



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
23 September 2025

Why do we do semantics?

"We take meaning to be the relation between the form and something external to language."

— Bender & K., "octopus paper", 2020

"Semantics with no treatment of truth-conditions is not semantics."

— David Lewis, *General Semantics*, 1972

Why do we care about truth conditions?

We would like to *solve complex problems* that are specified in language, and this is easier if we first map it to a formal language.

Many complex problems rely on the truth conditions of the sentence. Logic connects truth conditions and proof theory.

All men are mortal.

Socrates is a man.

Therefore, Socrates is mortal.

(Syllogism, Aristotle, 350 BC)

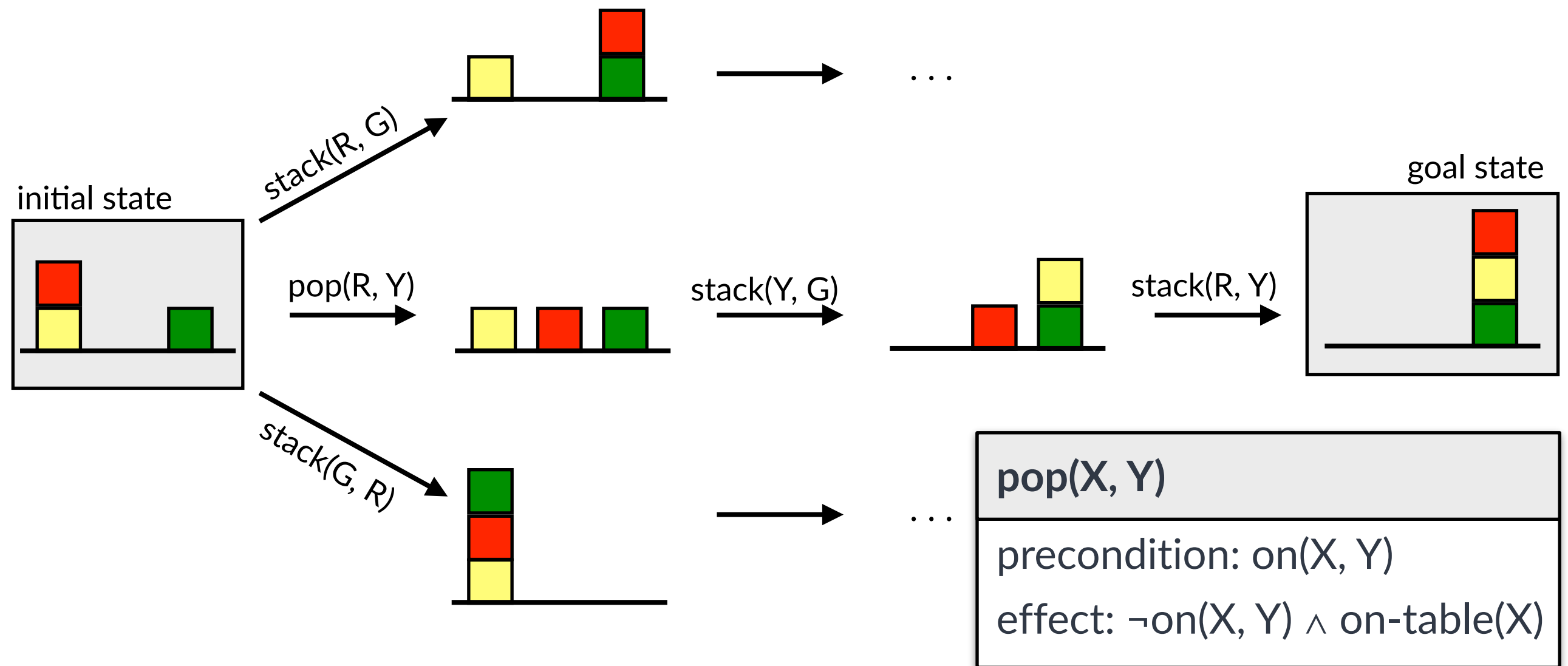
At the other end of Pennsylvania Avenue,
people began to line up for a White House tour.

People formed a line at the end of
Pennsylvania Avenue.

(MNLI dataset, 2018)

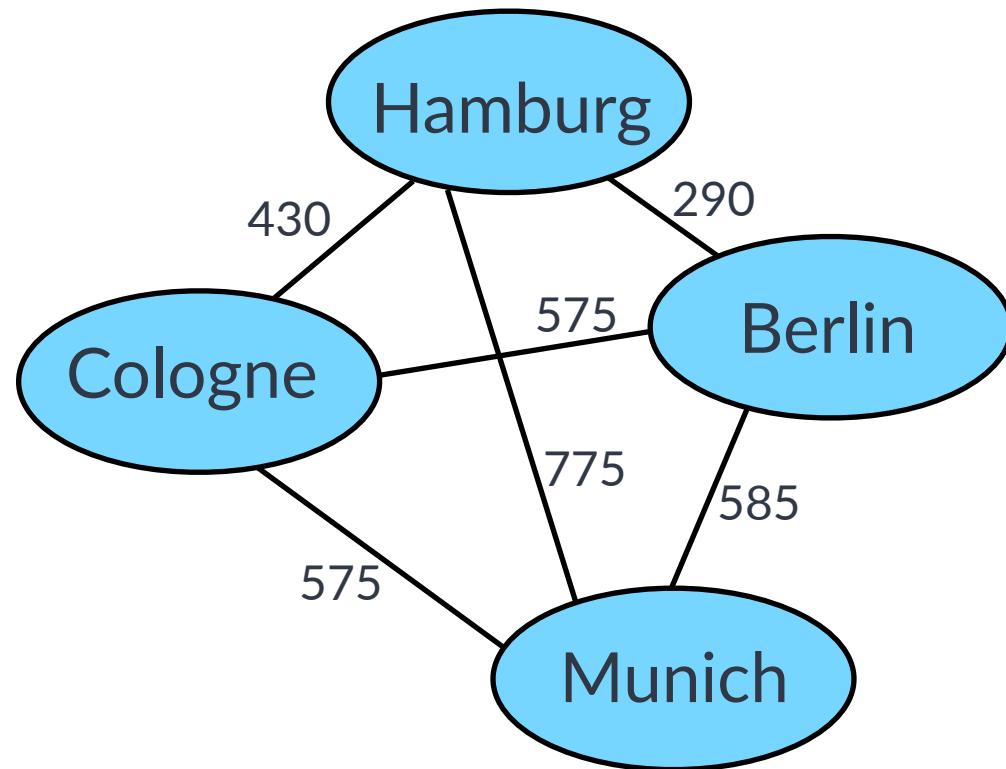
Complex problems: Planning

Logic can be used in many ways to address different complex problems. Truth conditions may look quite different than in semantics textbooks.



Complex problems: Optimization

Logic can be used in many ways to address different complex problems. Truth conditions may look quite different than in semantics textbooks.



"Find the shortest round trip."

Traveling Salesman Problem

Minimize

$$\begin{aligned} & 290 x_{12} + 585 x_{13} + 575 x_{14} \\ & + 290 x_{21} + 775 x_{23} + 430 x_{24} \\ & + 585 x_{31} + 775 x_{32} + 575 x_{34} \\ & + 575 x_{41} + 430 x_{42} + 575 x_{43} \end{aligned}$$

Subject To

$$\text{out1: } x_{12} + x_{13} + x_{14} = 1$$

$$\text{out2: } x_{21} + x_{23} + x_{24} = 1$$

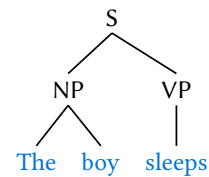
$$\text{out3: } x_{31} + x_{32} + x_{34} = 1$$

$$\text{out4: } x_{41} + x_{42} + x_{43} = 1$$

...

Linear Program (LP)

Broad-Coverage Problem Solving



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

\ Variables: x[i, j] = 1 if the tour goes from city i to city j
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} x[i, j] = 1;
subject to out_degree{i in 1..n}: sum{j in 1..n, j != i} x[i, j] = 1;
```



Step 3.3: Find the Optimal Tour

We evaluate all 24 possible tours:

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



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

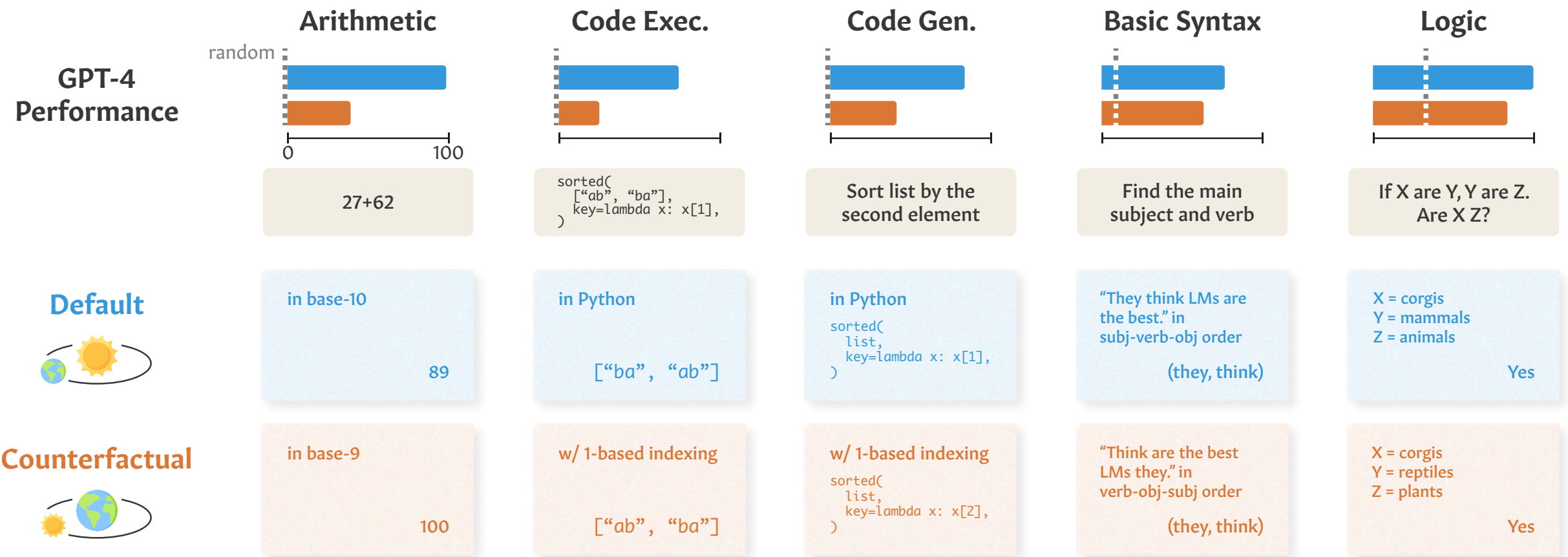
\ Variables: x[i, j] = 1 if the tour goes from city i to city j
var x{i in 1..n, j in 1..n} binary;
var u{i in 2..n} >= 1, <= n-1;

\ 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} x[i, j] = 1;
subject to out_degree{i in 1..n}: sum{j in 1..n, j != i} x[i, j] = 1;
```



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.

Themes for this talk

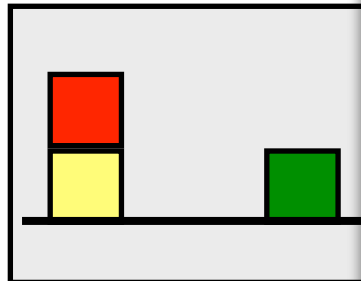
- Symbolic representations
- Generalization
- Truth conditions

#1 Planning

Planning

Example: Blocksworld (simplified)

initial state



```
=====
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]
```

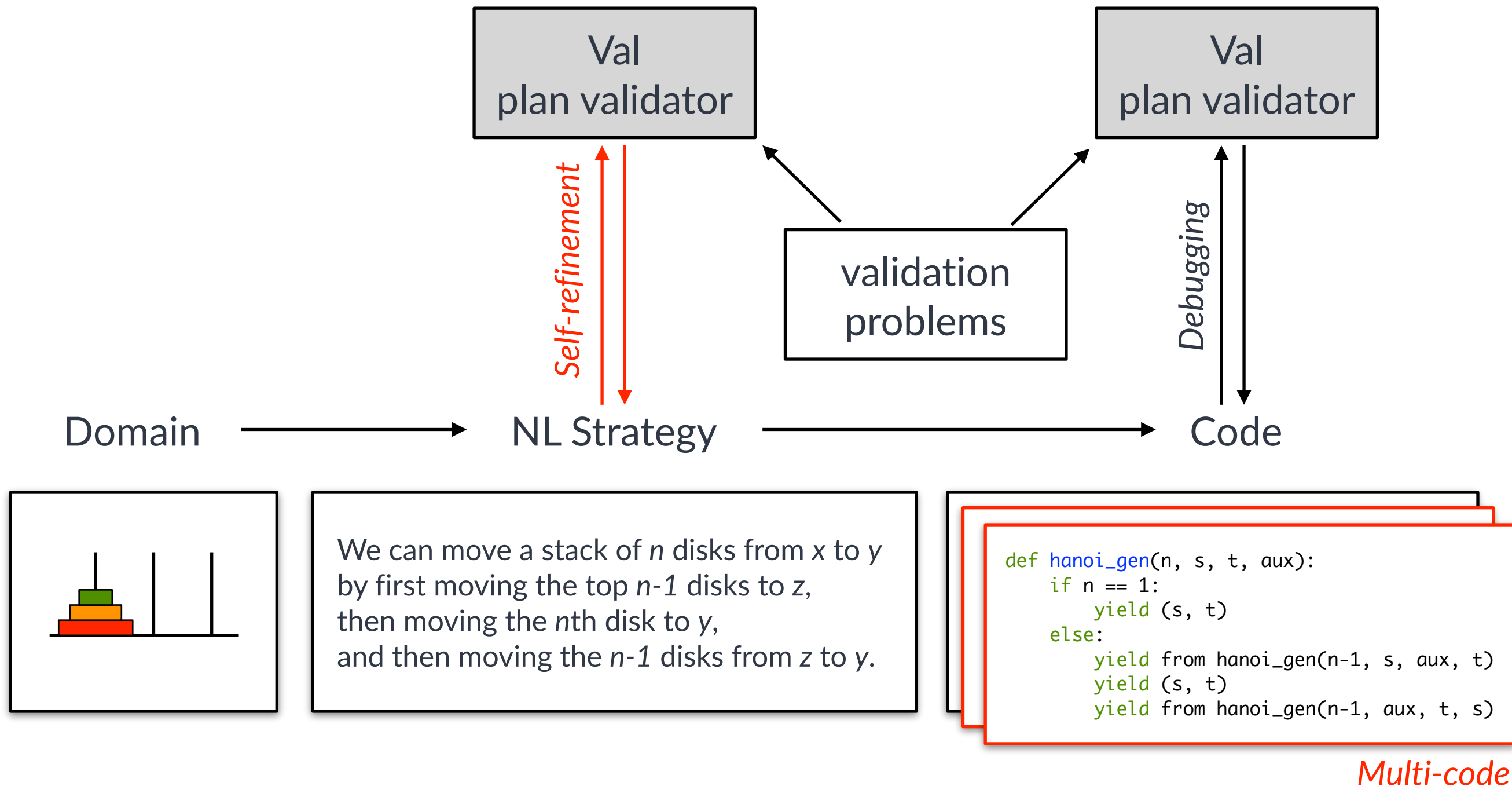
(One-shot prompting strategy of
Valmeekam et al. 2023)

LLM planning 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
childsack14 (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.

Generalized planning with LLMs



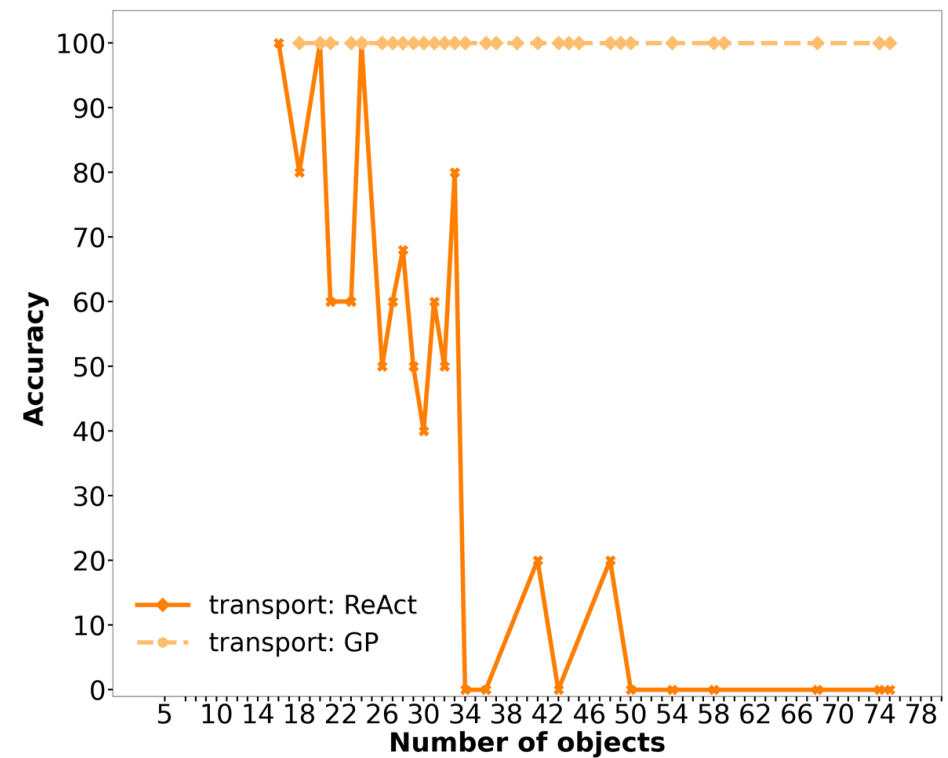
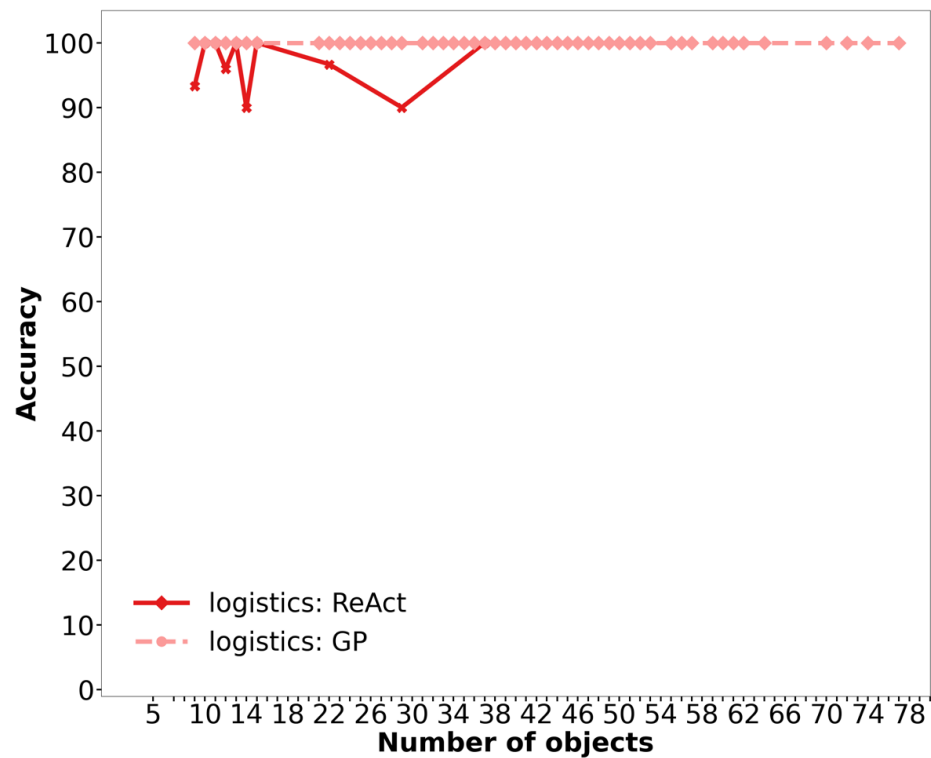
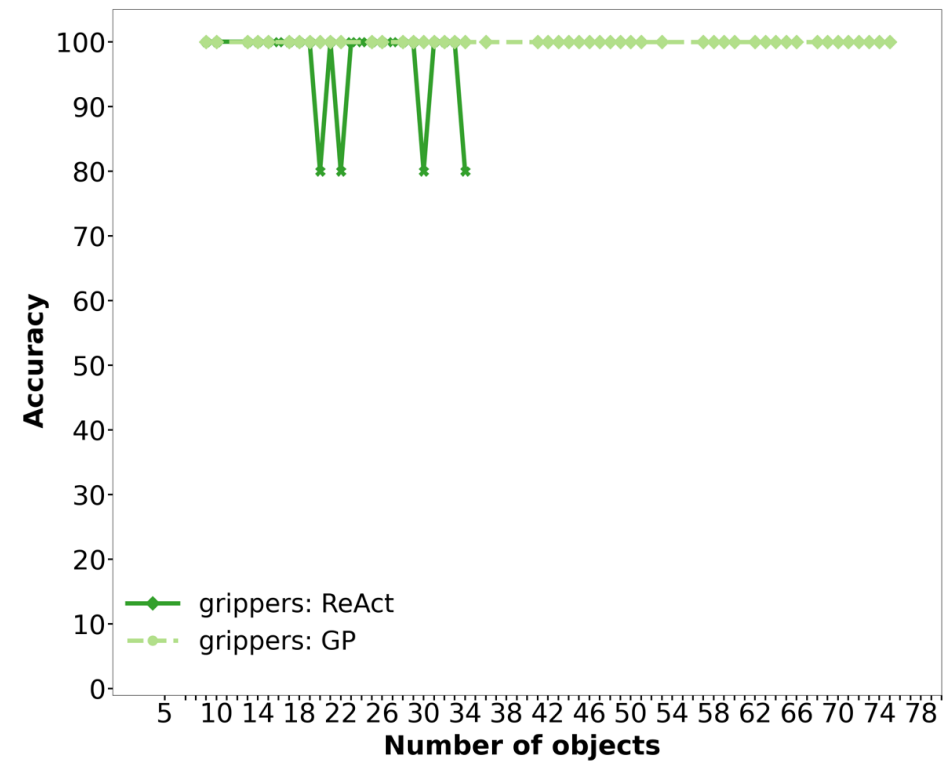
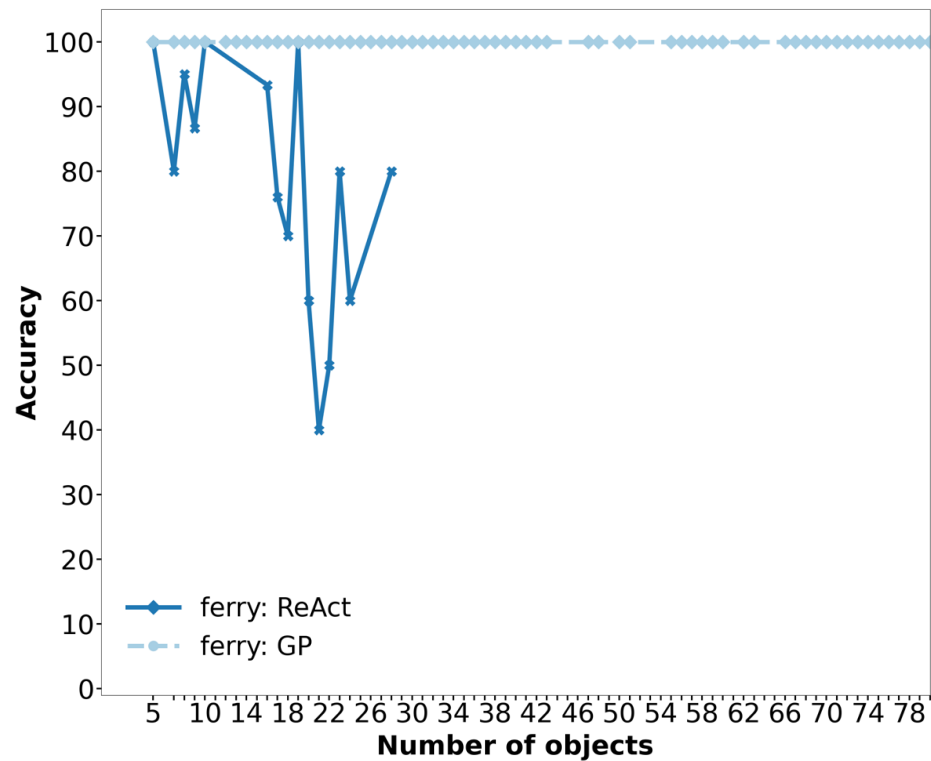
Results

Domain	Silver et al.	Ours	Ours w/o multicode	Ours w/o strat. refinement
Logistics	44	100	94	76
Visitall	80	100	33	78
Blocksworld	11	7	6	5
Goldminer	0	10	2	3
Minigrid	30	48	36	37
<i>Miconic</i>	4	68	0	1
<i>Spanner</i>	6	67	33	67
<i>Ferry</i>	100	100	35	100
<i>Heavy</i>	67	100	100	100

(% solved correctly for GPT-4o; domains in *italics* were already in Silver et al. 2024)

Length generalization

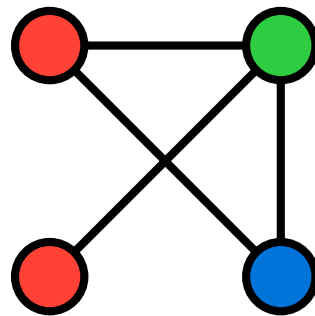
Accuracy Generalized Planning (GP, best seed) vs ReAct by number of objects



#2

Optimization

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?

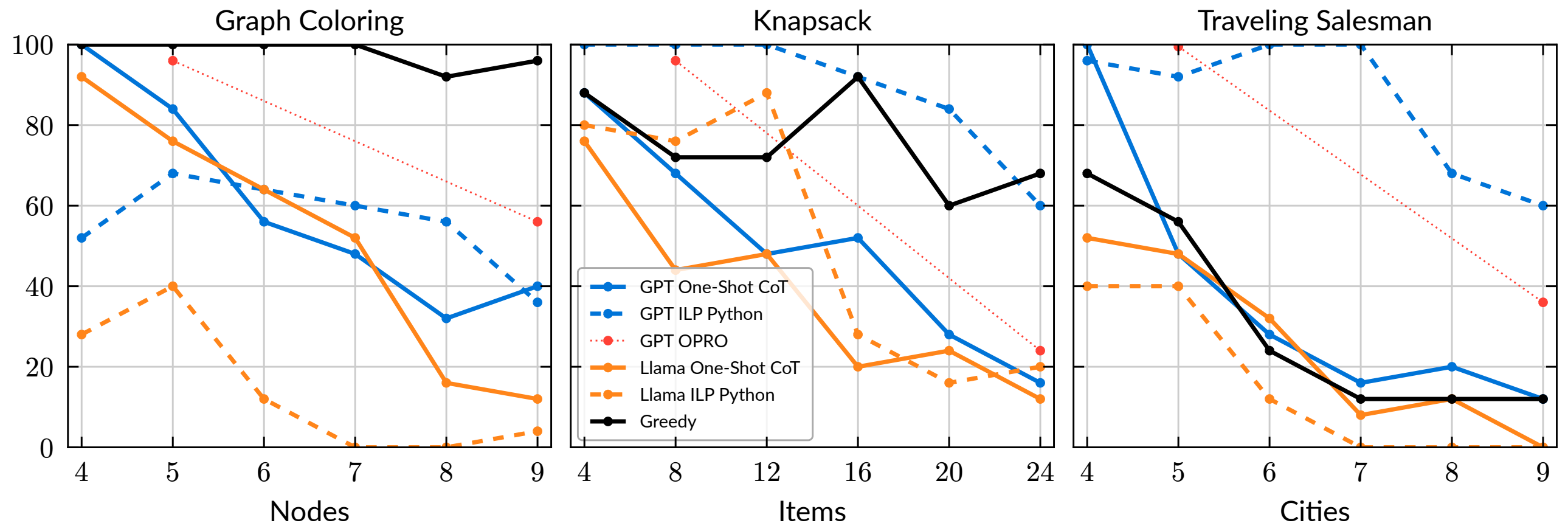
Inverted problem

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.

Evaluation

- EHOP dataset: 3 NP-hard problems x 4 costumes x inverted/not; 25 random instances for each of 6 instance sizes.
- "Traditional" LLMs: GPT-4o, Llama-3.1-70B-Instruct, Qwen 3
"Reasoning" LLMs: DeepSeek-R1, Qwen 3 thinking
- Investigate how LLMs solve the problems by themselves ...
- ... and as "semantic parsers" that map the NL description into linear programs, which are then solved by an exact solver.







Scaling to larger instances is hard



Problems solved much more accurately with help from the exact solver (ILP).




LLMs by themselves rarely beat the greedy heuristics.

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	

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.
- **Reasoning models (DeepSeek-R1)** are more robust to presentation, but still do not reason reliably.

Problem	Variant	Zero-Shot	ILP Python
 GCP	Textbook	98.0	94.0
	Inverted	-75.0	-56.0
	Costumed	-4.0	+3.3
 KSP	Textbook	48.7	97.3
	Inverted	+14.0	+0.7
	Costumed	+5.1	+1.6
 TSP	Textbook	32.0	72.7
	Inverted	-0.7	+8.6
	Costumed	-10.7	+4.2

(DeepSeek-R1 on EHOP-HARD)

#3

Some general thoughts

Solving complex problems with LLMs

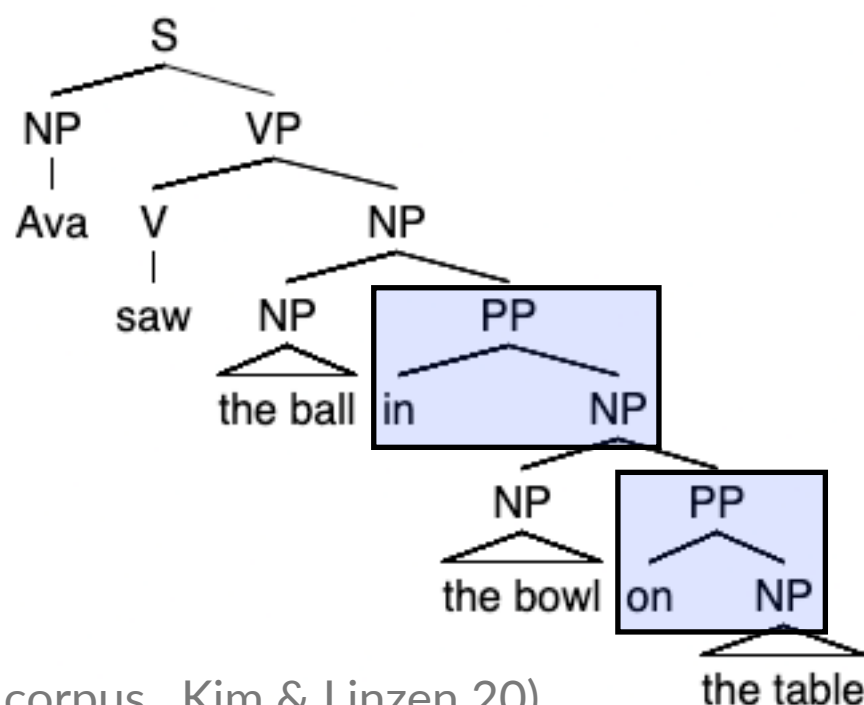
- LLMs offer unprecedented breadth of coverage and will play a role in putting domain-independent problem solving into lay hands.
- Domain independence not as strong as one might think.
- Inclusion of symbolic representation helps with generalization.

Task	Challenge	Symbolic components
Planning	length generalization	generalized plans
Optimization	presentation dependence	linear programs
Collaborative optimization	state tracking	symbolic memory

Symbolic models are generalization machines

```
def hanoi(n, source, destination, aux):  
    if n == 1:  
        print(f"move({source}, {destination})")  
    else:  
        hanoi(n-1, source, aux, destination)  
        print(f"move({source}, {destination})")  
        hanoi(n-1, aux, destination, source)
```

If a symbolic program performs correctly for input sizes 1, ..., 10, it probably works for 20 too.




(COGS corpus, Kim & Linzen 20)

If a (neuro)symbolic model performs correctly for depth 0-2, it probably works for depth 3-12 too (= compositionality).

(AM parser; Groschwitz, ..., K. ACL 2018)


LLMs don't generalize in that way

 City 1 and city 2 are 15 miles apart.
City 1 and city 3 are 14 miles apart.
City 1 and city 4 are 14 miles apart.
City 2 and city 3 are 16 miles apart.
City 2 and city 4 are 1 miles apart.
City 3 and city 4 are 16 miles apart.

```
from gurobipy import GRB, Model, quicksum

def f():
    # Create the model
    model = Model("TSP")

    # Objective: Minimize the total travel distance
    model.setObjective(quicksum(distances[i, j] * x[i, j]
                                for i in range(n) for j in range(n) if i != j), GRB.MINIMIZE)
```

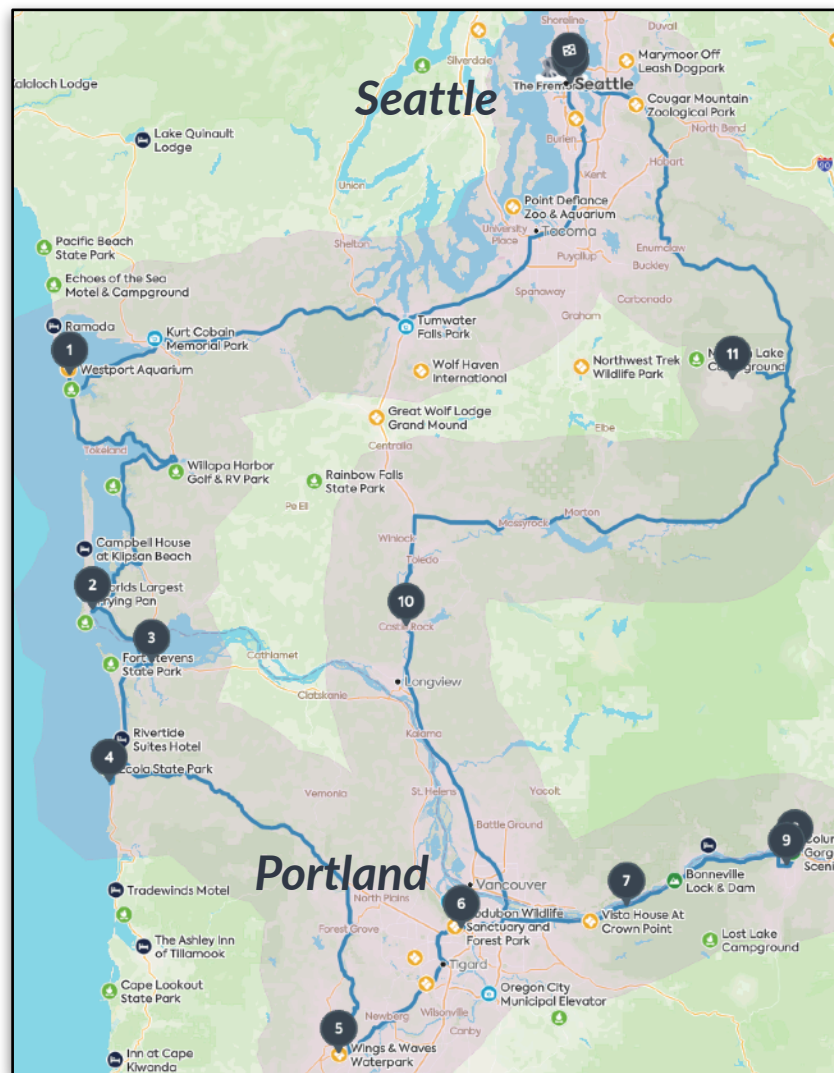
 City 1 and city 2 are 8 miles apart.
City 1 and city 3 are 14 miles apart.
City 1 and city 4 are 13 miles apart.
City 2 and city 3 are 6 miles apart.
City 2 and city 4 are 15 miles apart.
City 3 and city 4 are 3 miles apart.

```
from gurobipy import GRB, Model, quicksum

def f():
    # Create the model
    model = Model("TSP")

    # Objective: Minimize the total travel distance
    model.setObjective(quicksum(distances[i+1, j+1] * x[i, j]
                                for i in range(n) for j in range(n) if i != j), GRB.MINIMIZE)
```


Risk profiles



WSJ Nuclear Power Is Back. And This Time, AI Can Help Manage the Reactors.

THE WALL STREET JOURNAL.

English Edition | Print Edition | Video | Audio | Latest Headlines | More

Latest World Business U.S. Politics Economy Tech Markets & Finance Opinion Arts Lifestyle Real Estate Personal Finance Health Style Sports

CIO JOURNAL

Nuclear Power Is Back. And This Time, AI Can Help Manage the Reactors.

Argonne National Lab has an AI-based tool that can help design and operate nuclear reactors—at a time when AI itself is feeding a power frenzy.

By Belle Lin [Follow](#)

April 11, 2025 7:00 am ET

[Share](#) [Resize](#) [Listen \(2 min\)](#)



AI in Banks: Unleash a New Era of Software Engineering Productivity

Banks that effectively deploy AI tools across the software development life cycle could realize significant cost savings by 2028

WSJ Stay Informed, Stay Ahead Just €2/week [SUBSCRIBE NOW](#)

<https://www.wsj.com/articles/nuclear-power-is-back-and-this-time-ai-can-help-manage-the-reactors-5ce03ae7>

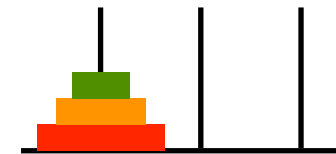
Can we get correctness guarantees?

- No output of an LLM should ever be trusted; there are no guarantees of correctness or generalization.
- *Inductive risk* (Hempel 1965): How do you infer universal correctness from finite observations?
- We *can* potentially verify the universal correctness of an LLM-generated symbolic artefact (generalized plan, LP, ...).
- But there is no verification without truth conditions!

Conclusion

“Semantics with no treatment of truth-conditions is not semantics.”
(Lewis 1972)

Semantics needs truth conditions



```
def hanoi(n, source, destination, aux):
    if n == 1:
        print(f"move({source}, {destination})")
    else:
        hanoi(n-1, source, aux, destination)
        print(f"move({source}, {destination})")
        hanoi(n-1, aux, destination, source)
```

Generalized planning with LLMs

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	

Optimization with LLMs



Verification needs truth conditions

Thank you!