-Debate (Du et al., 2024); spatially adaptive methods including GPTSwarm (Zhuge et al., 2024), G-Designer (Zhang et al., 2025a), ARG-Designer (Li et al., 2026), and SafeSieve (Zhang et al., 2025d); and temporally adaptive methods such as AFlow (Zhang et al., 2025b), SpecReason (Pan et al., 2025), STEER (Lee et al., 2026), and DMoA (Wu et al., 2026).

Table 1: Accuracy (%) across nine benchmarks. Red and blue indicate the highest and second-highest distinct displayed scores in each column, respectively. Subscripts indicate percentage-point differences from Vanilla. Avg. is the unweighted arithmetic mean of the nine displayed scores.

Methods	MMLU	GSM8K	MultiArith	SVAMP	AQuA	HumanEval	DS-1000	HotpotQA	DDXPlus	Avg.
Vanilla	80.47	87.15	93.40	87.25	69.27	73.28	38.40	82.35	56.40	74.22
Single-Agent Systems										
CoT	81.85$_{\uparrow 1.38}$	87.35$_{\uparrow 0.20}$	93.60$_{\uparrow 0.20}$	87.90$_{\uparrow 0.65}$	72.40$_{\uparrow 3.13}$	74.05$_{\uparrow 0.77}$	42.35$_{\uparrow 3.95}$	81.04$_{\downarrow 1.31}$	64.22$_{\uparrow 7.82}$	76.08
SC	82.95$_{\uparrow 2.48}$	87.66$_{\uparrow 0.51}$	94.40$_{\uparrow 1.00}$	87.40$_{\uparrow 0.15}$	71.85$_{\uparrow 2.58}$	75.82$_{\uparrow 2.54}$	37.66$_{\downarrow 0.74}$	82.24$_{\downarrow 0.11}$	66.43$_{\uparrow 10.03}$	76.27
Multi-Agent Systems										
Chain	82.40$_{\uparrow 1.93}$	87.23$_{\uparrow 0.08}$	92.88$_{\downarrow 0.52}$	87.16$_{\downarrow 0.09}$	69.85$_{\uparrow 0.58}$	73.40$_{\uparrow 0.12}$	38.25$_{\downarrow 0.15}$	80.77$_{\downarrow 1.58}$	65.28$_{\uparrow 8.88}$	75.25
Tree	81.55$_{\uparrow 1.08}$	86.35$_{\downarrow 0.80}$	92.95$_{\downarrow 0.45}$	88.25$_{\uparrow 1.00}$	71.44$_{\uparrow 2.17}$	75.33$_{\uparrow 2.05}$	43.50$_{\uparrow 5.10}$	81.50$_{\downarrow 0.85}$	69.25$_{\uparrow 12.85}$	76.68
Star	79.64$_{\downarrow 0.83}$	85.25$_{\downarrow 1.90}$	93.25$_{\downarrow 0.15}$	87.60$_{\uparrow 0.35}$	68.50$_{\downarrow 0.77}$	76.28$_{\uparrow 3.00}$	41.82$_{\uparrow 3.42}$	83.52$_{\uparrow 1.17}$	68.25$_{\uparrow 11.85}$	76.01
Complete Graph	82.75$_{\uparrow 2.28}$	88.05$_{\uparrow 0.90}$	94.73$_{\uparrow 1.33}$	88.20$_{\uparrow 0.95}$	70.44$_{\uparrow 1.17}$	84.25$_{\uparrow 10.97}$	45.05$_{\uparrow 6.65}$	83.35$_{\uparrow 1.00}$	70.35$_{\uparrow 13.95}$	78.57
Random Graph	83.20$_{\uparrow 2.73}$	88.25$_{\uparrow 1.10}$	94.80$_{\uparrow 1.40}$	87.10$_{\downarrow 0.15}$	70.16$_{\uparrow 0.89}$	84.14$_{\uparrow 10.86}$	47.48$_{\uparrow 9.08}$	84.28$_{\uparrow 1.93}$	70.44$_{\uparrow 14.04}$	78.87
AutoGen	83.25$_{\uparrow 2.78}$	89.40$_{\uparrow 2.25}$	95.05$_{\uparrow 1.65}$	88.15$_{\uparrow 0.90}$	71.50$_{\uparrow 2.23}$	83.50$_{\uparrow 10.22}$	48.55$_{\uparrow 10.15}$	83.25$_{\uparrow 0.90}$	71.83$_{\uparrow 15.43}$	79.39
LLM-Debate	83.28$_{\uparrow 2.81}$	89.95$_{\uparrow 2.80}$	95.45$_{\uparrow 2.05}$	88.30$_{\uparrow 1.05}$	73.52$_{\uparrow 4.25}$	85.24$_{\uparrow 11.96}$	49.26$_{\uparrow 10.86}$	84.55$_{\uparrow 2.20}$	71.33$_{\uparrow 14.93}$	80.10
MoA	82.65$_{\uparrow 2.18}$	88.52$_{\uparrow 1.37}$	94.50$_{\uparrow 1.10}$	88.24$_{\uparrow 0.99}$	67.35$_{\downarrow 1.92}$	82.48$_{\uparrow 9.20}$	39.75$_{\uparrow 1.35}$	84.02$_{\uparrow 1.67}$	71.85$_{\uparrow 15.45}$	77.71
DMoA	91.35$_{\uparrow 10.88}$	**98.87**$_{\uparrow 11.72}$	99.15$_{\uparrow 5.75}$	94.76$_{\uparrow 7.51}$	86.60$_{\uparrow 17.33}$	95.62$_{\uparrow 22.34}$	64.34$_{\uparrow 25.94}$	90.38$_{\uparrow 8.03}$	83.37$_{\uparrow 26.97}$	89.38
Adaptive Routing and Orchestration										
GPTSwarm	83.80$_{\uparrow 3.33}$	90.20$_{\uparrow 3.05}$	96.28$_{\uparrow 2.88}$	87.28$_{\uparrow 0.03}$	75.44$_{\uparrow 6.17}$	87.14$_{\uparrow 13.86}$	52.35$_{\uparrow 13.95}$	85.40$_{\uparrow 3.05}$	75.57$_{\uparrow 19.17}$	81.50
G-Designer	84.63$_{\uparrow 4.16}$	93.35$_{\uparrow 6.20}$	97.60$_{\uparrow 4.20}$	89.52$_{\uparrow 2.27}$	76.30$_{\uparrow 7.03}$	89.40$_{\uparrow 16.12}$	54.82$_{\uparrow 16.42}$	87.66$_{\uparrow 5.31}$	76.42$_{\uparrow 20.02}$	83.30
ARG-Designer	85.04$_{\uparrow 4.57}$	94.40$_{\uparrow 7.25}$	97.35$_{\uparrow 3.95}$	88.95$_{\uparrow 1.70}$	79.48$_{\uparrow 10.21}$	88.25$_{\uparrow 14.97}$	55.60$_{\uparrow 17.20}$	86.45$_{\uparrow 4.10}$	76.03$_{\uparrow 19.63}$	83.51
SafeSieve	84.65$_{\uparrow 4.18}$	93.20$_{\uparrow 6.05}$	97.80$_{\uparrow 4.40}$	90.10$_{\uparrow 2.85}$	80.40$_{\uparrow 11.13}$	90.15$_{\uparrow 16.87}$	53.73$_{\uparrow 15.33}$	87.22$_{\uparrow 4.87}$	77.94$_{\uparrow 21.54}$	83.91
AFlow	83.22$_{\uparrow 2.75}$	89.50$_{\uparrow 2.35}$	94.28$_{\uparrow 0.88}$	86.48$_{\downarrow 0.77}$	78.20$_{\uparrow 8.93}$	89.25$_{\uparrow 15.97}$	46.82$_{\uparrow 8.42}$	90.04$_{\uparrow 7.69}$	74.25$_{\uparrow 17.85}$	81.34
SpecReason	84.20$_{\uparrow 3.73}$	91.40$_{\uparrow 4.25}$	95.40$_{\uparrow 2.00}$	87.62$_{\uparrow 0.37}$	81.36$_{\uparrow 12.09}$	91.22$_{\uparrow 17.94}$	50.08$_{\uparrow 11.68}$	88.29$_{\uparrow 5.94}$	76.58$_{\uparrow 20.18}$	82.91
STEER	84.40$_{\uparrow 3.93}$	93.50$_{\uparrow 6.35}$	95.82$_{\uparrow 2.42}$	87.40$_{\uparrow 0.15}$	82.66$_{\uparrow 13.39}$	89.02$_{\uparrow 15.74}$	51.34$_{\uparrow 12.94}$	89.35$_{\uparrow 7.00}$	75.20$_{\uparrow 18.80}$	83.19
TeamRoute	**91.81**$_{\uparrow 11.34}$	97.40$_{\uparrow 10.25}$	**99.80**$_{\uparrow 6.40}$	**97.11**$_{\uparrow 9.86}$	**88.98**$_{\uparrow 19.71}$	**97.37**$_{\uparrow 24.09}$	**76.89**$_{\uparrow 38.49}$	**90.67**$_{\uparrow 8.32}$	**83.67**$_{\uparrow 27.27}$	**91.52**

5.1.3 Implementation Details

TeamRoute uses `gpt-oss-120b` for its task-specific agents. We encode contexts with a frozen `all-MiniLM-L6-v2` model, using embedding dimension $d = 384$. The Experience Brain uses a shared feature network and indexed role, pairwise, and cardinality heads. Their latent scores are decoded into correctness, quality, latency, and token estimates. During inference, a frozen skill-demand projection and semi-unbalanced OT generate candidate sets. The frozen Brain then supports budget screening, quality-tolerance filtering, and cost minimization. Selected agents execute in parallel, with answer production following the fixed task protocol.

5.2 Experimental Results

5.2.1 Main Results

Table 1 compares TeamRoute with single-agent, multi-agent, and adaptive routing baselines. TeamRoute achieves the highest average accuracy and leads on all evaluated benchmarks except GSM8K, where DMoA performs better. The improvements span different reasoning domains rather than being restricted to a single task family.

DMoA is the strongest competing baseline across the table. Relative to DMoA, TeamRoute shows its largest advantage on DS-1000, alongside improvements on AQuA, SVAMP, and HumanEval. The gains on MMLU, MultiArith, HotpotQA, and DDXPlus are more modest, indicating task-dependent benefits. Several fixed-topology configurations underperform Vanilla on individual benchmarks, showing that collaboration alone does not ensure better accuracy. TeamRoute also outperforms the other reported adaptive routing and orchestration baselines across the evaluated tasks. The following analyses examine the contributions of set-level prediction and candidate generation to this overall performance.

5.3 Ablation Study

Table 2 examines team-level prediction and OT-guided candidate generation. The full model achieves the best or joint-best accuracy across the evaluated benchmarks.

Table 2: Ablation accuracy (%).

Methods	HumanEval	AQuA	MultiArith
TeamRoute	97.37	88.98	99.80
w/o Pair	96.49	76.38	99.60
w/o Cardinality	90.35	80.71	99.60
w/ DeepSets	93.86	85.83	99.60
w/ Best Fixed	96.49	81.89	99.20
w/o Skill	97.37	74.80	99.40
w/ Softmax	93.86	85.43	99.60
w/ Random	93.86	84.25	97.60

Team-level modeling and selection. Removing pairwise interactions markedly reduces accuracy on AQuA, while removing cardinality modeling degrades both HumanEval and AQuA. These results support explicit interaction and activation-size factors in the latent outcome scores. The full predictor also outperforms DeepSets, favoring the structured parameterization in this setting. Its advantage over Best Fixed supports context-dependent selection rather than reusing one agent combination.

OT-guided candidate generation. Softmax routing and random proposals reduce accuracy relative to the full model across all evaluated tasks. Removing skill conditioning leaves HumanEval unchanged but substantially degrades AQuA, indicating task-dependent benefits from explicit skill demand. On MultiArith, random proposals cause a larger decline than the individual modeling ablations. These comparisons support guided candidate construction without implying that every component contributes equally on every task.

5.4 Generalization to Unseen Agent Combinations

We remove complete agent combinations from training and validation while retaining coverage of their members, pairs, and activation sizes. All predictors are retrained on the filtered data and select from the same held-out candidate family, independently of OT. As shown in Table 3, Full Brain outperforms w/o Pair and DeepSets on all evaluated benchmarks, with the clearest advantage on HumanEval. These results support reusing structured factors to guide selection among unobserved combinations of known agents. They do not test generalization to new agent identities. The holdout protocol is described in Appendix C.7.

Figure 3: Accuracy (%) on unseen agent combinations.

Predictor	HumanEval	AQuA	MultiArith
w/o Pair	89.47	88.58	98.40
DeepSets	90.35	88.98	99.20
Full Brain	92.98	89.76	99.80

5.5 Candidate Coverage and Selection Quality

We compare Random, Softmax, and OT using the same frozen Brain and matched candidate budgets. This diagnostic uses stochastic proposals in place of the main scheduler's deterministic candidate truncation. Coverage measures whether at least one candidate answers correctly, while final accuracy evaluates the selected candidate. Table 4 shows that OT achieves the highest coverage and final accuracy on all evaluated benchmarks. The improvement in final accuracy is particularly visible on HumanEval despite relatively small coverage differences. OT also yields the smallest coverage–accuracy gap, indicating a more effective interaction between its proposals and the fixed selector. Final accuracy reaches coverage on MultiArith, whereas selection gaps remain on HumanEval and AQuA. These

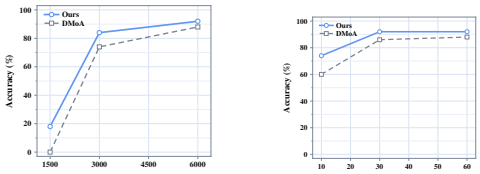
Figure 4: Candidate coverage / accuracy (%).

Generator	HumanEval	AQuA	MultiArith
Random	99.12 / 86.84	93.31 / 88.19	99.20 / 98.40
Softmax	99.12 / 88.60	93.70 / 88.19	99.60 / 99.20
OT	100.00 / 92.11	94.09 / 89.37	99.80 / 99.80

are empirical candidate diagnostics rather than direct measurements of the feasible-oracle gap in the theoretical bound. The protocol is specified in Appendix C.8.

5.6 Execution-Cost Analysis

Figure 5 compares TeamRoute and DMoA under varying token and latency budgets. TeamRoute achieves higher accuracy at every evaluated setting, with larger advantages under tighter constraints. Both methods deteriorate at the smallest token budget, showing that severe resource restrictions remain challenging. The gaps narrow as budgets increase but remain positive. In the latency sweep, TeamRoute reaches its observed accuracy plateau before the largest budget, whereas DMoA continues to improve. These results support a favorable accuracy–budget trade-off within the evaluated settings. The budget configurations are given in Appendix C.6.



(a) Token budget (b) Latency budget (s)

Figure 5: Accuracy under token- and latency-budget sweeps, with the other resource budget held fixed.

5.7 Scalability

Figure 6 varies the configuration (N, K), where N denotes the number of agents in the pool and K the maximum number that can be activated in a routing decision. HumanEval improves as these configurations scale up, while AQuA benefits more from the larger settings. MultiArith reaches a plateau earlier and maintains its performance. Thus, expanding the agent pool and activation limit yields task-dependent gains rather than uniform improvements. Because N and K vary jointly, the experiment characterizes their combined effect, not their separate contributions. It evaluates accuracy across system configurations rather than computational scaling or zero-shot adaptation to new agents. The evaluated configurations are listed in Appendix C.9.

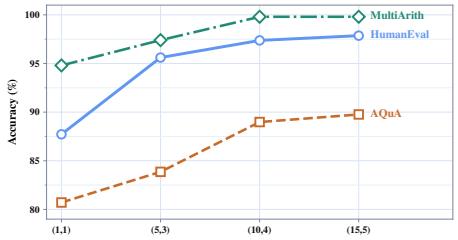


Figure 6: Accuracy across scaling configurations on HumanEval, AQuA, and MultiArith.

Illustrative role configurations are provided in Appendix D.

6 Conclusion

We presented TeamRoute, which separates offline learning of collective outcomes from online agent selection. The Experience Brain combines role, pairwise, and cardinality factors in a latent score space, while the Rational Scheduler uses skill-aware OT to generate candidates and sequentially applies budget screening, quality filtering, and cost minimization. Our analysis characterizes a limitation of additive latent scores and provides conditional quality bounds for the selection rule. Experiments show improved average accuracy and favorable accuracy–budget trade-offs across the evaluated settings. Ablations and diagnostic experiments further support structured outcome prediction, selection among unseen combinations of known agents, and transport-guided proposals. Future work will examine previously unseen agent identities and more reliable resource prediction under changing execution conditions.

AI Use Statement

Generative AI tools assisted with manuscript drafting and editing, literature search, figure preparation, and result interpretation. The authors take responsibility for the final manuscript, including its claims, references, and experimental reporting.

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.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. URL https://arxiv.org/abs/2110.14168.

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.

Arsene Fansi Tchango, Rishab Goel, Zhi Wen, Julien Martel, and Joumana Ghosn. DDXPlus: A new dataset for automatic medical diagnosis. *arXiv preprint arXiv:2205.09148*, 2022. URL https://arxiv.org/abs/2205.09148.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VtmBAGCN7o.

Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-Tau Yih, Daniel Fried, Sida Wang, and Tao Yu. DS-1000: A natural and reliable benchmark for data science code generation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 18319–18345. PMLR, 2023. URL https://proceedings.mlr.press/v202/lai23b.html.

Sangmook Lee, Dohyung Kim, Hyukhun Koh, Nakyeong Yang, and Kyomin Jung. Confidence-guided stepwise model routing for cost-efficient reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 31483–31491, 2026.

Shiyuan Li, Yixin Liu, Qingsong Wen, Chengqi Zhang, and Shirui Pan. Assemble your crew: Automatic multi-agent communication topology design via autoregressive graph generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 23142–23150, 2026.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*, 2017.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic LLM-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=XII0Wp1XA9.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5503a7c69d48a2f86fc00b3dc09de686-Abstract-Conference.html.

Rui Pan, Yinwei Dai, Zhihao Zhang, Gabriele Oliaro, Zhihao Jia, and Ravi Netravali. SpecReason: Fast and accurate inference-time compute via speculative reasoning. *arXiv preprint arXiv:2504.07891*, 2025.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In *Proceedings of NAACL-HLT*, pp. 2080–2094, 2021.

Subhro Roy and Dan Roth. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1743–1752, 2015.

Chunhao Tian, Yutong Wang, Xuebo Liu, Zhexuan Wang, Liang Ding, Miao Zhang, and Min Zhang. AgentInit: Initializing LLM-based multi-agent systems via diversity and expertise orchestration for effective and efficient collaboration. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 11870–11902, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.636. URL https://aclanthology.org/2025.findings-emnlp.636/.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-Agents enhances large language model capabilities. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=h0ZfDIrj7T.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Xingjian Wu, Junkai Lu, Siyu Yan, Xiangfei Qiu, Jilin Hu, Chenjuan Guo, and Bin Yang. Differentiable Mixture-of-Agents incentivizes swarm intelligence of large language models. *arXiv preprint arXiv:2605.15706*, 2026. doi: 10.48550/arXiv.2605.15706. URL https://arxiv.org/abs/2605.15706.

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.

Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. MasRouter: Learning to route LLMs for multi-agent systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15549–15572. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.757. URL https://aclanthology.org/2025.acl-long.757/.

Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. G-Designer: Architecting multi-agent communication topologies via graph neural networks. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 76678–76692. PMLR, 2025a. URL https://proceedings.mlr.press/v267/zhang25cu.html.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xiong-Hui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5492ecbce4439401798dcd2c90be94cd-Abstract-Conference.html.

Jusheng Zhang, Zimeng Huang, Yijia Fan, Ningyuan Liu, Mingyan Li, Zhuojie Yang, Jiawei Yao, Jian Wang, and Keze Wang. KABB: Knowledge-aware Bayesian bandits for dynamic expert coordination in multi-agent systems. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 74966–74996. PMLR, 2025c. URL https://proceedings.mlr.press/v267/zhang25aa.html.

Ruijia Zhang, Xinyan Zhao, Ruixiang Wang, Sigen Chen, Guibin Zhang, An Zhang, Kun Wang, and Qingsong Wen. SafeSieve: From heuristics to experience in progressive pruning for LLM-based multi-agent communication. *arXiv preprint arXiv:2508.11733*, 2025d.

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*, pp. 62743–62767, 2024.

A Supplementary Method Details

A.1 Predictor Parameterization and Training

Shared features and indexed heads. A shared network maps x to features $\mathbf{h}_\theta(x)$ using layer normalization, an affine projection, GELU, and dropout. Role-indexed, unordered-pair-indexed, and cardinality-indexed affine heads map these features to the factors in Eq. (4). The indices refer to the fixed agent pool; sharing factors across sets supports evaluation of new combinations of known agents, not automatic generalization to previously unseen agent identities.

Output decoding. Let $\mathrm{clip}(z; a, b) = \min\{b, \max\{a, z\}\}$. For positive numerical thresholds a_{prob} and a_{cost}, define

$$\psi(\mathbf{z}) = \begin{bmatrix} \sigma(\mathrm{clip}(z^{\mathrm{acc}}; -a_{\mathrm{prob}}, a_{\mathrm{prob}})) \\ \sigma(\mathrm{clip}(z^{\mathrm{qual}}; -a_{\mathrm{prob}}, a_{\mathrm{prob}})) \\ \mathrm{expm1}(\mathrm{clip}(z^{\mathrm{lat}}; 0, a_{\mathrm{cost}})) \\ \mathrm{expm1}(\mathrm{clip}(z^{\mathrm{tok}}; 0, a_{\mathrm{cost}})) \end{bmatrix}. \tag{24}$$

The clipping operations are inference-time numerical safeguards. All additive factors refer to the raw, pre-clipping scores $\mathbf{z}$, not the decoded outcomes $\psi(\mathbf{z})$.

Losses and normalization. Index training observations by n, let $\mathbf{z}_n = \mathbf{z}_\theta(x_n, S_n)$, and write $\mathbf{y}_n = \mathbf{y}_n^{\mathrm{obs}}$. Training uses raw correctness logits and sigmoid quality outputs:

$$\mathcal{L}_{\mathrm{acc}} = \frac{1}{|\mathcal{D}|} \sum_n \left[\mathrm{softplus}(z_n^{\mathrm{acc}}) - y_n^{\mathrm{acc}} z_n^{\mathrm{acc}} \right], \tag{25}$$

$$\mathcal{L}_{\mathrm{qual}} = \frac{1}{|\mathcal{D}|} \sum_n \left[\sigma(z_n^{\mathrm{qual}}) - y_n^{\mathrm{qual}} \right]^2. \tag{26}$$

Let $\mathcal{N}_{\mathrm{cost}}$ index valid cost records and $\varsigma_r > 0$ be the floored training standard deviation of $\log(1+y^r)$ for $r \in \{\mathrm{lat}, \mathrm{tok}\}$. The cost loss is

$$\mathcal{L}_{\mathrm{cost}} = \frac{\sum_{n \in \mathcal{N}_{\mathrm{cost}}} \sum_{r \in \{\mathrm{lat},\mathrm{tok}\}} \left[\dfrac{z_n^r - \log(1 + y_n^r)}{\varsigma_r} \right]^2}{2 \max\{1, |\mathcal{N}_{\mathrm{cost}}|\}}. \tag{27}$$

Invalid cost records are omitted from this sum, rather than assigned zero-cost targets. The raw-cost reference scales s_r used in selection are floored medians of valid training costs; they are distinct from the log-cost training scales ς_r. All scales and model parameters are fixed before testing.

Interpretation of resource estimates. Log-cost regression need not estimate the conditional arithmetic mean after decoding. Without clipping, its population squared-error target is $z^{r,\dagger} = \mathbb{E}[\log(1 + Y^r) \mid x, S]$, and generally

$$\exp(\mathbb{E}[\log(1 + Y^r) \mid x, S]) - 1 \neq \mathbb{E}[Y^r \mid x, S]. \tag{28}$$

The method therefore uses decoded costs as operational estimates, not as guaranteed unbiased expectations or calibrated upper bounds. Theoretical statements involving true expected budgets require their explicitly stated screening assumptions.

A.2 Offline Semantic Construction

Let $\mathbf{e}_m$ and $\mathbf{a}_i$ be normalized embeddings of skill and agent-role descriptions in the context embedding space. For temperature $\vartheta > 0$, define the semantic affinity

$$A_{mi} = \frac{\exp(\mathbf{e}_m^\top \mathbf{a}_i / \vartheta)}{\sum_j \exp(\mathbf{e}_m^\top \mathbf{a}_j / \vartheta)}. \tag{29}$$

Index training context-repeat groups by q, with observations $\mathcal{O}_q$ and representation x_q. For observation o, let $v_o = (y_o^{\mathrm{acc}} + y_o^{\mathrm{qual}})/2$. An agent-associated score averages outcomes of sets containing

it:

$$u_{qi} = \frac{\sum_{o \in \mathcal{O}_q} \mathbb{I}\{i \in S_o\} v_o}{\max\left\{1, \sum_{o \in \mathcal{O}_q} \mathbb{I}\{i \in S_o\}\right\}}. \tag{30}$$

This is an association statistic, not a standalone or causal contribution estimate. The skill target is $\mathbf{t}_q = \mathrm{Softmax}(A\mathbf{u}_q/\vartheta)$. Let X and T contain the contexts and targets as rows, and E_{skill} contain the skill embeddings as rows. A regularized fit combined with the semantic projection gives

$$\begin{aligned} \Theta &= (X^\top X + \lambda_W I)^{-1} X^\top \log(T + \zeta_W), \\ W &= (1 - \beta_W) E_{\mathrm{skill}}/\vartheta + \beta_W \Theta^\top, \end{aligned} \tag{31}$$

where $\lambda_W, \zeta_W > 0$, $\beta_W \in [0, 1]$, and the logarithm is elementwise.

For each agent, form the outcome-weighted context centroid

$$\overline{\mathbf{c}}_i = \frac{\sum_q \sum_{o \in \mathcal{O}_q} \mathbb{I}\{i \in S_o\}(\delta + v_o) x_q}{\sum_q \sum_{o \in \mathcal{O}_q} \mathbb{I}\{i \in S_o\}(\delta + v_o)}, \qquad \delta > 0, \tag{32}$$

setting it to zero when there are no contributing observations. For mixing weight $\beta_E \in [0, 1]$, define

$$\widetilde{\mathbf{e}}_i = \mathrm{norm}\left((1 - \beta_E)\mathbf{a}_i + \beta_E \,\mathrm{norm}(\overline{\mathbf{c}}_i)\right), \tag{33}$$

where normalization uses a positive norm floor. Both W and C_{base} depend only on training observations. In a held-out-combination protocol, excluded sets must be omitted from these construction stages as well as from Brain training.

A.3 Generalized Sinkhorn Updates

Restrict computation to available agent columns and let

$$\mathbf{K}_t = \exp(-C_{t,\mathcal{I}_t}/\varepsilon), \qquad \gamma = \frac{\rho}{\rho + \varepsilon}. \tag{34}$$

Starting from a positive vector $\mathbf{v}$, alternate

$$\mathbf{u} \leftarrow \boldsymbol{\mu}_t \oslash (\mathbf{K}_t \mathbf{v}), \qquad \mathbf{v} \leftarrow \left[\boldsymbol{\nu}_{t,\mathcal{I}_t} \oslash (\mathbf{K}_t^\top \mathbf{u})\right]^\gamma. \tag{35}$$

The exponent follows from the relaxed column marginal. At a stationary point, writing $\mathbf{q} = P^\top \mathbf{1}_M$ gives $\mathbf{v} = (\boldsymbol{\nu}/\mathbf{q})^{\rho/\varepsilon}$ and $\mathbf{q} = \mathbf{v} \odot (\mathbf{K}_t^\top \mathbf{u})$; substitution yields Eq. (35).

The implementation uses log-domain updates. After termination, recompute $\mathbf{u} = \boldsymbol{\mu}_t \oslash (\mathbf{K}_t \mathbf{v})$ and set

$$P_{t,\mathcal{I}_t} = \mathrm{diag}(\mathbf{u}) \mathbf{K}_t \,\mathrm{diag}(\mathbf{v}). \tag{36}$$

Unavailable columns remain zero. The row marginal is preserved up to numerical precision, while finite-iteration error is not assumed to vanish. The column marginal is neither fixed to the capacity prior nor constrained by a hard upper bound.

B Proofs of the Theoretical Results

B.1 Proof of Proposition 1

Proof. Fix x, B, and distinct $i, j, \ell, h \notin B$. All sets in $\mathcal{Q}_B$ have cardinality $|B| + 2$. For any additive score A_x, its contrast is

$$A_x(S_{ij}) + A_x(S_{\ell h}) - A_x(S_{ih}) - A_x(S_{\ell j}) = 0. \tag{37}$$

Every background contribution occurs equally often with positive and negative signs. Each of the four varying agents also appears once with each sign, and the common cardinality contribution cancels.

Let $e(S) = f_x(S) - A_x(S)$. Substituting into Eq. (16) gives

$$\Xi_x = e(S_{ij}) + e(S_{\ell h}) - e(S_{ih}) - e(S_{\ell j}).$$

The triangle inequality yields

$$|\Xi_x| \le \sum_{S \in \mathcal{Q}_B} |e(S)| \le 4 \max_{S \in \mathcal{Q}_B} |e(S)|.$$

Dividing by four proves the claim. □

Relation to the implemented predictor. For the raw correctness score in Eq. (4), role terms, cardinality terms, pairs within B, and pairs joining B to a varying agent all cancel in the same contrast. The remaining terms are

$$\widehat{\Xi}_x^{\mathrm{score}} = p_{ij}^{\mathrm{acc}}(x) + p_{\ell h}^{\mathrm{acc}}(x) - p_{ih}^{\mathrm{acc}}(x) - p_{\ell j}^{\mathrm{acc}}(x). \tag{38}$$

Thus, removing the pair head enforces a zero raw-score contrast, whereas the full model can represent nonzero contrasts. The background set permits comparison at a common admissible cardinality and can contain mandatory agents.

Scope. The reference logit f_x is finite only when the corresponding correctness probability lies strictly between zero and one. The proposition concerns approximation in the raw score space, before nonlinear decoding or numerical clipping. It is not a lower bound on classification accuracy or decoded probability error. In particular, a sigmoid applied to additive scores can itself produce a nonzero probability-space contrast. The result does not establish unique identification of factors, universal sufficiency of pairwise interactions, or a limitation of arbitrary Deep Sets or attention-based set networks.

B.2 Proof of Theorem 1

Proof. Condition on the current context, resource snapshot, candidate collection, preferences, and tolerance. Because $\mathcal{V}_t$ is finite and nonempty, a true-quality maximizer within the screened collection exists:

$$S_t^{\mathcal{V}} \in \arg\max_{S \in \mathcal{V}_t} Q_t(S).$$

Membership of $S_t^\star$ in $\mathcal{V}_t(\tau_t)$ implies

$$\widehat{Q}_t(S_t^\star) \ge \max_{S \in \mathcal{V}_t} \widehat{Q}_t(S) - \tau_t \ge \widehat{Q}_t(S_t^{\mathcal{V}}) - \tau_t. \tag{39}$$

Both sets belong to $\mathcal{V}_t$, so the assumed prediction-error bound applies to each. Consequently,

$$\begin{aligned} Q_t(S_t^{\mathcal{V}}) - Q_t(S_t^\star) &= \big[Q_t(S_t^{\mathcal{V}}) - \widehat{Q}_t(S_t^{\mathcal{V}})\big] \\ &\quad + \big[\widehat{Q}_t(S_t^{\mathcal{V}}) - \widehat{Q}_t(S_t^\star)\big] \\ &\quad + \big[\widehat{Q}_t(S_t^\star) - Q_t(S_t^\star)\big] \\ &\le \epsilon_t + \tau_t + \epsilon_t. \end{aligned} \tag{40}$$

The left-hand side is nonnegative by the definition of $S_t^{\mathcal{V}}$, proving Eq. (19).

Under Eq. (20), the screened collection is precisely $\mathcal{C}_t \cap \mathcal{F}_t$. Hence

$$Q_t(S_t^{\mathrm{oracle}}) - Q_t(S_t^{\mathcal{V}}) = \Delta_t^{\mathrm{cand}}.$$

Adding this identity to Eq. (40) gives

$$Q_t(S_t^{\mathrm{oracle}}) - Q_t(S_t^\star) \le \Delta_t^{\mathrm{cand}} + 2\epsilon_t + \tau_t. \tag{41}$$

Correct screening makes $S_t^\star$ truly feasible, so optimality of S_t^{oracle} gives nonnegativity of this final gap. □

A sufficient outcome-error condition. The theorem uses quality errors after decoding. Let $\mathbf{D}_s = \mathrm{diag}(1, 1, s_{\mathrm{lat}}, s_{\mathrm{tok}})$ and $\widetilde{\boldsymbol{\alpha}}_t = [\alpha_{t,\mathrm{acc}}, \alpha_{t,\mathrm{qual}}, 0, 0]^\top$. Since $\|\widetilde{\boldsymbol{\alpha}}_t\|_1 = 1$, a sufficient condition is

$$\max_{S \in \mathcal{V}_t} \left\| \mathbf{D}_s^{-1} \big(\widehat{\mathbf{y}}_\theta(x_t, S) - \mathbf{y}(x_t, S) \big) \right\|_\infty \le \epsilon_t. \tag{42}$$

Indeed, Hölder's inequality gives

$$|\widehat{Q}_t(S) - Q_t(S)| = \left| \widetilde{\boldsymbol{\alpha}}_t^\top \mathbf{D}_s^{-1} (\widehat{\mathbf{y}}_\theta - \mathbf{y}) \right| \le \epsilon_t.$$

This condition is imposed on decoded outcomes, not on logits or log-cost training targets, and is not guaranteed by a low empirical training loss.

Interpretation. The first bound holds relative to the actual screened candidates without assuming their true feasibility. The oracle bound additionally requires correct screening on those candidates; neither log-cost regression nor slot reservation alone establishes this condition. Cost minimization satisfies the quality bound because the chosen set remains within $\mathcal{V}_t(\tau_t)$, not because true cost optimality has been proved. The argument is conditional on a single context and does not require independence between candidate generation and prediction. It does not establish cumulative multi-step performance or robustness to unmodeled changes in execution latency.

Connection to the diagnostic experiments. Unseen-combination accuracy tests selection among held-out sets of known agents; it does not provide a uniform prediction-error guarantee. Likewise, empirical coverage—whether at least one executed candidate answers correctly—is distinct from Δ_t^{cand}, which compares expected quality with a feasible oracle. Coverage and final accuracy provide complementary diagnostics, not a numerical verification of the theorem's oracle gap.

C Additional Experimental Settings

C.1 Dataset Details

We evaluate TeamRoute on nine benchmarks covering broad-domain reasoning, mathematics, programming, multi-hop question answering, and medical diagnosis. Table 3 summarizes the datasets, task categories, and evaluation metrics.

We use public dataset splits and official evaluation procedures where available. All methods use `gpt-oss-120b` as the backbone and are evaluated on the same query set for each benchmark. For methods requiring offline optimization, we adopt a common few-shot training budget of 40–80 queries per benchmark, sampled from the corresponding training split. The underlying LLM agents remain frozen during adaptation.

Table 3: Datasets, task categories, and evaluation metrics.

Dataset	**Task category**	**Metric**
MMLU (Hendrycks et al., 2020)	Broad-domain reasoning	Accuracy
GSM8K (Cobbe et al., 2021)	Grade-school math reasoning	Accuracy
MultiArith (Roy & Roth, 2015)	Arithmetic reasoning	Accuracy
SVAMP (Patel et al., 2021)	Mathematical word problems	Accuracy
AQuA (Ling et al., 2017)	Algebraic reasoning	Accuracy
HumanEval (Chen et al., 2021)	Code generation	Pass@1 / Accuracy
DS-1000 (Lai et al., 2023)	Code reasoning and debugging	Pass@1 / Accuracy
HotpotQA (Yang et al., 2018)	Multi-hop question answering	Accuracy / F1
DDXPlus (Fansi Tchango et al., 2022)	Medical diagnosis reasoning	Accuracy

C.2 Task-Specific Agent Pool Configuration

We construct task-specific agent pools using a shared backbone model and specialized role prompts. Agents are differentiated by their functional responsibilities and dynamically selected according to the task context.

Math-oriented agents. These agents specialize in problem formalization, quantitative relation analysis, solution planning, numerical computation, and independent verification. Their complementary responsibilities include identifying assumptions, tracing intermediate steps, checking arithmetic consistency, and exploring alternative solution methods.

Code-oriented agents. These agents specialize in specification analysis, algorithm design, implementation planning, and correctness checking. Their responsibilities cover programming-language and library semantics, data structures, dataflow, boundary cases, testing, and computational efficiency.

Open-domain QA agents. These agents specialize in question interpretation, domain-specific reasoning, evidence assessment, and consistency checking. Their responsibilities include decomposing questions, examining alternative explanations, identifying unsupported assumptions, and reconciling conflicting evidence.

Aggregation agent. A separate aggregation component integrates the selected specialists' contributions or selects among their candidate solutions to produce the final response.

C.3 Prompt Construction

Our prompts comprise a shared task description, a role-specific instruction, and a task-dependent output requirement. Selected agents process the same task context independently, while their role instructions encourage complementary contributions.

Generic agent system prompt.

```
You are a specialized expert for the given task.

Role: {role_name}
Responsibility: {role_description}

Analyze the task independently from your assigned perspective.
Follow the task constraints and provide the requested output.

Output requirement: {output_contract}
```

Task context prompt.

```
[TASK]
{task_description}

[TASK-SPECIFIC CONSTRAINTS]
{applicable_constraints}
```

Aggregation prompt.

```
Given the original task and the selected experts' contributions,
produce a single final solution.

Treat expert outputs as advisory evidence. Resolve disagreements
against the task requirements rather than following them blindly.
Follow the required output format.

[TASK]
{task_description}

[SELECTED EXPERT CONTRIBUTIONS]
{role-labelled_outputs}

[OUTPUT REQUIREMENT]
```

```
{final_output_contract}
```

Task-specific instantiation. For mathematical tasks, prompts emphasize quantity interpretation, explicit derivation, and numerical verification. For coding tasks, they emphasize input–output requirements, implementation semantics, and executable correctness. Depending on the task, specialists provide structured advice or complete candidate solutions.

C.4 Skill Demand Decomposition

We define a compact set of task-specific skill descriptions covering problem interpretation, solution construction, implementation or computation, and verification. These descriptions specify complementary capability dimensions for routing rather than a one-to-one assignment of skills to agents.

Given a context embedding $x_t \in \mathbb{R}^d$, an offline-fitted projection $W \in \mathbb{R}^{M\times d}$ produces a normalized distribution over the M skills:

$$\boldsymbol{\mu}_t = \mathrm{Softmax}(Wx_t), \qquad \mu_{t,m} \geq 0, \qquad \sum_{m=1}^{M} \mu_{t,m} = 1. \tag{43}$$

The projection combines semantic representations with training execution outcomes, as described in Appendix A.2. It remains frozen during inference, while $\boldsymbol{\mu}_t$ is recomputed for each context. Each component represents the relative demand for a skill, allowing multiple capabilities to contribute to the same task.

Role in routing. The skill-demand distribution guides transport in two ways. It specifies the required skill marginal and modulates the semantic cost through $C_t = \mathrm{diag}(\mathbf{1}_M + \boldsymbol{\mu}_t)C_{\mathrm{base}}$. A more strongly demanded skill therefore receives greater transport mass and a larger penalty for semantic mismatch. This mechanism guides candidate generation, while the Experience Brain subsequently evaluates the collective quality and cost of the resulting agent sets.

C.5 Optimization Hyperparameters

Table 4 summarizes the optimization and routing settings. We train the Experience Brain using AdamW, with each training example corresponding to a context–agent-set observation. Early stopping monitors the validation loss, and the checkpoint with the lowest validation loss is retained. The coefficient λ_{cost} weights the cost-prediction term in Eq. (5); it is distinct from the cost preferences used for online selection.

For candidate generation, generalized Sinkhorn scaling is performed in the log domain with a final row-scaling update. The iteration limit controls numerical computation and does not assume exact convergence to the transport optimum.

The backbone LLM and sentence encoder remain frozen throughout training and evaluation. After offline learning, the Experience Brain, skill-demand projection W, and base semantic cost matrix C_{base} are fixed during inference. Skill demands, context-adjusted costs C_t, and transport plans are recomputed for each routing context without updating these learned components.

C.6 Execution-Cost Analysis

We compare TeamRoute and DMoA by varying one execution budget while holding the other fixed. The token-budget sweep fixes the latency limit at $60\,\mathrm{s}$, and the latency-budget sweep fixes the token limit at 6000. The evaluated configurations are

$$\begin{aligned} \text{Token sweep:} \quad & B^{\mathrm{lat}} = 60\,\mathrm{s}, \quad B^{\mathrm{tok}} \in \{1500, 3000, 6000\}, \\ \text{Latency sweep:} \quad & B^{\mathrm{tok}} = 6000, \quad B^{\mathrm{lat}} \in \{10, 30, 60\}\,\mathrm{s}. \end{aligned} \tag{44}$$

Both methods are compared at the same budget pairs. The horizontal axes represent configured limits rather than realized token consumption or measured execution latency. The curves therefore evaluate accuracy under different resource allowances, not equal realized costs or guaranteed budget compliance.

Table 4: Optimization and routing hyperparameters for TeamRoute. The hidden dimension refers to the Experience Brain's shared feature network, not the sentence encoder's output dimension.

Parameter	Setting
Experience Brain	
Optimizer	AdamW
Learning rate	5×10^{-4}
Weight decay	0.01
Batch size	64 context–agent-set examples
Shared hidden dimension	64
Activation function	GELU
Dropout probability	0.3
Maximum training epochs	240
Early-stopping patience	24 epochs
Gradient-norm clipping threshold	1.0
Cost-prediction loss weight λ_{cost}	0.2
Optimal-transport routing	
Entropy regularization ε	0.1
Capacity-marginal KL penalty ρ	0.5
Maximum Sinkhorn iterations	160

C.7 Generalization to Unseen Agent Combinations

Combination holdout. We retain the original training, validation, and test query splits. With ten agents and admissible cardinalities $\mathcal{K} = \{2, 3, 4\}$, there are $\binom{10}{2} + \binom{10}{3} + \binom{10}{4} = 375$ complete combinations. We hold out twenty percent within each cardinality, yielding a common family $\mathcal{S}_{\text{hold}}$ of 75 sets.

Table 5: Cardinality-stratified combination holdout. Counts refer to distinct agent sets.

Set size	All	Held out	Remaining
$k = 2$	45	9	36
$k = 3$	120	24	96
$k = 4$	210	42	168
Total	375	75	300

All records associated with $\mathcal{S}_{\text{hold}}$ are removed from training and validation and do not contribute to predictor fitting, early stopping, or hyperparameter selection. The remaining training data cover all agents, all 45 unordered pairs, and all admissible cardinalities. Pair coverage includes co-occurrence in larger retained sets; a held-out two-agent set can therefore have its pair represented within a larger training combination. This protocol tests unseen complete combinations, not unseen agents, pairs, or activation sizes.

Predictors and selection. Full Brain, w/o Pair, and DeepSets are trained from scratch on the same filtered data. Full Brain retains the role, pairwise, and cardinality factors; w/o Pair removes only the pairwise factors. DeepSets uses the query-conditioned sum-pooling implementation.

For each test query, every predictor scores the same 75 held-out sets and applies the sequential selection rule from Section 4.3. Predicted latency and token consumption determine budget admissibility. Predicted correctness and auxiliary quality form $\widehat{Q}_t$, candidates within τ_t of the best predicted quality are retained, and predicted weighted cost determines the final choice. Budgets, quality weights, tolerance, and cost preferences are shared across predictors. No OT prior score is added to the selection criterion.

Only the chosen sets are executed. Selections of the same set for the same query share one execution, so at most three distinct selected-set executions are needed per query. We report the accuracy of each predictor's selected answer. Because candidates are supplied directly, this experiment isolates predictor-guided selection from OT candidate generation.

C.8 Candidate Coverage and Selection Quality

Controlled setup. We fix the test queries, original Full Brain checkpoint, and selection-policy parameters without retraining the Brain. Random, Softmax, and OT each propose twelve distinct sets per query, with four sets for each cardinality in $\{2, 3, 4\}$. Within each cardinality, Gumbel-perturbed ranking of the proposal weights selects candidates without replacement. This diagnostic replaces deterministic top-J_k truncation with matched-budget stochastic sampling while keeping the candidate count and cardinality allocation fixed.

Generators and final selection. Random assigns equal weights to combinations within a fixed cardinality and samples them uniformly. Softmax replaces Sinkhorn with cost-based softmax normalization to produce agent weights from the original transport costs. OT uses column masses of the original Sinkhorn plan, $\mathbf{w}_t = P_t^\top \mathbf{1}_M$. Generator-specific weights guide proposals but do not enter the final selection score.

Every generator uses the same frozen Brain and the same budget-screening, quality-tolerance, and cost-minimization rule. Candidate correctness labels are used only for evaluation.

Metrics. Let $\mathcal{U}_{\text{test}}$ contain n test queries. For generator g, write $\mathcal{C}_g(u)$ for its candidate family, $S_g^\star(u)$ for the selected set, and $r(u, S) \in \{0, 1\}$ for the correctness of the evaluated answer. Candidate coverage and final accuracy are

$$\text{Coverage}_g = \frac{100}{n} \sum_{u \in \mathcal{U}_{\text{test}}} \max_{S \in \mathcal{C}_g(u)} r(u, S), \tag{45}$$

$$\text{FinalAcc}_g = \frac{100}{n} \sum_{u \in \mathcal{U}_{\text{test}}} r\big(u, S_g^\star(u)\big). \tag{46}$$

All proposed candidates are evaluated for this diagnostic, and both metrics use the same candidate outcomes. Their difference measures the fraction of queries with a correct proposal that the selector does not choose. Unlike the default online procedure, executing all candidates is needed only to measure diagnostic coverage. Neither metric directly estimates the expected feasible-oracle gap in Theorem 1.

C.9 Scalability Configurations

Each configuration is represented by (N, K), where N is the agent-pool size and K is the maximum number of agents selected for a routing decision. We evaluate

$$(N, K) \in \{(1, 1), (5, 3), (10, 4), (15, 5)\}.$$

The selected cardinality $|S_t|$ is bounded by K and need not always attain that limit. The configuration $(1, 1)$ has only one possible nonempty agent set; the larger configurations permit alternative combinations. Figure 6 reports accuracy on HumanEval, AQuA, and MultiArith.

Because the pool size and activation limit increase together, this experiment measures the combined effect of the two changes. It does not isolate their individual contributions or measure routing runtime, memory scaling, or throughput. Accuracy across these configurations does not by itself establish zero-shot adaptation to newly introduced agent identities.

D SHOWCASES

The following examples illustrate stepwise role configurations for MultiArith, HumanEval, and AQuA. Each column denotes a reasoning step, with the listed roles forming a parallel team.

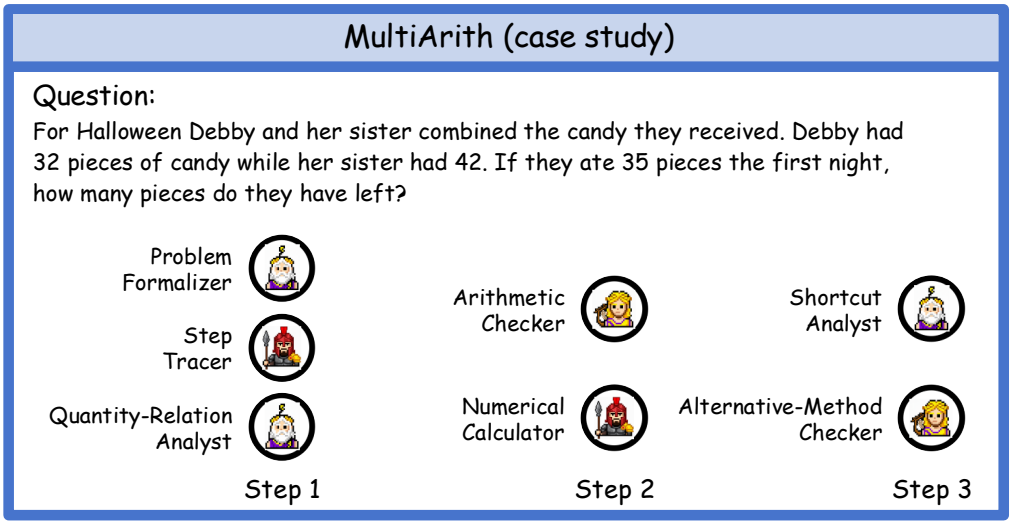


Figure 7: Case study on MultiArith.

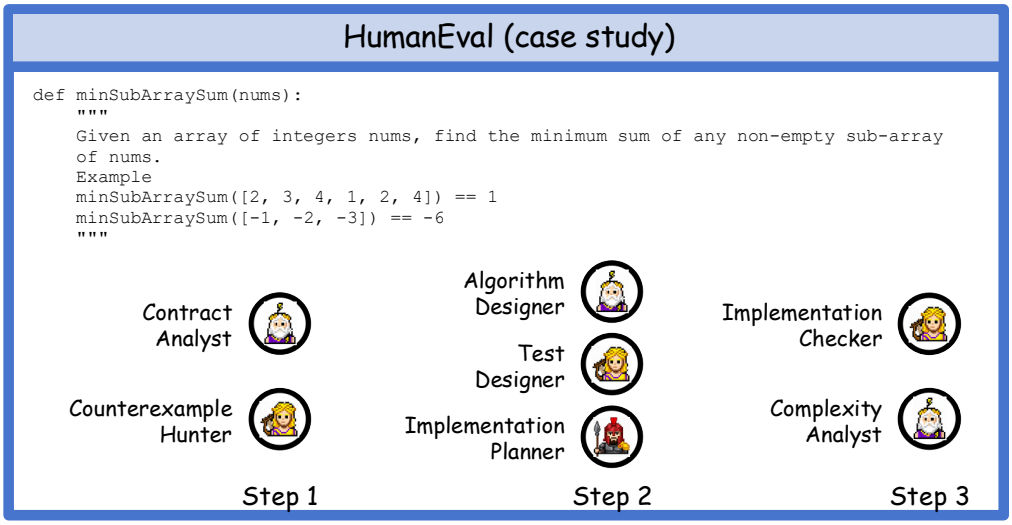


Figure 8: Case study on HumanEval.

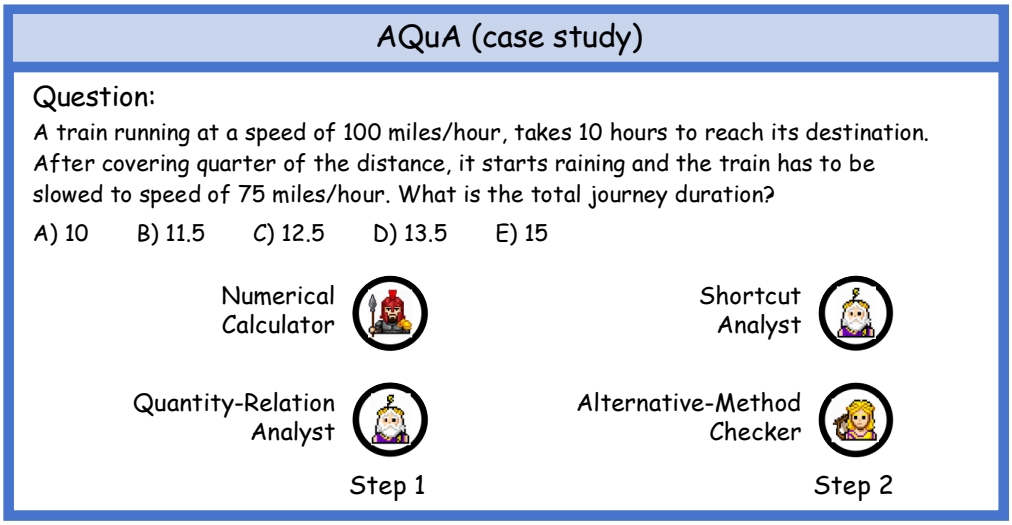


Figure 9: Case study on AQuA