UNIVERSITÄT DES SAARLANDES

NEURO EXPLICIT MODELS

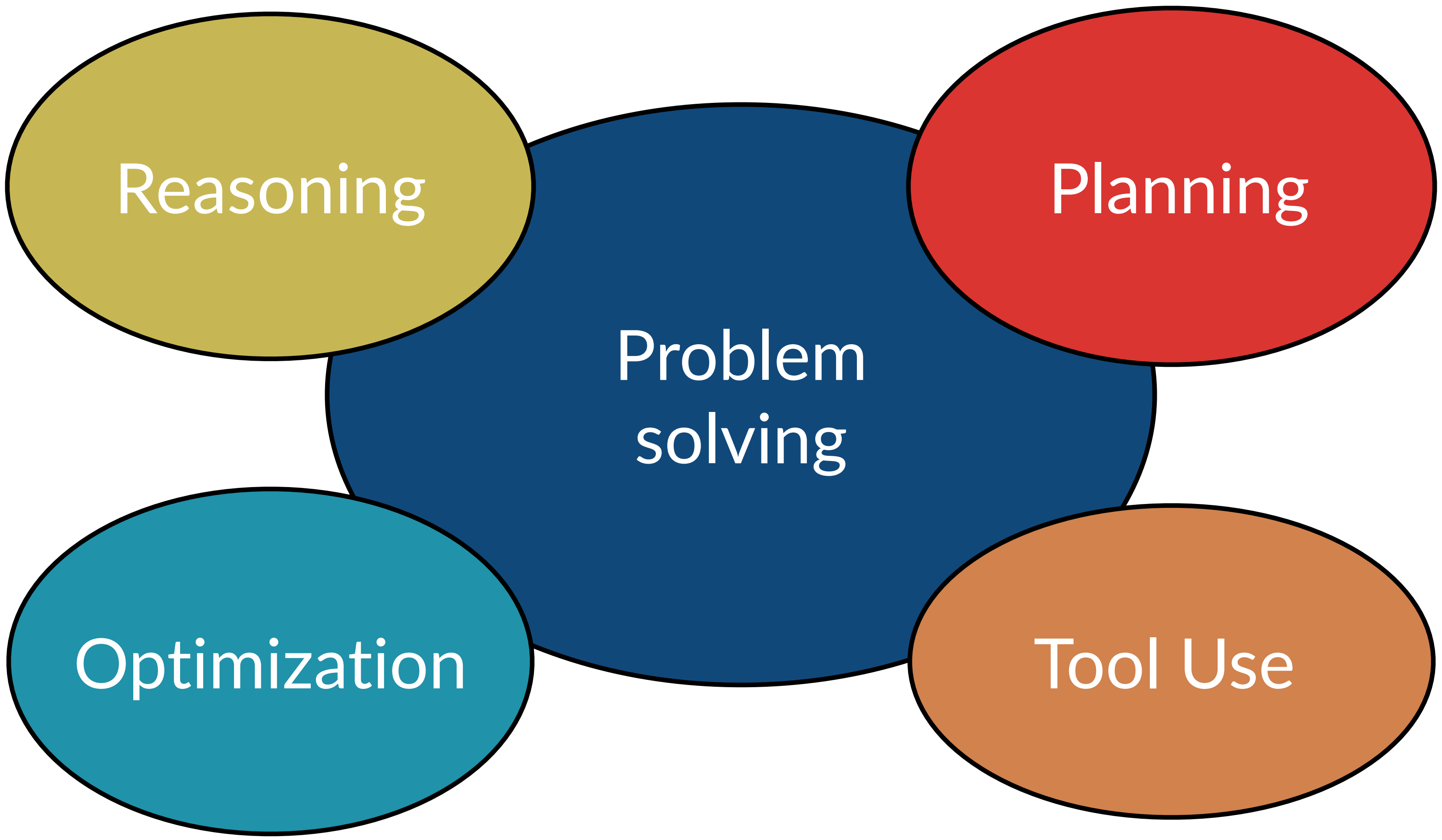
DEPARTMENT OF LANGUAGE SCIENCE AND TECHNOLOGY
SAARLAND UNIVERSITY

Solving complex problems with large language models

Alexander Koller
Saarland University

30 April 2025

Solving complex problems



LLMs for Problem Solving

```
\ Problem instance with n cities
param n := 5;

\ Variables: x[i, j] = 1 if the tour goes from city
var x{i in 1..n, j in 1..n} binary;
var u{i in 2..n} >= 1, <= n-1;

\ Objective: Minimize total travel cost
minimize total_cost: sum{i in 1..n, j in 1..n} c[i, j]

\ Constraints: Every city has exactly one incoming and
subject to in_degree{j in 1..n}: sum{i in 1..n, i != j}
subject to out_degree{i in 1..n}: sum{j in 1..n, j != i
```

Step 3.3: Find the Optimal Tour

We evaluate all 24 possible tours:

1. **(1 → 2 → 3 → 4 → 5 → 1):**
   Cost = 10 + 35 + 12 + 18 + 25 = **100**
2. **(1 → 2 → 3 → 5 → 4 → 1):**
   Cost = 10 + 35 + 8 + 18 + 20 = **91**
3. **(1 → 2 → 4 → 3 → 5 → 1):**
   Cost = 10 + 30 + 12 + 8 + 25 = **85**

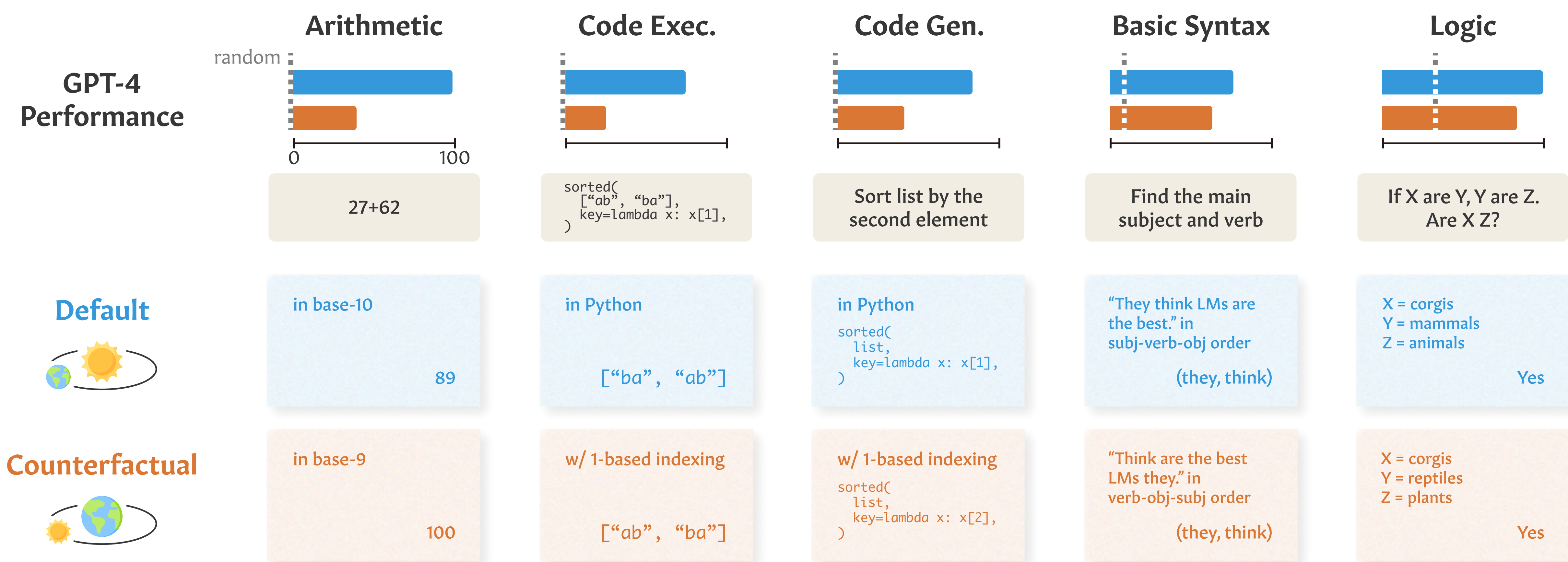

```
\ Problem instance with n cities
param n := 5;

\ Variables: x[i, j] = 1 if the tour goes from city
var x{i in 1..n, j in 1..n} binary;
var u{i in 2..n} >= 1, <= n-1;

\ Objective: Minimize total travel cost
minimize total_cost: sum{i in 1..n, j in 1..n} c[i, j]

\ Constraints: Every city has exactly one incoming and
subject to in_degree{j in 1..n}: sum{i in 1..n, i != j}
subject to out_degree{i in 1..n}: sum{j in 1..n, j != i
```

Reasoning or reciting?



"Reasoning": system actually solves the problem, generalize to arbitrary instances.

"Reciting": system replicates solutions (or solution methods) from training data, expect worse generalization.

(Wu et al. 2024)

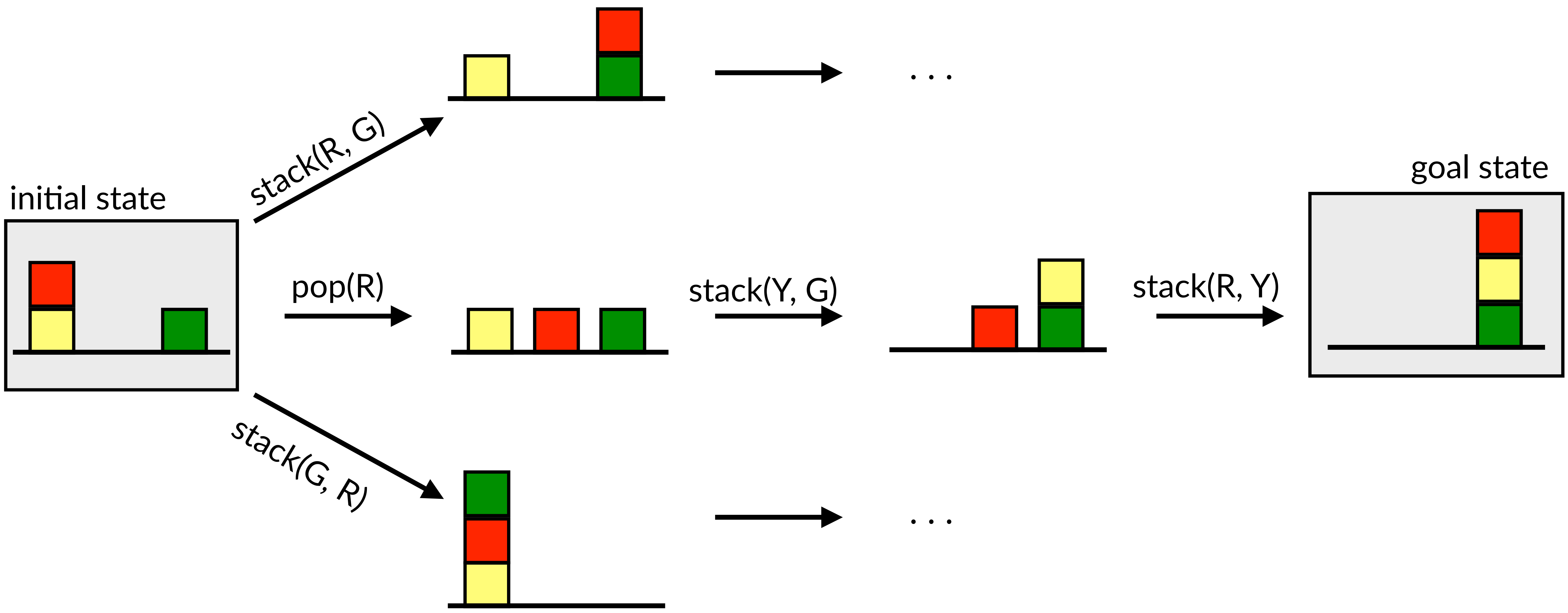
Outline

1. Planning
2. NP-hard optimization problems
3. Collaborative problem solving

#1
Planning

Planning

Example: Blocksworld (simplified)



Planning with LLMs

```
========================================
I am playing with a set of blocks where I need to arrange the blocks into stacks. Here are the
↪  actions I can do

Pick up a block
Unstack a block from on top of another block
Put down a block
Stack a block on top of another block

I have the following restrictions on my actions:
I can only pick up or unstack one block at a time.
I can only pick up or unstack a block if my hand is empty.

[. . .]

[STATEMENT]
As initial conditions I have that, the red block is clear, the blue block is clear, the yellow
↪  block is clear, the hand is empty, the blue block is on top of the orange block, the red block
↪  is on the table, the orange block is on the table and the yellow block is on the table.
My goal is to have that the orange block is on top of the blue block.

My plan is as follows:

[PLAN]
unstack the blue block from on top of the orange block
put down the blue block
pick up the orange block
stack the orange block on top of the blue block
[PLAN END]
```

(One-shot prompting strategy of Valmeekam et al. 2023)

"LLMs can't plan"

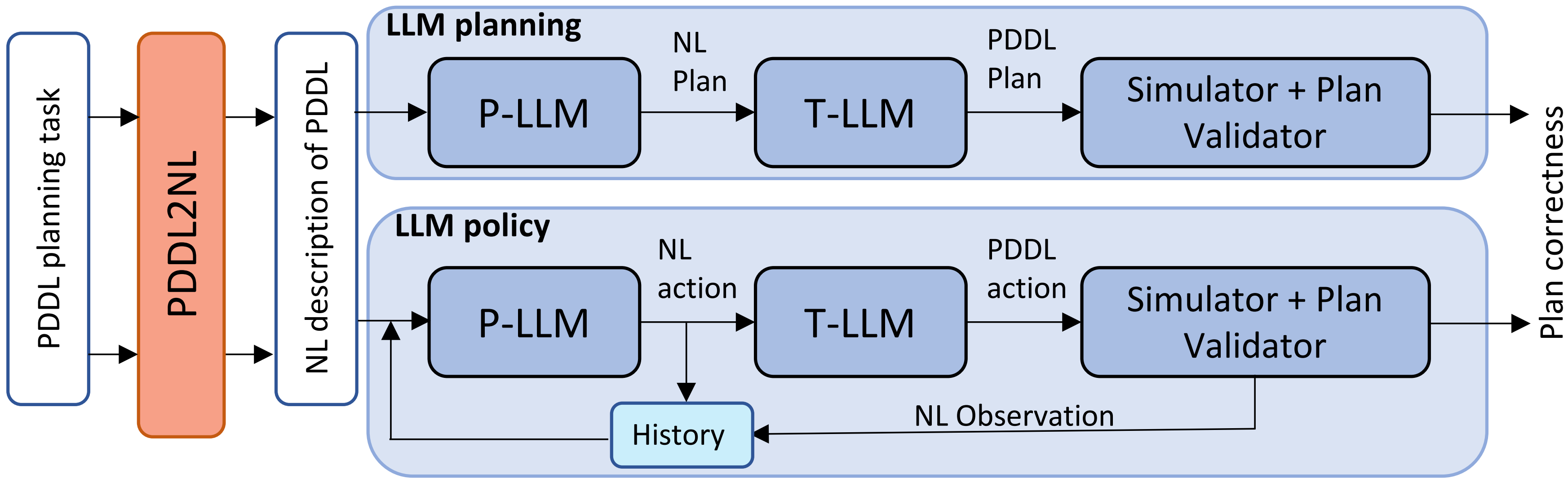
Domain	Method	Instances correct					
		GPT-4o	GPT-4-Turbo	Claude-3-Opus	LLaMA-3 70B	Gemini Pro	GPT-4
Blocksworld (BW)	One-shot	170/600 (28.33%)	138/600 (23%)	289/600 (48.17%)	76/600 (12.6%)	68/600 (11.3%)	206/600 (34.3%)
	Zero-shot	213/600 (35.5%)	241/600 (40.1%)	356/600 (59.3%)	205/600 (34.16%)	3/600 (0.5%)	210/600 (34.6%)
Mystery BW (Deceptive)	One-shot	5/600 (0.83%)	5/600 (0.83%)	8/600 (1.3%)	15/600 (2.5%)	2/500 (0.4%)	26/600 (4.3%)
	Zero-shot	0/600 (0%)	1/600 (0.16%)	0/600 (0%)	0/600 (0%)	(0/500) (0%)	1/600 (0.16%)

"Mystery Blocksworld": all symbols replaced by random strings.
Poor performance of LLMs on mystery domains is evidence against reasoning (but could be seen as evidence of useful world knowledge).

(PlanBench, Valmeekam et al. 2023; table from Kambhampati et al. 2024)

… or can they?

- Let's evaluate on more planning domains:
  Automatically translate all IPC benchmark domains into English.
- Let's evaluate more prompting strategies:
  Include chain-of-thought, ReAct (Yao et al. 2023).
- Obvious that transformers without CoT will not solve PSPACE-complete problem in quadratic time.



(AutoPlanBench: Stein, ..., K. ICAPS 2025)

Prompting strategies

LLM planning — No thoughts (Basic)

```
Input: My goal is that in the end
       #GOAL#
       My current initial situation
       is as follows:
       #OBJECTS + INITIAL STATE#
Model: #NL ACTION 1#
       #NL ACTION 2#
       ...
       [PLAN END]
```

LLM policy — No thoughts (ACT)

```
Input: My goal is that in the end #GOAL#
       My current initial situation is
       as follows:
       #OBJECTS + INITIAL STATE#
Model: #NL ACTION 1#
Input: #OBSERVATION FROM ENVIRONMENT#
Model: #NL ACTION 2#
Input: #OBSERVATION FROM ENVIRONMENT#
...
Model: You are finished
```

LLM planning — Thoughts (COT)

```
Input: My goal is that in the end
       #GOAL#
       My current initial situation
       is as follows:
       #OBJECTS + INITIAL STATE#
       Let's think step by step
Model: Think: #Thought 1#
       Instruction: #NL ACTION 1#
       Think: #Thought 2#
       Instruction: #NL ACTION 2#
       ...
       Think: #Thought N#
       Instruction: You are finished
       [PLAN END]
```

LLM policy — Thoughts (ReAct)

```
Input: My goal is that in the end #GOAL#
       My current initial situation is
       as follows:
       #OBJECTS + INITIAL STATE#
Model: Think: #Thought 1#
       Instruction: #NL ACTION 1#
Input: #OBSERVATION FROM ENVIRONMENT#
Model: Think: #Thought 2#
       Instruction: #NL ACTION 2#
Input: #OBSERVATION FROM ENVIRONMENT#
...
Model: Think: #Thought N#
       Instruction: You are finished
```

(AutoPlanBench: Stein, ..., K. ICAPS 2025)

Results on generated problem instances

Domains	LLM-based Approaches								Symb. Baselines			
	PDDL2NL				Tpl		PDDL					
	Bas	CoT	Act	ReA	Bas	Act	Bas	Act	rnd	BrFS	lmc	ff
Block.	13	15	17	**18**	8	12	13	14	1	20	20	20
↳Valm23	15	14	17	18								
Logistics	5	10	16	**19**	2	7	6	15	0	20	20	20
↳Valm23	3	5	17	12								
Depot	0	5	5	13	0	5	0	3	0	20	20	20
↳Valm23	3	6	6	**15**								
Ferry	7	12	14	**18**	0	13	8	17	0	20	20	20
Floortile	0	0	0	0	0	0	0	0	0	18	20	20
Goldm.	1	1	3	1	1	3	1	**4**	0	20	20	20
Grid	8	6	16	**18**	1	12	6	12	0	20	20	20
Grippers	9	17	17	**20**	10	**20**	12	19	0	20	20	20
Movie	**20**	**20**	**20**	**20**	**20**	**20**	**20**	**20**	3	20	20	20
Rovers	0	0	**18**	**18**	1	17	1	11	1	20	20	20
Satellite	14	16	**20**	**20**	11	18	14	18	0	20	20	20
Visitall	19	19	**20**	**20**	18	**20**	**20**	**20**	8	20	20	20
Σ (240)	96	121	166	**185**	72	147	101	153	13	238	240	240
Further scaled selected domains:												
Block.	3	3	12	**14**	0	6	1	4	0	12	19	20
Ferry	0	0	7	15	0	9	0	**17**	0	8	13	20
Gripper	17	12	**20**	19	16	**20**	16	**20**	0	10	8	20
Visitall	9	2	16	**18**	7	17	14	16	1	10	18	20
Σ (80)	29	17	55	**66**	23	52	31	57	1	40	58	80

(AutoPlanBench: Stein, ..., K. ICAPS 2025)

Results on IPC benchmarks

Domains	PDDL2NL		Symbolic Baselines			
	CoT	ReA	rnd	BrFS	lmc	ff
barman11/14 (10)	0	3	0	10	3	10
blocks00 (35)	3	22	0	21	28	35
childsnack14 (16)	6	15	0	0	0	16
gripper98 (19)	12	19	0	7	6	19
logistics98/00 (29)	1	28	0	12	21	29
movie98 (29)	29	29	0	29	29	29
rovers06 (6)	1	5	0	6	6	6
satellite02 (5)	1	4	0	5	5	5
transport08/11 (31)	3	23	0	18	19	31
visitall11/14 (13)	6	13	0	13	13	13
others (482 in 27 domains)	4	18	1	291	311	482
Σ (675)	66	179	1	412	441	675

We excluded the remaining four IPC domains for cost reasons.

(AutoPlanBench: Stein, ..., K. ICAPS 2025)

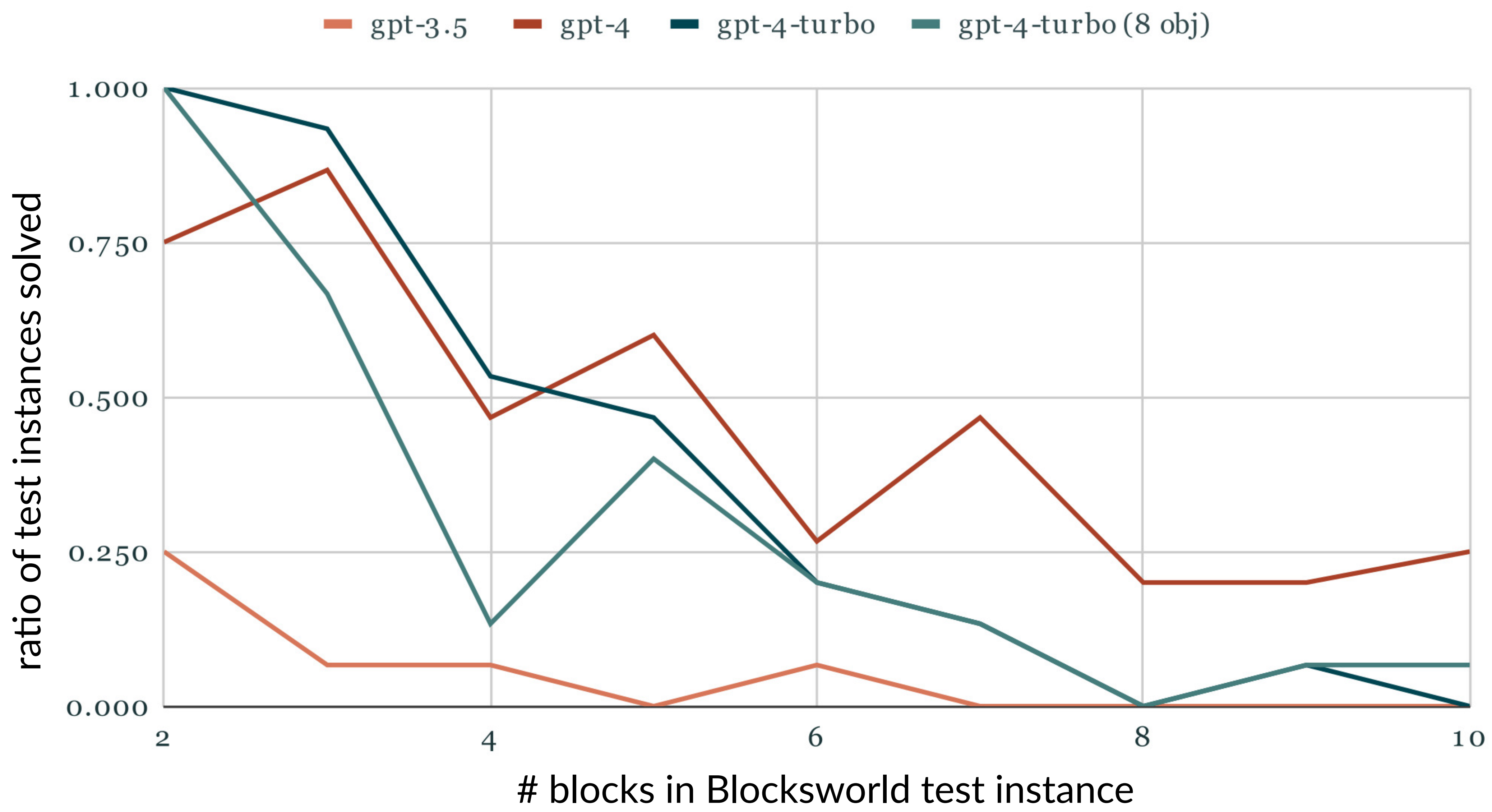
Takeaways

- **LLM policies**, which can observe the environment after each action, can be **much more accurate** than methods that predict the whole plan in one go.
  - This comes at the cost of having to actually execute these actions, which may be impractical in many domains.

- Really important to **evaluate LLMs across multiple domains**. ReAct outperforms symbolic planners on some domains, is unpredictably bad on others.

- **Length generalization** is a challenge: transformers are not good at generalizing to larger or more complex instances than their training data/few-shot examples (e.g. Yao & K. EMNLP 2022).

Length generalization



MSc thesis Lynn Zhou, 2024; in-context examples have four blocks

#2
NP-hard optimization problems

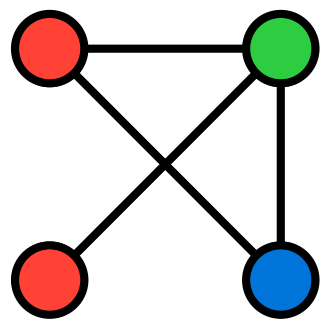
Everyday Optimization Problems

Planning is PSPACE-complete, results are messy. More focused evaluation of complex problem solving?

Many everyday problems are NP-hard optimization problems. Can LLMs solve them?

Reasoning vs. reciting: Textbook form easier than everyday versions or not?

Hard Everyday Optimization Problems



Textbook problem (GRAPH-COLORING)
Given an undirected graph G = (V, E), assign colors to the nodes such that no two adjacent nodes have the same color. Use as few colors as possible.

Costumed problem (💔 Parties With Exes)
Your birthday is coming up, and you want to celebrate with all your friends. You do not want people who used to be in a relationship at the same party.
How many parties do you need?

Inverted problem
Given an undirected graph G = (V, E), assign colors to the nodes such that no two *non-adjacent* nodes have the same color. Use as few colors as possible.

(Duchnowski, Pavlick, K., submitted)

Evaluation

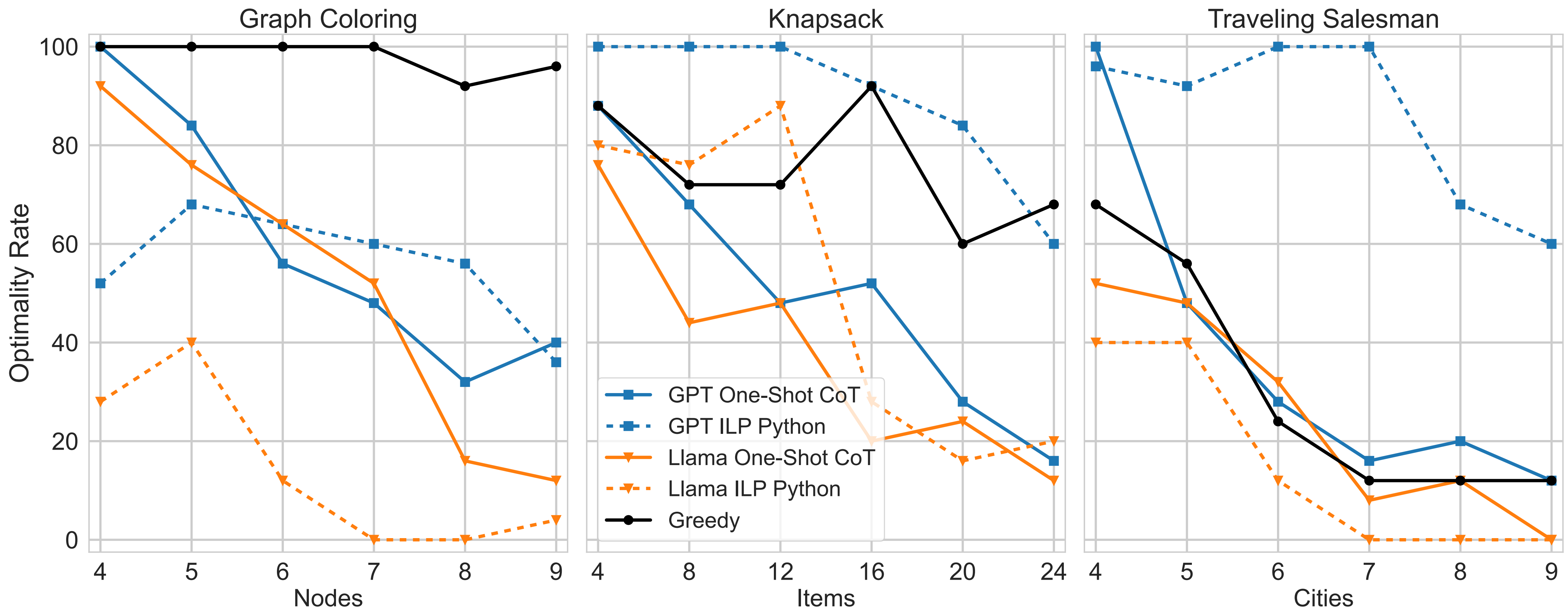
- EHOP dataset: 3 NP-hard problems x 4 costumes x inverted?
  25 random instances for each of 6 instance sizes.

- GPT-4o and Llama-3.1-70B-Instruct

- Various prompting techniques, including ILP-Python:
  Use LLM to translate NL problem into linear program, then
  use optimal LP solver to find optimal solution.

(Duchnowski, Pavlick, K., submitted)

Scaling to larger instances is hard



Observe how rarely CoT beats the greedy heuristics.

(Duchnowski, Pavlick, K., submitted)

Textbook is easier than "everyday" variants

Problem	Variant	One-Shot	Zero-Shot CoT	One-Shot CoT	ILP Python	Greedy
🖌️ GCP	Textbook	42.0	60.7	60.0	56.0	98.0
	Inverted	−39.3	−59.4	−59.3	−41.3	
	Costumed	−6.2	−6.5	−4.7	−43.8	
🎒 KSP	Textbook	22.7	48.0	50.0	89.3	75.3
	Inverted	+4.6	+2.7	−4.7	−0.6	
	Costumed	−2.0	−1.8	−2.2	−7.5	
✈️ TSP	Textbook	34.7	31.3	37.3	86.0	30.7
	Inverted	−20.7	−14.0	−9.3	−10.7	
	Costumed	−8.3	−1.7	−9.1	−37.1	

(Duchnowski, Pavlick, K., submitted)

Takeaways

- LLM solvers methods **do not scale well** to larger instances. Neurosymbolic "ILP-Python" method works best overall.

- All methods are **vulnerable to costuming** and especially inversion. LLMs adapt solution paths for frequent textbook problems, rather than performing general-purpose problem solving.

- We did not try to help the LLM by **specifying the base problem**. This would not be feasible in the real world either.

- Initial results with **reasoning models** (DeepSeek-R1): closes the gap, because it seems to recognize the base problem in a thought.

RTG Neuroexplicit Models of Language, Vision, Action

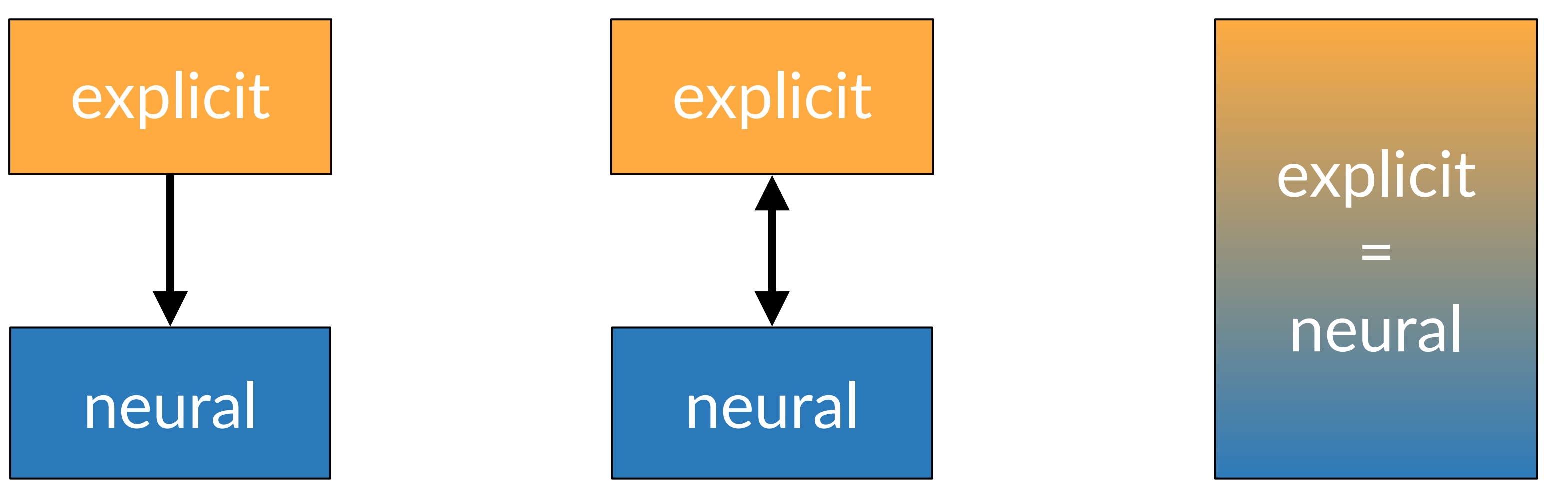
NEURO EXPLICIT MODELS

UNIVERSITÄT DES SAARLANDES | max planck institut informatik | MAX PLANCK INSTITUTE FOR SOFTWARE SYSTEMS | CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY | dfki ai

Goal 1: Overcome limitations of purely neural models, especially with respect to generalization, robustness, and interpretability.

Goal 2: Investigate theoretical and practical principles of designing effective neuroexplicit models.

Funds 24 PhD students from 2023-2028. Including 13 PIs and ~20 associated members, is the largest research center on neurosymbolic models in Europe. See http://www.neuroexplicit.org/ for more information.



#3
Collaborative problem-solving in dialogue

Collaborative problem solving

Please help me plan a two-week vacation in the Pacific Northwest. I am traveling with my wife and our two toddlers.

Okay! What's your approximate budget?

Let's not spend more than $200 per night on hotels and AirBnBs.

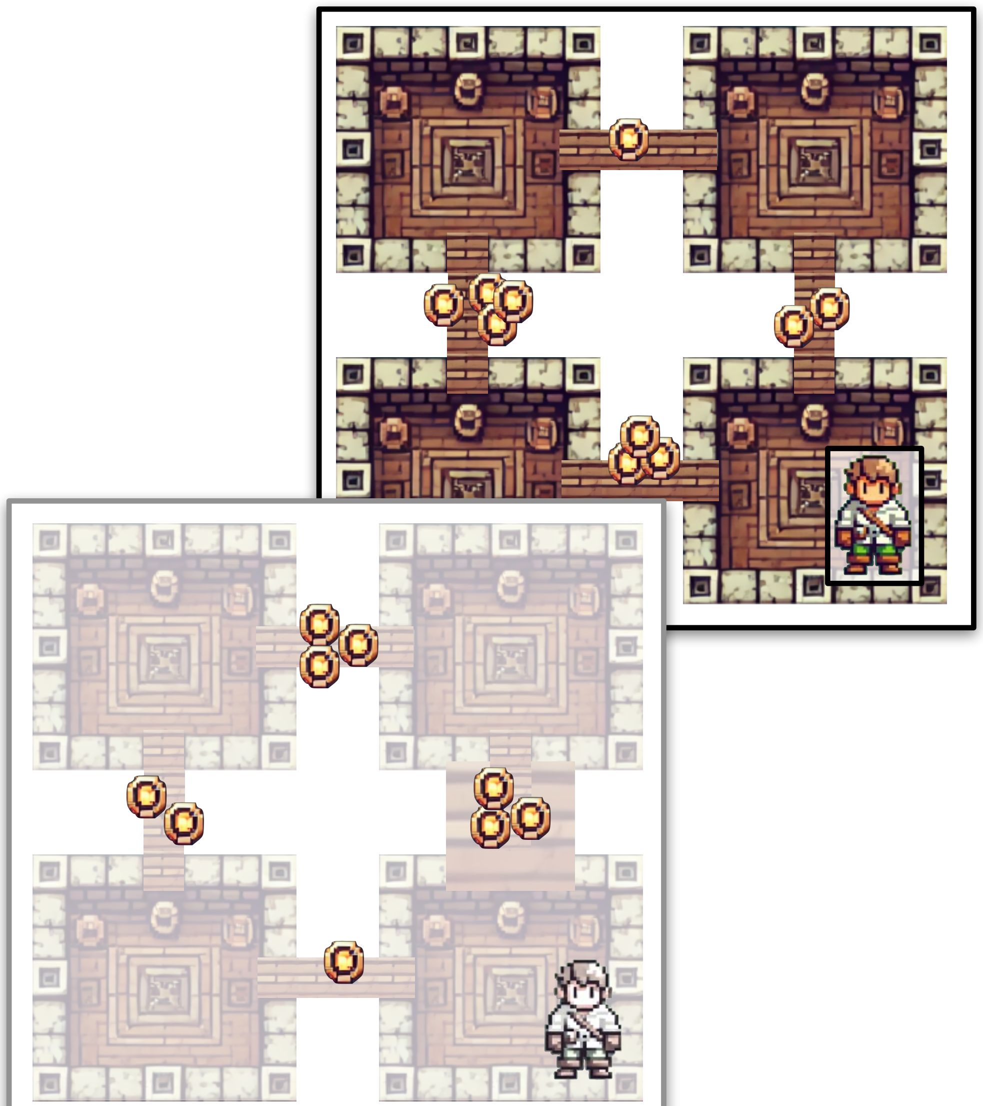
Great. Here's a first proposal.

Let me know what you like and dislike about it, and we'll fix it together.

The railway museum is a great idea, but let's make sure we're not driving for more than two hours each day.

Two-player Traveling Salesman

Joint solving of everyday NP-hard problems as platform for observing and modeling *collaborative* problem solving.



Shall we move to the left?

I'd prefer to go up instead.

But I'd really like to avoid the top hallway if we can.

...

(Jeknić, Schlangen, K., SIGDIAL 2024; Jeknić & K., submitted)

Baseline agent

[Thought]: Since this is the beginning of the game, I'll look for the best path for myself and see what the options are. I am in the light world and have the
fol
init
(K)
mi

[Ac

[M
livi
wo
gh

[Thought]: From the living room, I can also collect 6 ghost coins if we go to the bathroom, and I can collect 1 ghost coin if we head to the kitchen. If we both go to the bathroom, we would collect a total of 12 coins (6+6), and if we go to
th
th

[A

[M
th
gh
to

[Thought]: The user's response indicates they can collect the same number of ghost coins as I can collect light coins on the path to the bathroom (B), giving us a total of 12 coins. This is the optimal choice for us both from the living room, maximizing our combined coin collection at the start.

[Action]: agree(L to B)

[Message]: That sounds perfect! Going to the bathroom first gives us the maximum number of coins, with 12 in total. Let's start by heading to the bathroom!

(Self-play of GPT-4o baseline agent on a pair of six-node graphs)

Neurosymbolic agents

	Baseline	Grounding	State-tracking	Problem-solving
Agent WSR	✓	✓	✓	✓
Partner WSR		✓	✓	✓
Action History	✓	✓		
Visited			✓	✓
Remaining			✓	✓
IBP				✓

Agent WSR

```
[["L", "E", 6],
 ["L", "B", 4],
 ["L", "K", 2], ... ]
```

Partner WSR

```
[["L", "B", 5],
 ["B", "K", 3],
 ["B", "A", 6], ... ]
```

Visited

```
["L", "B", "K"]
```

Remaining

```
["L", "E", "A", "C"]
```

Action History

```
[suggest(["L", "C"]),
 agree-inner(["L", "C"]),
 ask("coin distribution
      between L and C")]
```

Intermediate Best Path

```
["L", "B", "K", "A", "E",
 "C", "L"]
```

Agent tends to confuse own graph and what it knows about the other agent's graph

→ *track symbolically what we know about other player's graph*

Agent tends to generate invalid paths

→ *track of partial path that has been agreed on so far*

Results

	Identical	Correct	Optimal
Baseline	99	71	28
Grounding	100	65	25
State-tracking	99	86	17
Problem-solving	98	98	45

Improved agent can negotiate best solution

User: : 👍 Great, let's head to the kitchen next! After that, would going to the children's room work for you? I can collect 6 light coins on the way. There are also 3 coins to the empty room. How does that match up with your best path from the kitchen?

Bot: : While going to the children's room from the kitchen would only yield 1 ghost coin for me, I can gather 6 ghost coins if we go to the empty room instead. How would this path work for you? Would you consider it given your coin distribution, or do you think the previous suggestion aligns better with your own strategy going forward? Could you also let me know what your best path forward from the empty room would be?

User: : Got it, let's head to the empty room then! From there, I can gather 6 light coins by heading to the attic. How does this fit with your strategy moving from the empty room?

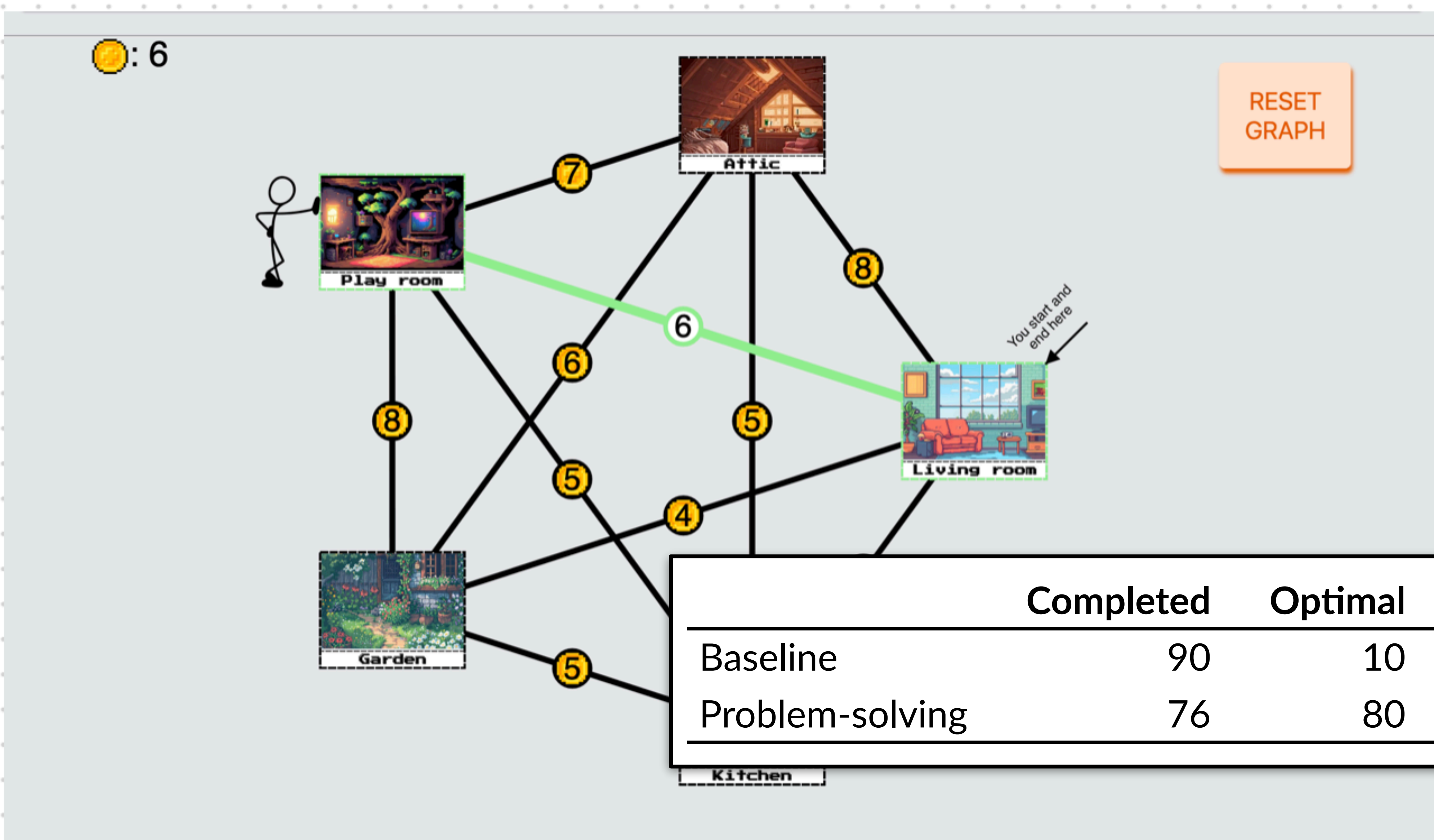
Bot: : Great, let's head to the empty room, and then from there, proceed to the attic. Going to the attic works perfectly for me, as I can collect 5 ghost coins along that path. That moves us forward nicely. Could you let me know about the light coins between the attic and the children's room? This should be our next stop based on our progress so far.

User: : Perfect, let's head to the attic next! I can collect 6 light coins on our way there. For the following move, going to the children's room sounds great as well – there are 3 light coins on that path. This should set us up well to complete our journey! Let's continue this way.

Bot: : Perfect, let's make our way to the children's room. I can collect 6 ghost coins on the way there, which aligns well with your path. Once we're in the children's room, heading back to the living room will complete our journey and allow us to finish the game by visiting all rooms. I'll receive 4 ghost coins on that final path. Looks like we're on the right track together – let's do this and wrap up our journey!

(self-play of two "problem-solving" agents with GPT-4o)

Evaluation with human subjects



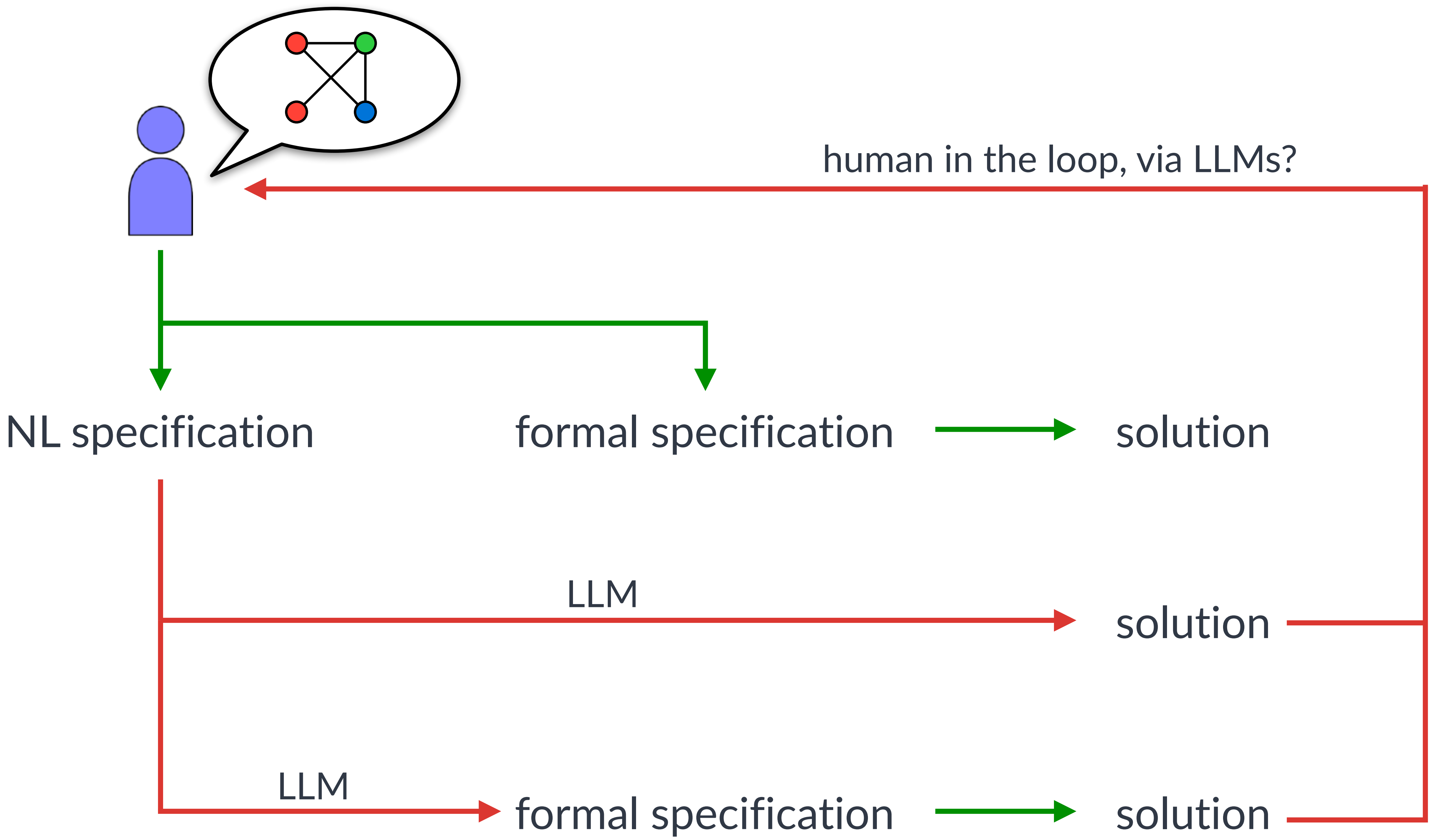
	Completed	Optimal	Score >90%
Baseline	90	10	25
Problem-solving	76	80	32

(using the Slurk platform, Götze et al. 2022)

Takeaways

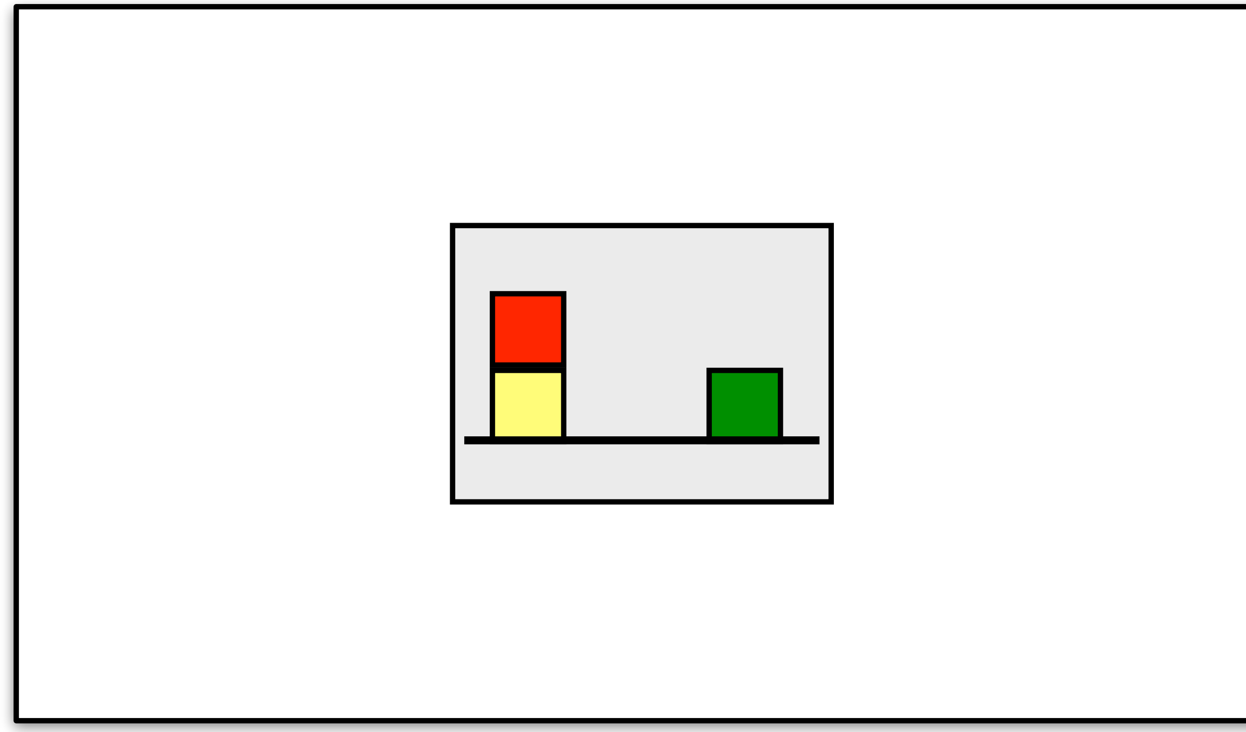
- Baseline CoT system **struggles to maintain goal-oriented dialogue** to help user solve their problem.
- Keeping key information in **symbolic memory** and injecting it into the thoughts helps both with problem-solving and with conversational grounding.
- LLMs open up opportunities for (relatively) **domain-independent** collaborative problem solving. Domain-specific pieces of our prompting strategy could be easily replaced for other tasks.

How to establish trust?

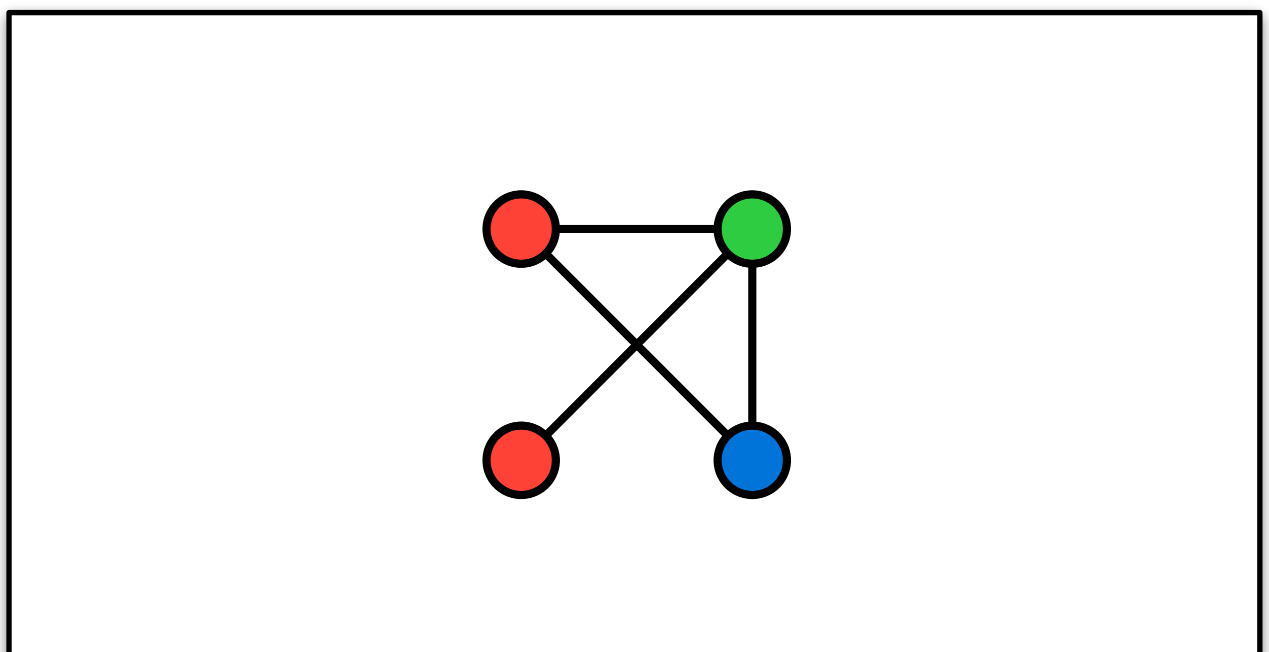


Conclusion

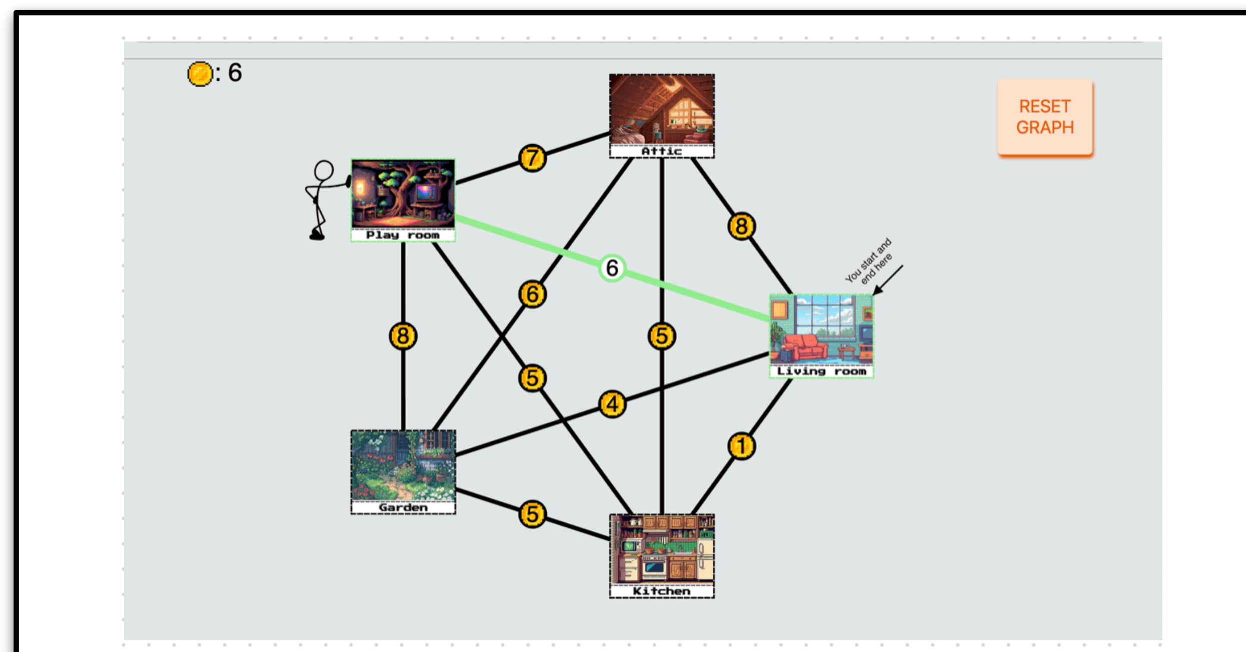
LLMs for solving complex problems



Planning with LLMs



Optimization with LLMs



Collaborative problem solving

Thank you!