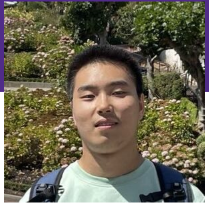
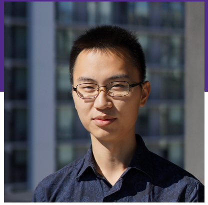
Can Language Models Solve Graph Problems in Natural Language?



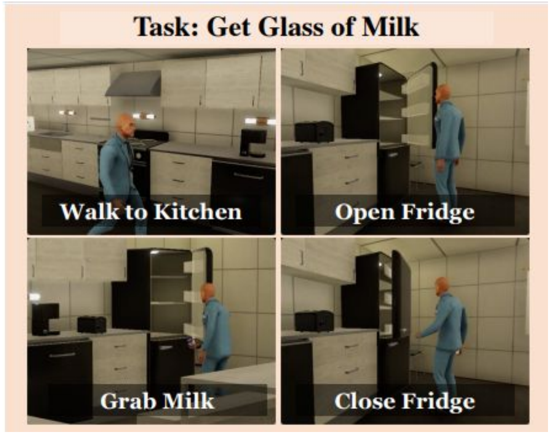


Heng Wang*[1] Shangbin Feng*[2] Tianxing He[2] Zhaoxuan Tan[3] Xiaochuang Han[2] Yulia Tsvetkov[2]

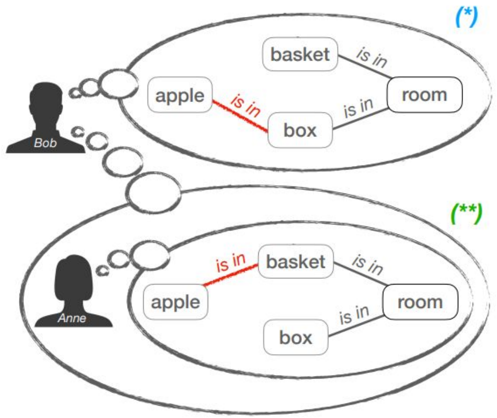
[1]Xi'an Jiaotong University, [2]University of Washington, [3]University of Notre Dame

LLMs are increasingly adopted for tasks with *implicit graph structures*

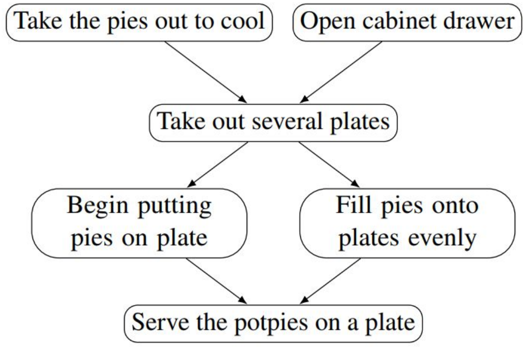
- Planning in robotics
- Updating belief state in theory-of-mind
- Structured commonsense reasoning



planning in robotics



belief state in theory-of-mind


Take the pies out to cool
Open cabinet drawer
Take out several plates
Begin putting pies on plate
Fill pies onto plates evenly
Serve the potpies on a plate

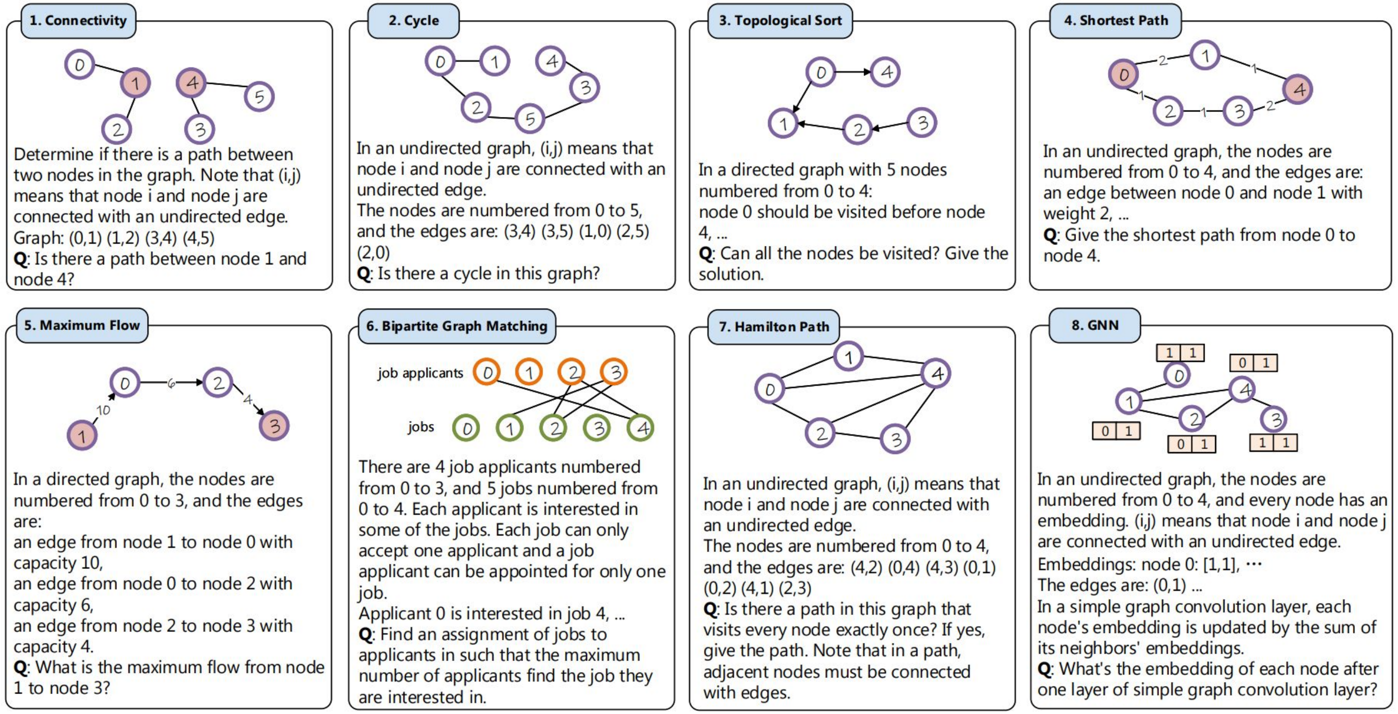
structured commonsense reasoning

LLMs are increasingly adopted for tasks with *implicit graph structures*

- LLMs have advanced the state-of-the-art on these tasks with structure implication.
- However, one underlying yet crucial question remains underexplored: *Are LLMs graph thinkers?*
- More concretely, are LLMs capable of mapping textual descriptions of graphs and structures to grounded conceptual spaces and performing basic operations?

The NLGraph Benchmark

- graph-based problem solving designed in natural language



1. Connectivity

Determine if there is a path between two nodes in the graph. Note that (i,j) means that node i and node j are connected with an undirected edge.
Graph: (0,1) (1,2) (3,4) (4,5)
Q: Is there a path between node 1 and node 4?

2. Cycle

In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge.
The nodes are numbered from 0 to 5, and the edges are: (3,4) (3,5) (1,0) (2,5) (2,0)
Q: Is there a cycle in this graph?

3. Topological Sort

In a directed graph with 5 nodes numbered from 0 to 4:
node 0 should be visited before node 4, ...
Q: Can all the nodes be visited? Give the solution.

4. Shortest Path

In an undirected graph, the nodes are numbered from 0 to 4, and the edges are:
an edge between node 0 and node 1 with weight 2, ...
Q: Give the shortest path from node 0 to node 4.

5. Maximum Flow

In a directed graph, the nodes are numbered from 0 to 3, and the edges are:
an edge from node 1 to node 0 with capacity 10,
an edge from node 0 to node 2 with capacity 6,
an edge from node 2 to node 3 with capacity 4.
Q: What is the maximum flow from node 1 to node 3?

6. Bipartite Graph Matching

There are 4 job applicants numbered from 0 to 3, and 5 jobs numbered from 0 to 4. Each applicant is interested in some of the jobs. Each job can only accept one applicant and a job applicant can be appointed for only one job.
Applicant 0 is interested in job 4, ...
Q: Find an assignment of jobs to applicants in such that the maximum number of applicants find the job they are interested in.

7. Hamilton Path

In an undirected graph, (i,j) means that node i and node j are connected with an undirected edge.
The nodes are numbered from 0 to 4, and the edges are: (4,2) (0,4) (4,3) (0,1) (0,2) (4,1) (2,3)
Q: Is there a path in this graph that visits every node exactly once? If yes, give the path. Note that in a path, adjacent nodes must be connected with edges.

8. GNN

In an undirected graph, the nodes are numbered from 0 to 4, and every node has an embedding. (i,j) means that node i and node j are connected with an undirected edge.
Embeddings: node 0: [1,1], ⋯
The edges are: (0,1) ...
In a simple graph convolution layer, each node's embedding is updated by the sum of its neighbors' embeddings.
Q: What's the embedding of each node after one layer of simple graph convolution layer?

The NLGraph Benchmark

- We employ a random graph generator to generate graphs and structures while controlling for network size, graph sparsity.

- 5,902 problems in the standard version, 29,370 problems in the deluxe version

- *Accuracy* (whether the True/False answer is correct, whether the proposed solution is valid) is the default evaluation metric, while the shortest path task, the maximum flow task, and the graph neural network task have additional *partial-credit* metrics (PC).

Experiment Setting

- Baselines: zero-shot prompting, few-shot in-context learning, chain-of-thought prompting, zero-shot chain-of-thought, least-to-most, and self-consistency.
- We also adopt random baseline for a fuller comparison.

- Model: text-davinci-003 as the default model, other LLMs (GPT-3.5-turbo, code-davinci-002, and GPT-4) are evaluated on part of the benchmark.

Results

- Key findings:
  - LLMs *do* posses preliminary graph thinking abilities
  - The benefit of advanced prompting methods diminishes with complex problems
  - While in-context learning is widely credited for teaching LLMs to learn from examples, it did not happen on complex graph reasoning tasks.
  - LLMs are (un)surprisingly vulnerable to spurious correlations in graph settings

LLMs *are* (Preliminary) Graph Thinkers

- We first find that on simple graph reasoning tasks, LLMs achieve impressive performance and demonstrate preliminary graph thinking abilities.

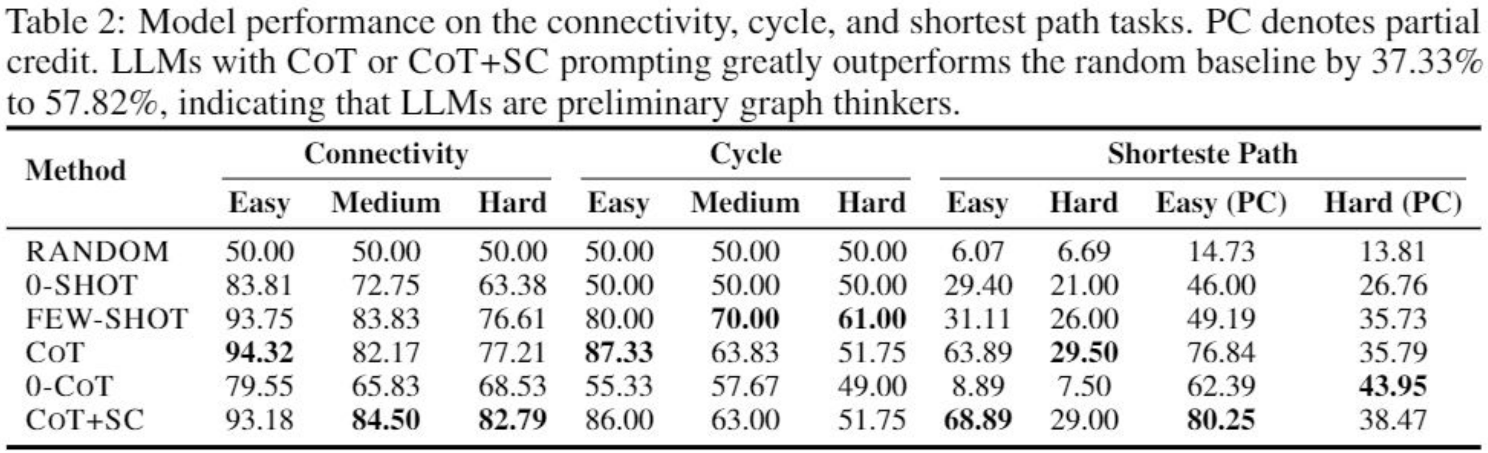
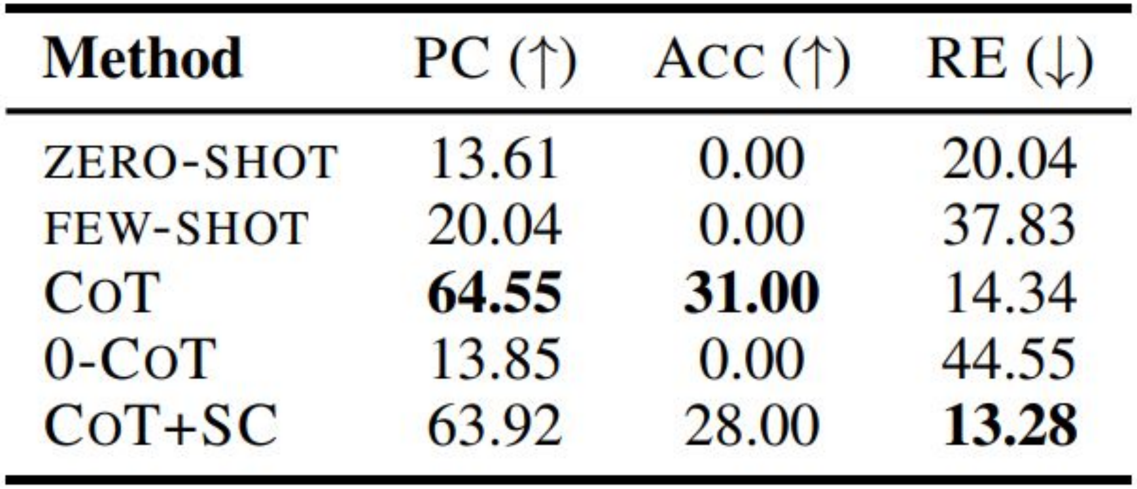


Table 2: Model performance on the connectivity, cycle, and shortest path tasks. PC denotes partial credit. LLMs with CoT or CoT+SC prompting greatly outperforms the random baseline by 37.33% to 57.82%, indicating that LLMs are preliminary graph thinkers.

Method	**Connectivity**			**Cycle**			**Shorteste Path**			
	Easy	**Medium**	**Hard**	**Easy**	**Medium**	**Hard**	**Easy**	**Hard**	**Easy (PC)**	**Hard (PC)**
RANDOM	50.00	50.00	50.00	50.00	50.00	50.00	6.07	6.69	14.73	13.81
0-SHOT	83.81	72.75	63.38	50.00	50.00	50.00	29.40	21.00	46.00	26.76
FEW-SHOT	93.75	83.83	76.61	80.00	**70.00**	**61.00**	31.11	26.00	49.19	35.73
CoT	**94.32**	82.17	77.21	**87.33**	63.83	51.75	63.89	**29.50**	76.84	35.79
0-CoT	79.55	65.83	68.53	55.33	57.67	49.00	8.89	7.50	62.39	**43.95**
CoT+SC	93.18	**84.50**	**82.79**	86.00	63.00	51.75	**68.89**	29.00	**80.25**	38.47

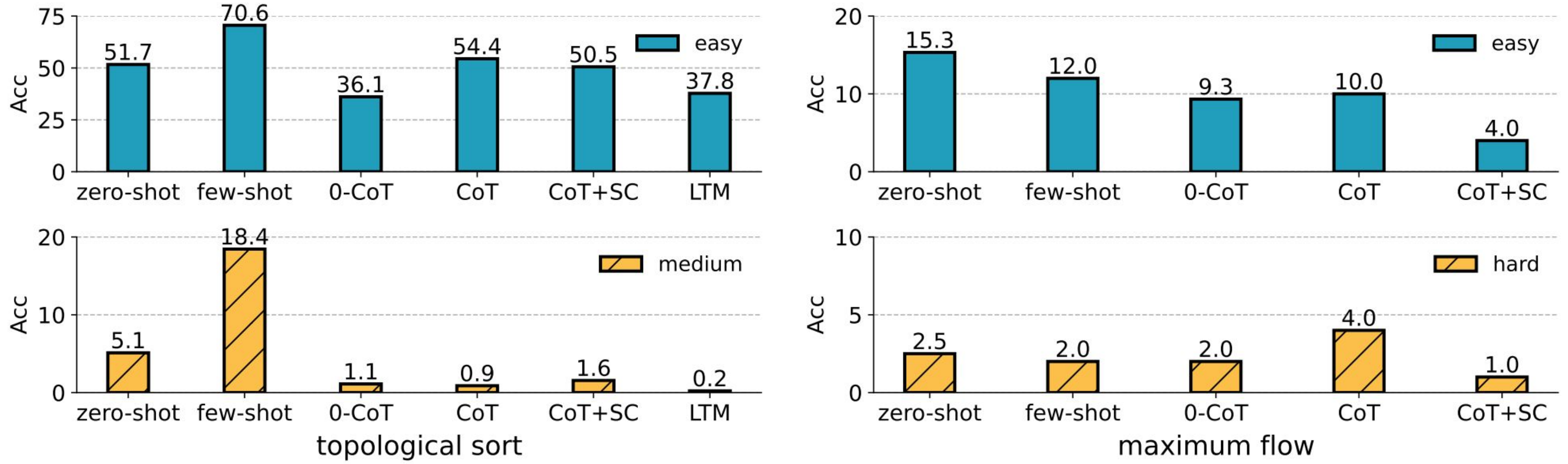
Advanced Prompting is Double-Edged

- As mentioned before, advanced prompting methods successfully improve performance on simple graph reasoning tasks.
- For the task of simulating graph neural networks, it is also the case.



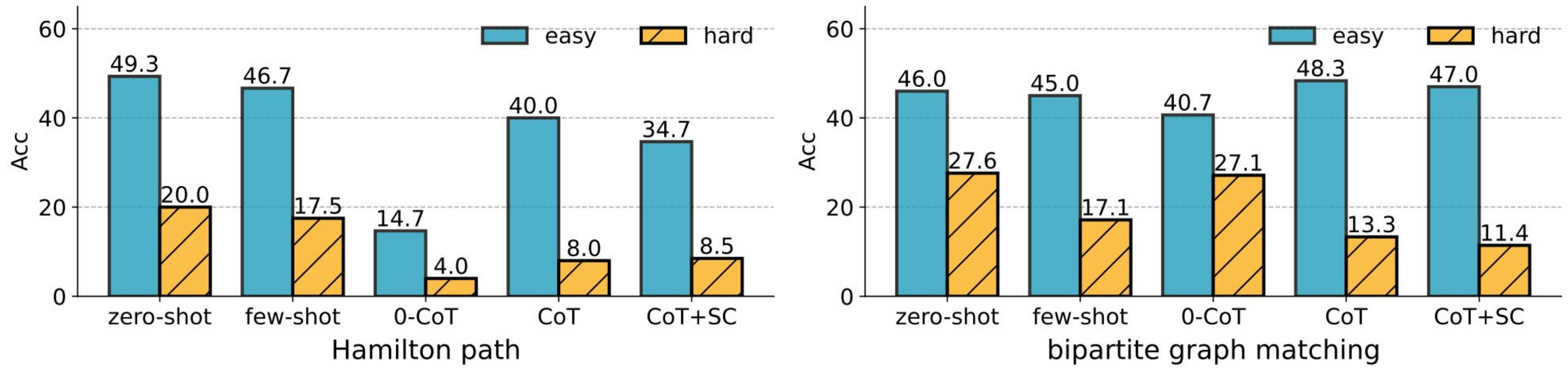
Method	PC (↑)	Acc (↑)	RE (↓)
ZERO-SHOT	13.61	0.00	20.04
FEW-SHOT	20.04	0.00	37.83
CoT	**64.55**	**31.00**	14.34
0-CoT	13.85	0.00	44.55
CoT+SC	63.92	28.00	**13.28**

Advanced Prompting is Double-Edged



For topological sort task and maximum flow task, few-shot prompting outperforms advanced prompting techniques.

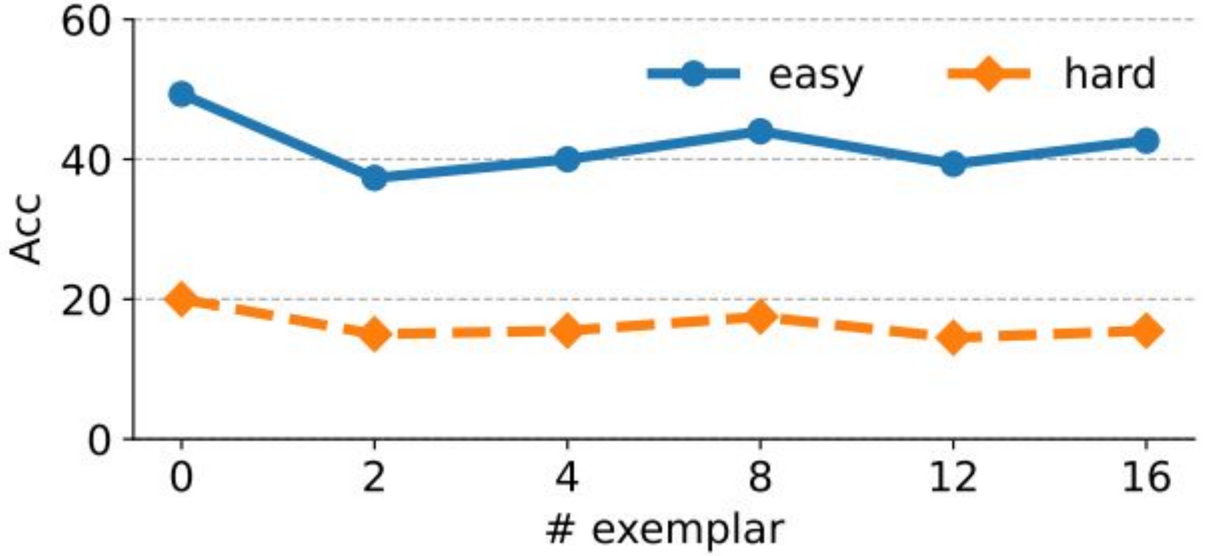
In-Context Learning could be Counterproductive



For Hamilton path task and bipartite graph matching task, zero-shot prompting consistently outperforms all other prompting techniques.

In-Context Learning could be Counterproductive

- We study whether increasing the number of in-context exemplars could help on the Hamilton path task.

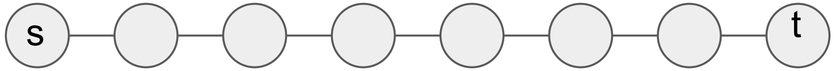


LLMs are (Un)surprisingly Brittle

- LLM may reach the correct answer depending on some spurious correlation
- To this end, we design two special cases for the connectivity task.

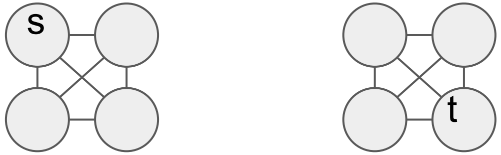
- Chain: 120 problems. All the graphs consist of multiple chains.

  Query nodes: at the two ends of a chain (connected).



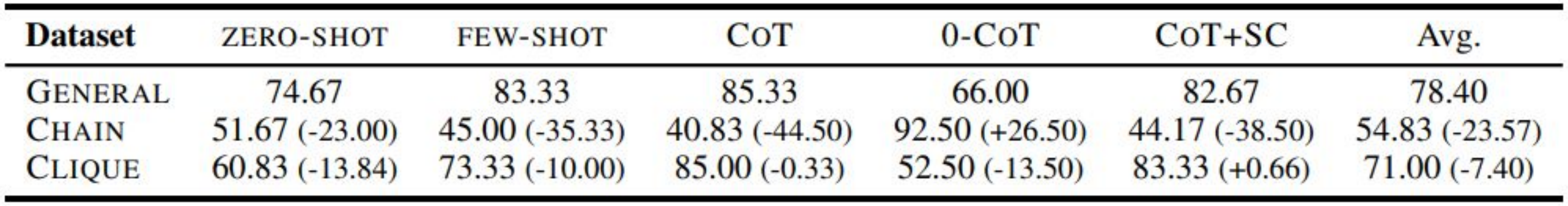
- Clique: 120 problems. All the graphs consist of several densely connected subgraphs.

  Query nodes: at two different subgraphs (unconnected).



LLMs are (Un)surprisingly Brittle

- Results:



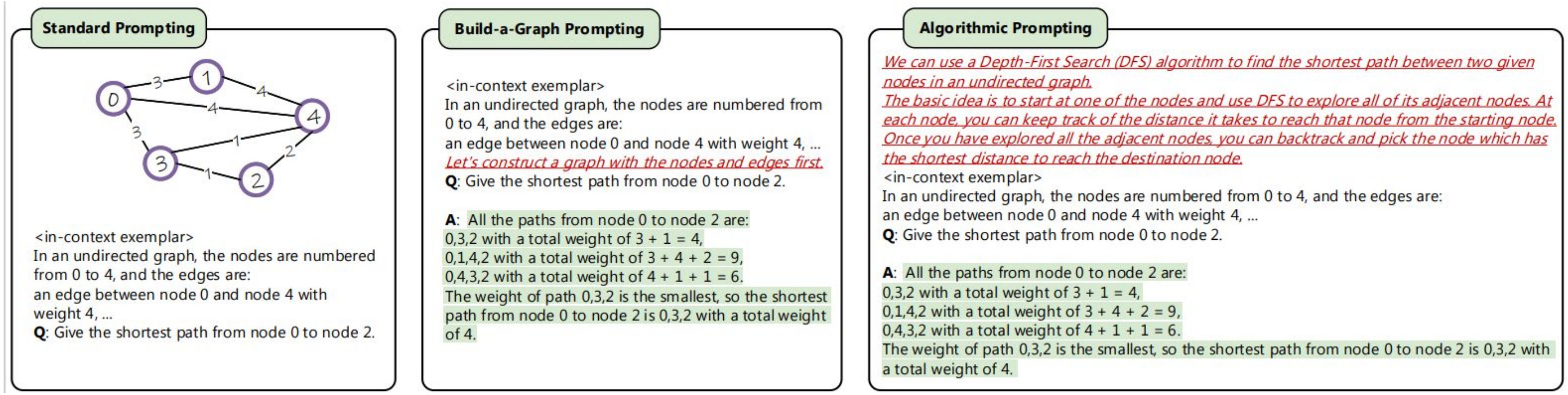
Dataset	ZERO-SHOT	FEW-SHOT	CoT	0-CoT	CoT+SC	Avg.
GENERAL	74.67	83.33	85.33	66.00	82.67	78.40
CHAIN	51.67 (-23.00)	45.00 (-35.33)	40.83 (-44.50)	92.50 (+26.50)	44.17 (-38.50)	54.83 (-23.57)
CLIQUE	60.83 (-13.84)	73.33 (-10.00)	85.00 (-0.33)	52.50 (-13.50)	83.33 (+0.66)	71.00 (-7.40)

Making Language Models Better Graph Thinkers

- We provide two simple prompting techniques that help improve the graph thinking ability of large language models. The two techniques can be used together with any other prompting techniques.

Making Language Models Better Graph Thinkers

- Build-a-Graph Prompting
- Algorithmic Prompting


0
1
2
3
4
3
4
4
3
1
2
1

Standard Prompting

<in-context exemplar>
In an undirected graph, the nodes are numbered from 0 to 4, and the edges are:
an edge between node 0 and node 4 with weight 4, ...
Q: Give the shortest path from node 0 to node 2.

Build-a-Graph Prompting

<in-context exemplar>
In an undirected graph, the nodes are numbered from 0 to 4, and the edges are:
an edge between node 0 and node 4 with weight 4, ...
Let's construct a graph with the nodes and edges first.
Q: Give the shortest path from node 0 to node 2.

A: All the paths from node 0 to node 2 are:
0,3,2 with a total weight of 3 + 1 = 4,
0,1,4,2 with a total weight of 3 + 4 + 2 = 9,
0,4,3,2 with a total weight of 4 + 1 + 1 = 6.
The weight of path 0,3,2 is the smallest, so the shortest path from node 0 to node 2 is 0,3,2 with a total weight of 4.

Algorithmic Prompting

We can use a Depth-First Search (DFS) algorithm to find the shortest path between two given nodes in an undirected graph.
The basic idea is to start at one of the nodes and use DFS to explore all of its adjacent nodes. At each node, you can keep track of the distance it takes to reach that node from the starting node. Once you have explored all the adjacent nodes, you can backtrack and pick the node which has the shortest distance to reach the destination node.
<in-context exemplar>
In an undirected graph, the nodes are numbered from 0 to 4, and the edges are:
an edge between node 0 and node 4 with weight 4, ...
Q: Give the shortest path from node 0 to node 2.

A: All the paths from node 0 to node 2 are:
0,3,2 with a total weight of 3 + 1 = 4,
0,1,4,2 with a total weight of 3 + 4 + 2 = 9,
0,4,3,2 with a total weight of 4 + 1 + 1 = 6.
The weight of path 0,3,2 is the smallest, so the shortest path from node 0 to node 2 is 0,3,2 with a total weight of 4.

Making Language Models Better Graph Thinkers

- Results:

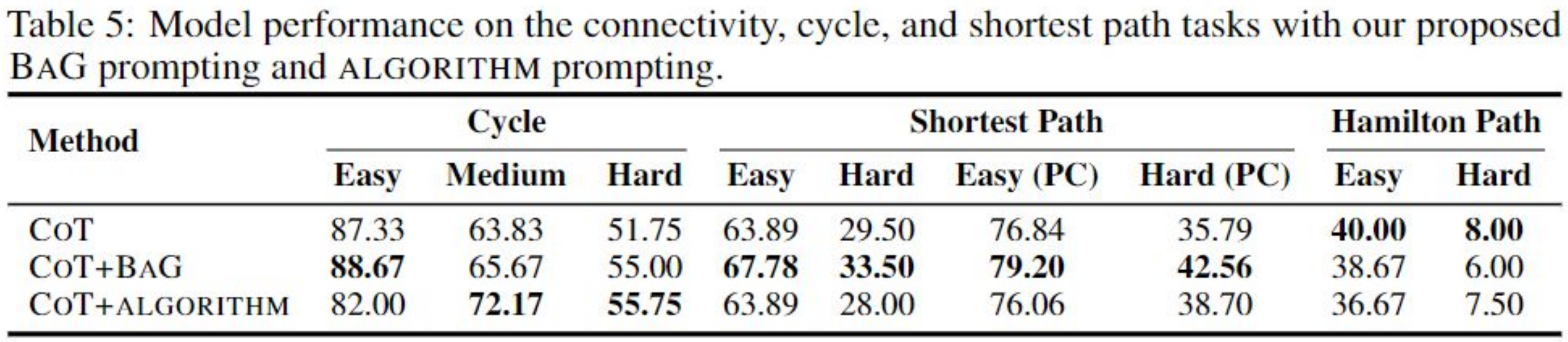


Table 5: Model performance on the connectivity, cycle, and shortest path tasks with our proposed BAG prompting and ALGORITHM prompting.

Method	Cycle			Shortest Path				Hamilton Path	
	Easy	Medium	Hard	Easy	Hard	Easy (PC)	Hard (PC)	Easy	Hard
CoT	87.33	63.83	51.75	63.89	29.50	76.84	35.79	**40.00**	**8.00**
CoT+BAG	**88.67**	65.67	55.00	**67.78**	**33.50**	**79.20**	**42.56**	38.67	6.00
CoT+ALGORITHM	82.00	**72.17**	**55.75**	63.89	28.00	76.06	38.70	36.67	7.50

- The most complicated graph reasoning problems remain an open research question

Thank you!

Paper Data

- NLGraph benchmark: a comprehensive testbed for graph reasoning
- Key findings:
  - LLMs *do* posses preliminary graph thinking abilities
  - The benefit of advanced prompting methods diminishes with complex problems
  - While in-context learning is widely credited for teaching LLMs to learn from examples, it did not happen on complex graph reasoning tasks.
  - LLMs are (un)surprisingly vulnerable to spurious correlations in graph settings