ICML
International Conference
On Machine Learning

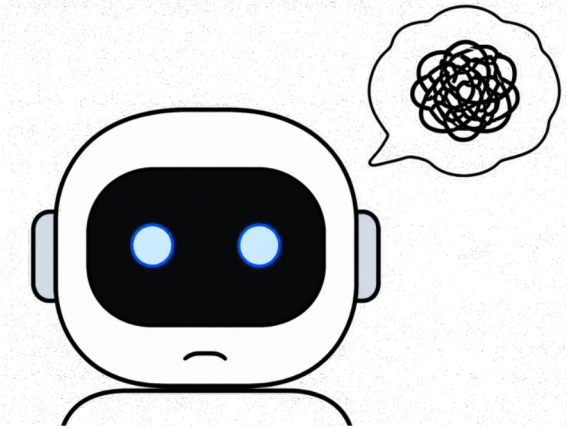
RADAR: Redundancy-Aware Diffusion for Multi-Agent Communication Structure Generation

Zhen Zhang[1], Wanjing Zhou[1], Juncheng Li[2],
Hao Fei[3], Jun Wen[4], Wei Ji[1]
[1]Nanjing University, [2]Zhejiang University,
[3]University of Oxford, [4]MBZUAI

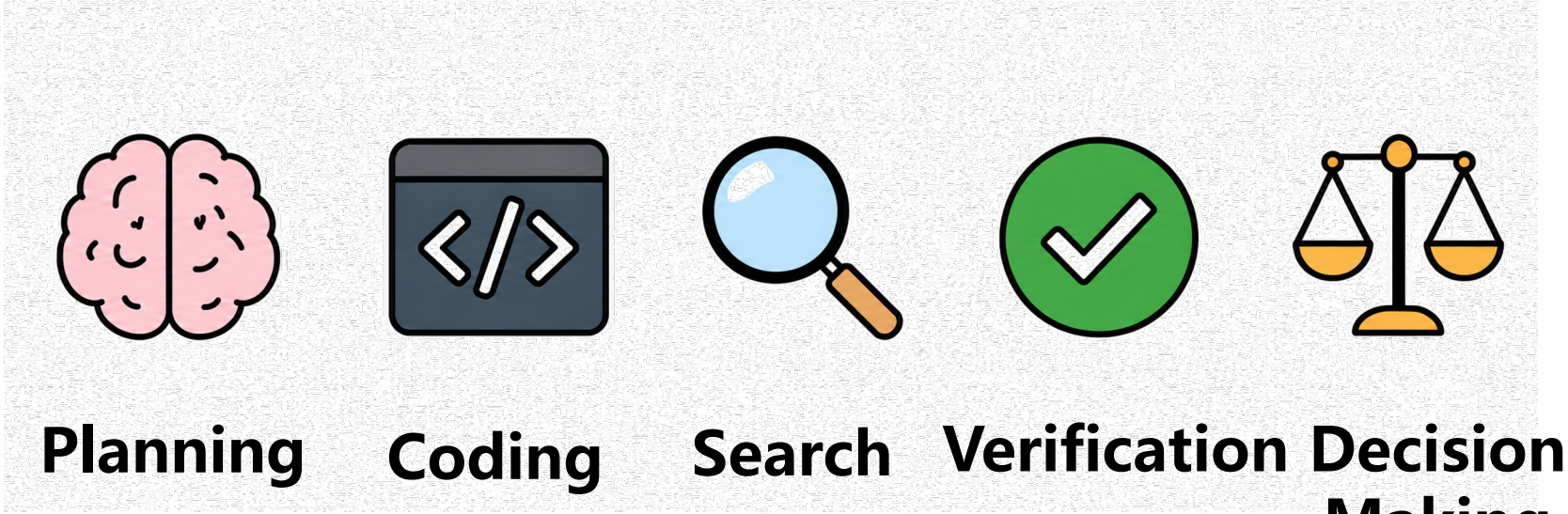
Background

- Large Language Model based agents have achieved great success across a wide spectrum of domains, including code generation, web navigation and complex reasoning, etc.

- Despite the impressive capabilities of LLM-based agents, single-agent architectures face limitations in specialization, parallel exploration, and robustness



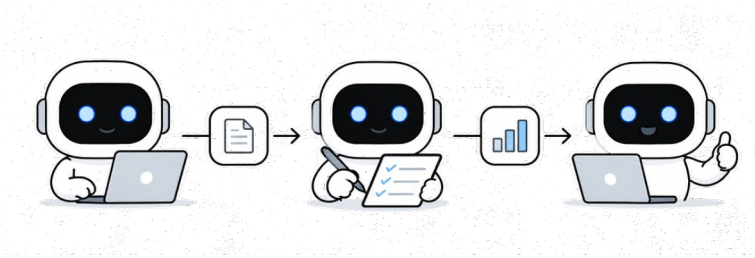
I have to do **EVERYTHING** by myself



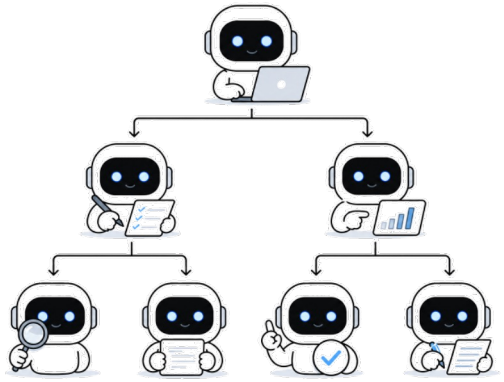
Background

- Recent research has turned to multi-agent systems (MAS), where multiple autonomous agents interact through communication, coordination, and role specialization

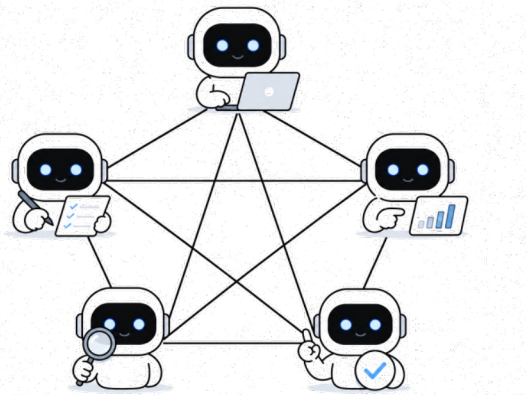
- Prior work on multi-agent systems has largely been constrained to static, hand-crafted configurations, such as chain structures, tree architectures, fully connected topologies, etc.



Chain



Tree

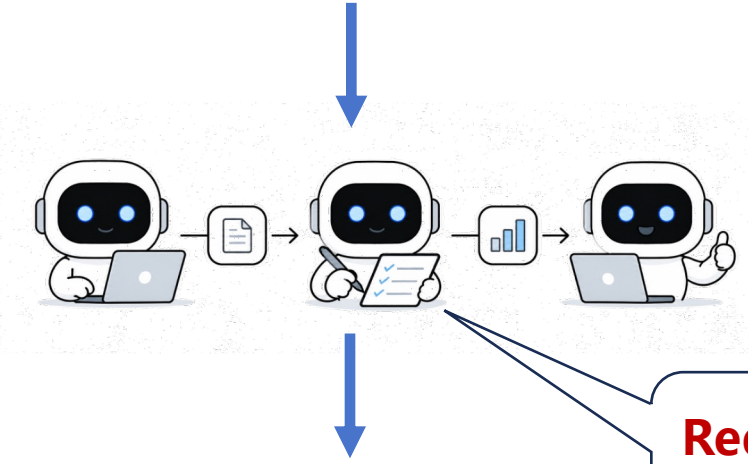


Complete Graph

Background

- **Applying a single fixed collaboration pattern across all tasks either incurs redundant communication overhead for simple problems or limits performance on more complicate ones**

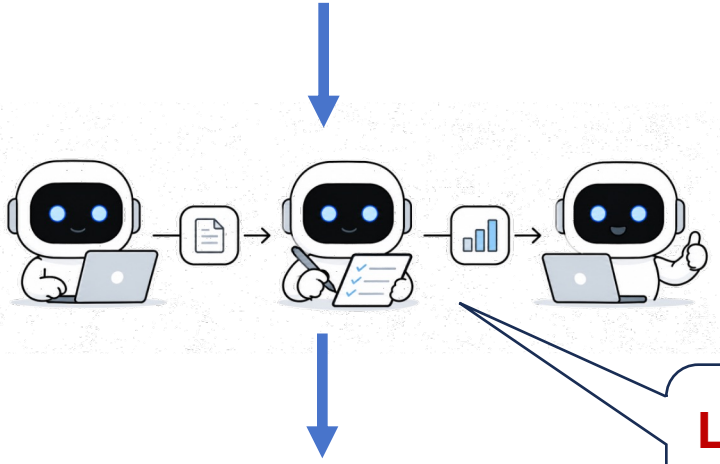
Query: Compute the value of the expression *1+1*?



Answer: 2

Redundant Communication

Query: Generate code that implements a *balanced binary tree*.



Answer: Code compile failed

Limited Performance

Background

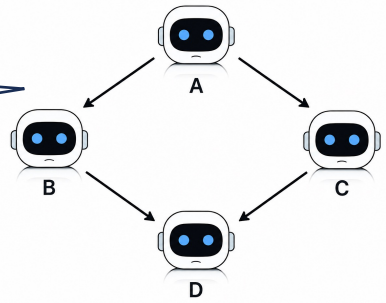
- In this paper, we propose RADAR, which synthesizes the entire collaboration graph via iterative conditional graph diffusion models and generates collective intelligence tailored to the specific task

Query: Compute the value of the expression *1+1*?

Answer: 2

Reduced Communication

Task Specific Structure



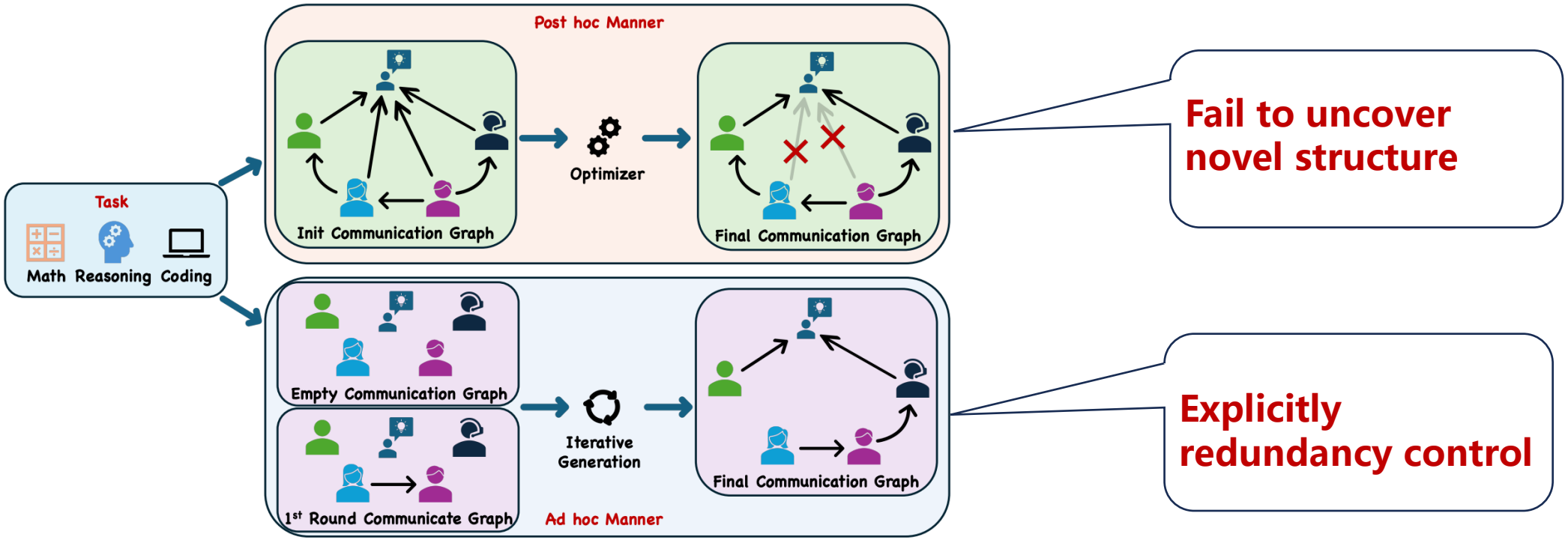
Query: Generate code that implements a *balanced binary tree*.

Answer: Code compile success

Related Work

- Recent models generate the communication structure in a ***single step.*** This often leads to inefficient topologies with redundant links, etc.

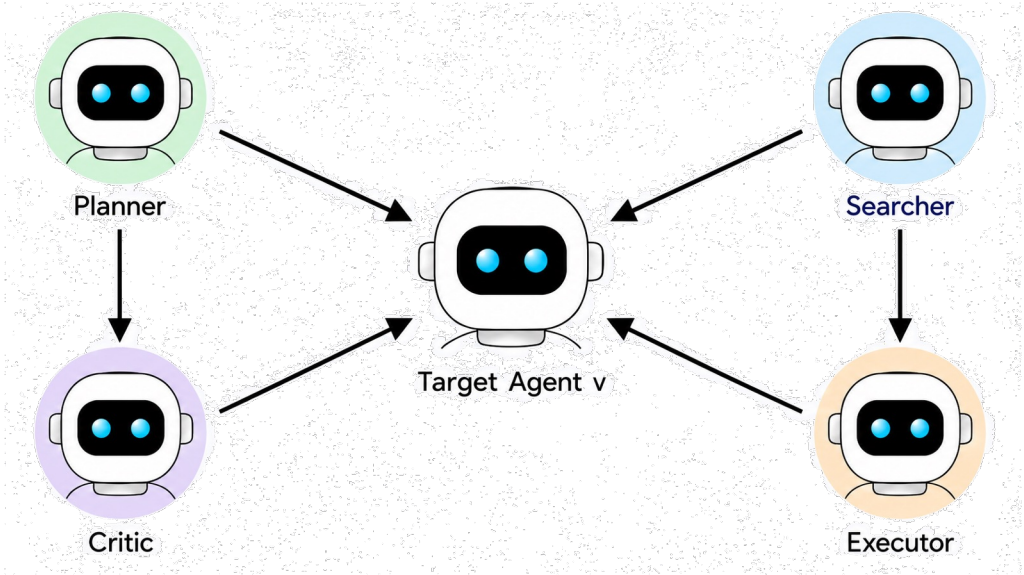
- To improve efficiency, they often optimize it in a ***post hoc manner***, rather than jointly reasoning about structure formation and redundancy control during topology construction



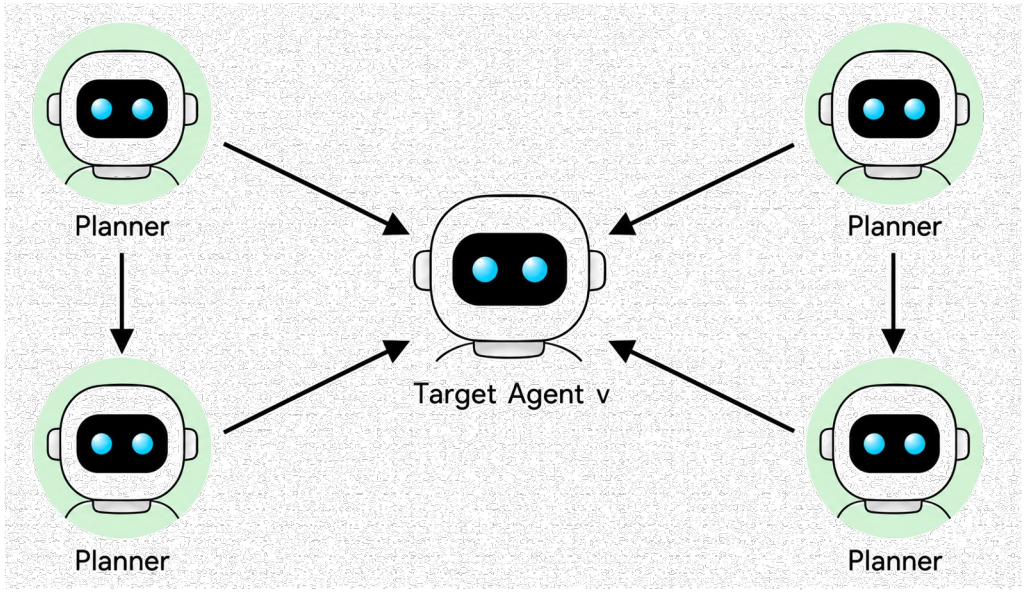
Method

- We formulate communication topology design as a step-by-step generation process, guided by the *effective size* of the graph

$$\varphi^i(v_k) = |N_i(v_k)| - \frac{\sum_{j,q \in N_i(v_k)} A_{jq} I[r(j) = r(q)]}{|N_i(v_k)|}$$



High Effective Size ---> Low Redundancy



Low Effective Size ---> High Redundancy

The Forward Process

- We introduce a forward diffusion process that progressively masks nodes together with their connected edges

$$q_{\varphi}(\pi|G_0,\varphi)=\prod_{t} q_{\varphi}\left(\pi_t|G_0,\varphi,\pi_{(<t)}\right)$$

- Ordering network $q_{\varphi}(\pi|G_0,\varphi)$ samples a node $v_{\pi(t)}$ to be masked at each diffusion step t, yielding the corresponding partially masked graph G_t

$$q_{\varphi}\left(\pi_t|G_0,\varphi,\pi_{(<t)}\right)=\frac{exp(h_t+\varphi(v_t))}{\sum_{j\notin\pi_{(<t)}} exp(h_j)}$$

- It outputs the probability of node v_t being selected at step t

The Reverse Process

- During the reverse generative phase, a denoising network $p_\theta(G_t|G_{t+1}, Q)$ progressively reconstructs the graph by inverting the forward diffusion process conditioning on the query Q

- At step t, the denoising network takes the partially masked graph G_{t+1} as input and maps each node v_i into a latent space

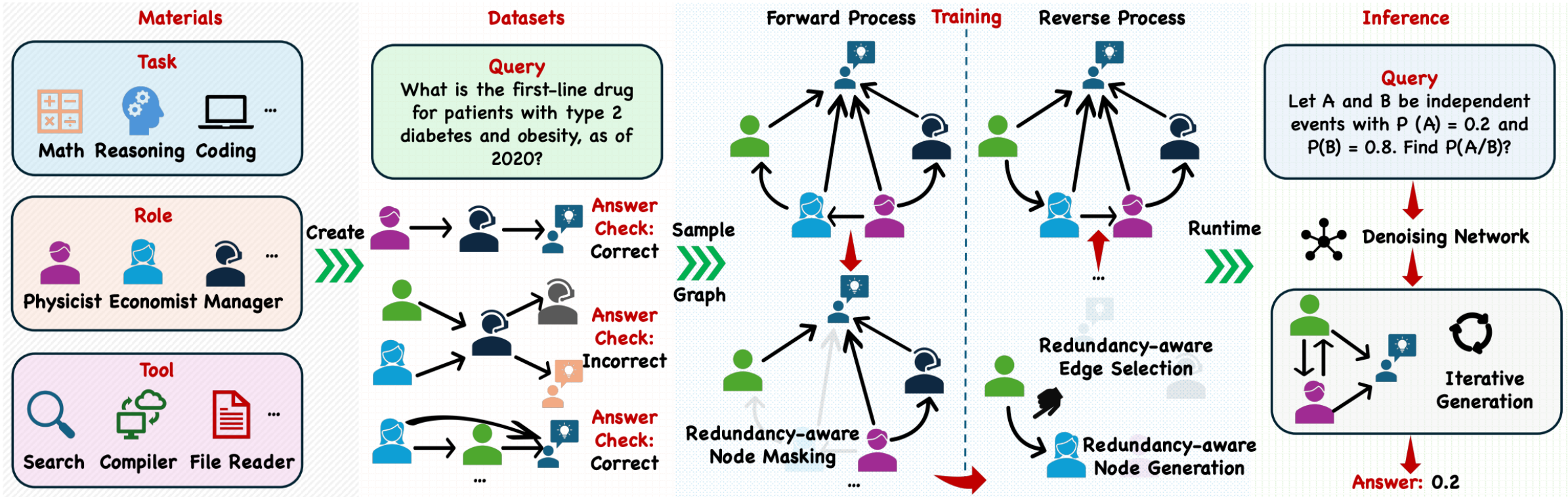
Any GNN model can be used!

- Based on these embeddings, multilayer perceptions are utilized to predict the agent role of the newly recovered node v_{π_t} and its connectivity to the set of previously denoised nodes $\{v_{\pi_{(>t)}}\}$

Method

南京大学 NANJING UNIVERSITY

➢ We present an overview of the proposed **RADAR** framework



Training

- We use a *reinforcement learning* based training strategy to jointly update the diffusion ordering network $q_{\varphi}(\pi|G_0,\varphi)$ and the denoising network $p_{\theta}(G_t|G_{t+1},Q)$ using gradient descent

- At each training iteration, for the i-th training graph $G_0^{(i)}$, we generate *M diffusion trajectories* by sampling node-masking orderings $\pi^{i,m}$ from the ordering network

- Each trajectory consists of a sequence of partially masked graphs $\{G_t^{i,m}\}_{1\le t\le N}$, where N denotes the number of nodes

- Conditioned on these sampled trajectories, the denoising network is trained to minimize the negative *variational lower bound* using stochastic gradient descent

Experiments

南京大学 NANJING UNIVERSITY

- **Datasets: we conduct experiments using three categories of publicly available datasets**

- **Three key category baselines:**
  - ***Single Agent Methods*, *Multi-Agent Systems*, *Autonomous Multi-Agent Systems***

Category	Dataset	Format	Metric	#Samples
General Reasoning	MMLU	Multi-choice	Accuracy	153
Math Solving	GSM8K	Number	Accuracy	1,319
	MultiArith	Number	Accuracy	600
	SVAMP	Number	Accuracy	1,000
	AQuA	Multi-choice	Accuracy	254
Code Generation	HumanEval	Code	Pass@1	164

Experiments

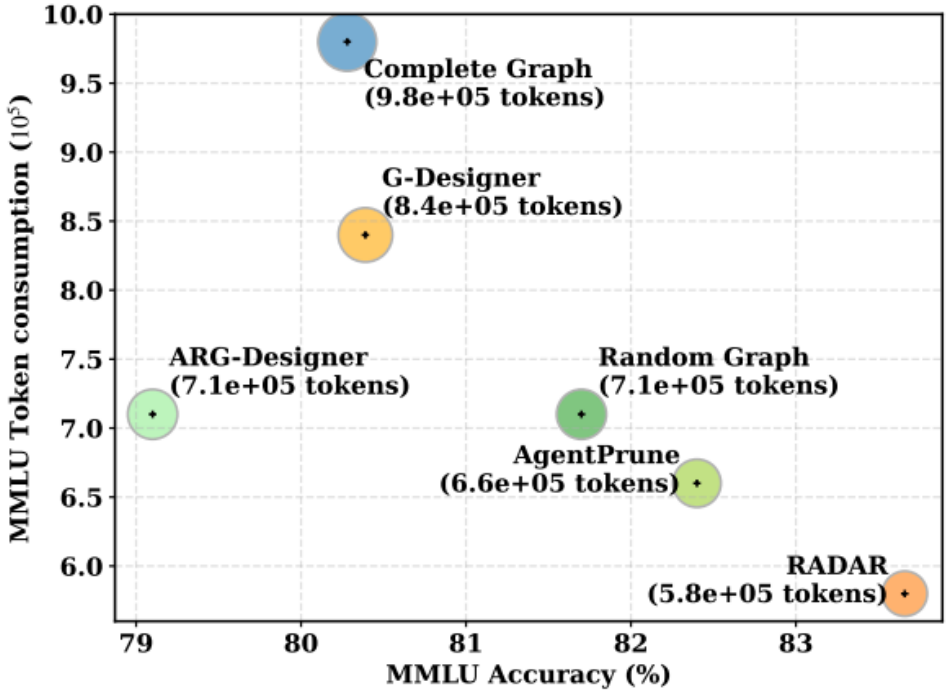
南京大学 NANJING UNIVERSITY

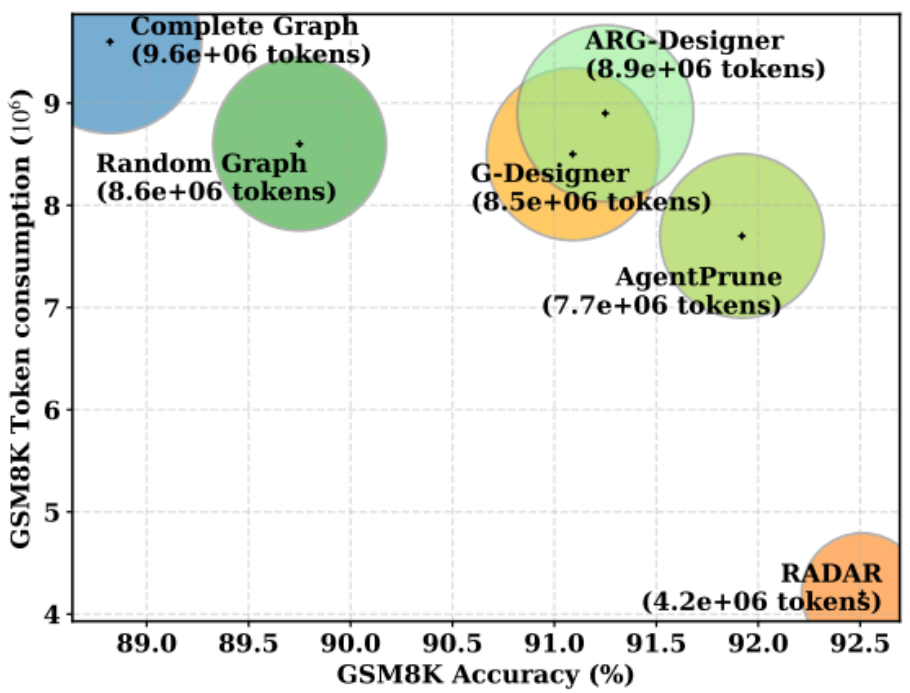
➢ Performance comparison across three categories of baselines

Methods	Auto	MMLU	GSM8K	MultiArith	SVAMP	AQuA	HumanEval	Avg.
Vanilla	✗	78.54	87.45	96.85	86.67	78.92	87.08	85.92
CoT	✗	79.26$_{\uparrow 0.72}$	87.10$_{\downarrow 0.35}$	96.31$_{\downarrow 0.54}$	87.33$_{\uparrow 0.66}$	75.20$_{\downarrow 3.72}$	88.13$_{\uparrow 1.05}$	85.55
ComplexCoT	✗	79.80$_{\uparrow 1.26}$	86.89$_{\downarrow 0.56}$	96.70$_{\downarrow 0.15}$	87.67$_{\uparrow 1.00}$	75.59$_{\downarrow 3.33}$	87.49$_{\uparrow 0.41}$	85.69
SC (COT×5)	✗	80.66$_{\uparrow 2.12}$	87.57$_{\uparrow 0.12}$	96.58$_{\downarrow 0.27}$	88.00$_{\uparrow 1.33}$	82.28$_{\uparrow 3.36}$	88.60$_{\uparrow 1.52}$	87.28
MultiPersona	✗	77.69$_{\downarrow 0.85}$	87.50$_{\uparrow 0.05}$	97.49$_{\uparrow 0.64}$	87.00$_{\uparrow 0.33}$	79.23$_{\uparrow 0.31}$	88.32$_{\uparrow 1.24}$	86.21
LLM-Debate	✗	80.56$_{\uparrow 2.02}$	89.47$_{\uparrow 2.02}$	97.33$_{\uparrow 0.48}$	89.00$_{\uparrow 2.33}$	79.70$_{\uparrow 0.78}$	88.68$_{\uparrow 1.60}$	87.46
LLM-Blender	✗	80.29$_{\uparrow 1.75}$	88.35$_{\uparrow 0.90}$	97.29$_{\uparrow 0.44}$	87.33$_{\uparrow 0.66}$	78.99$_{\uparrow 0.07}$	88.80$_{\uparrow 1.72}$	86.84
DyLAN	✗	79.86$_{\uparrow 1.32}$	89.98$_{\uparrow 2.53}$	97.12$_{\uparrow 0.27}$	88.67$_{\uparrow 2.00}$	79.59$_{\uparrow 0.67}$	90.42$_{\uparrow 3.34}$	87.61
AgentVerse	✗	78.39$_{\downarrow 0.15}$	89.91$_{\uparrow 2.46}$	97.50$_{\uparrow 0.65}$	88.33$_{\uparrow 1.66}$	77.47$_{\downarrow 1.45}$	89.29$_{\uparrow 2.21}$	86.82
MacNet	✗	79.55$_{\uparrow 1.01}$	87.95$_{\uparrow 0.50}$	96.03$_{\downarrow 0.82}$	86.00$_{\downarrow 0.67}$	79.23$_{\uparrow 0.31}$	84.57$_{\downarrow 2.51}$	85.55
AutoAgents	✓	79.59$_{\uparrow 1.05}$	87.69$_{\uparrow 0.24}$	96.42$_{\downarrow 0.43}$	86.34$_{\downarrow 0.33}$	78.65$_{\downarrow 0.27}$	87.64$_{\uparrow 0.56}$	86.05
GPTSwarm	✓	78.36$_{\downarrow 0.18}$	89.14$_{\uparrow 1.69}$	96.79$_{\downarrow 0.06}$	88.67$_{\uparrow 2.00}$	80.71$_{\uparrow 1.79}$	89.32$_{\uparrow 2.24}$	87.17
ADAS	✓	78.39$_{\downarrow 0.15}$	86.12$_{\downarrow 1.33}$	96.02$_{\downarrow 0.83}$	86.33$_{\downarrow 0.34}$	77.71$_{\downarrow 1.21}$	84.19$_{\downarrow 2.89}$	84.79
AgentSquare	✓	79.58$_{\uparrow 1.04}$	87.62$_{\uparrow 0.17}$	97.77$_{\uparrow 0.92}$	88.00$_{\uparrow 1.33}$	81.50$_{\uparrow 2.58}$	89.08$_{\uparrow 2.00}$	87.26
AFlow	✓	81.80$_{\uparrow 3.26}$	91.16$_{\uparrow 3.71}$	96.22$_{\downarrow 0.63}$	88.33$_{\uparrow 1.66}$	80.90$_{\uparrow 1.98}$	90.93$_{\uparrow 3.85}$	88.22
G-Designer	✓	80.39$_{\uparrow 1.85}$	91.09$_{\uparrow 3.64}$	97.78$_{\uparrow 0.93}$	90.00$_{\uparrow 3.33}$	80.75$_{\uparrow 1.83}$	89.37$_{\uparrow 2.29}$	88.23
AgentPrune	✓	82.40$_{\uparrow 3.86}$	91.92$_{\uparrow 4.47}$	97.88$_{\uparrow 1.03}$	90.37$_{\uparrow 3.70}$	80.93$_{\uparrow 2.01}$	87.17$_{\uparrow 0.09}$	88.22
GTD	✓	79.41$_{\uparrow 0.87}$	91.38$_{\uparrow 3.93}$	96.20$_{\downarrow 0.65}$	90.24$_{\uparrow 3.57}$	79.44$_{\uparrow 0.52}$	87.86$_{\uparrow 0.78}$	87.42
MaAS	✓	82.32$_{\uparrow 3.78}$	91.13$_{\uparrow 3.68}$	98.08$_{\uparrow 1.23}$	89.65$_{\uparrow 2.98}$	80.25$_{\uparrow 1.33}$	89.57$_{\uparrow 2.49}$	88.50
ARG-Designer	✓	79.10$_{\uparrow 0.56}$	91.25$_{\uparrow 3.80}$	98.55$_{\uparrow 1.70}$	92.21$_{\uparrow 5.54}$	81.10$_{\uparrow 2.18}$	89.19$_{\uparrow 2.11}$	88.57
RADAR	✓	**83.66**$_{\uparrow 5.12}$	**92.51**$_{\uparrow 5.06}$	**98.81**$_{\uparrow 1.96}$	**93.26**$_{\uparrow 6.59}$	**82.84**$_{\uparrow 3.92}$	**91.28**$_{\uparrow 4.20}$	**90.32**

Experiments

- **Our proposed RADAR demonstrates a favorable trade-off, maintaining strong performance while exhibiting high token efficiency**

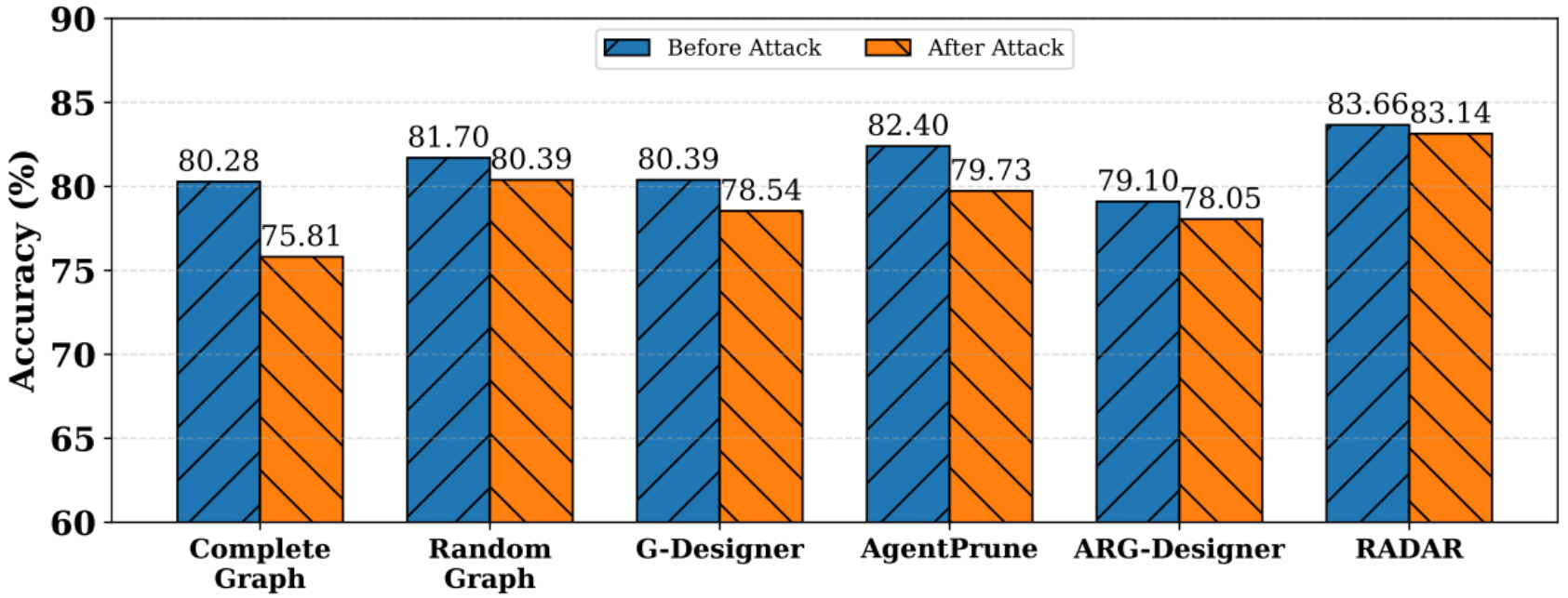




Experiments

- We inject system prompt attacks into two of the five agents in the collaboration framework

- RADAR exhibits exceptional resilience to adversarial perturbations, maintaining nearly identical performance before and after the attack



Conclusion

南京大学 NANJING UNIVERSITY

- We propose an iterative framework for multiagent collaboration topology design, which provides *high accuracy*, *low token consumption*, and *strong robustness* across diverse scenarios

- We employ a graph diffusion model that leverages graph effectiveness signals to iteratively construct collaboration topologies, *explicitly modeling redundancy* as part of the topology generation process

- We conduct extensive experiments on six benchmarks spanning *general reasoning*, *code generation* and *mathematical problem solving*, which collectively demonstrate the effectiveness of our method

Thanks
Q&A

Code & Data