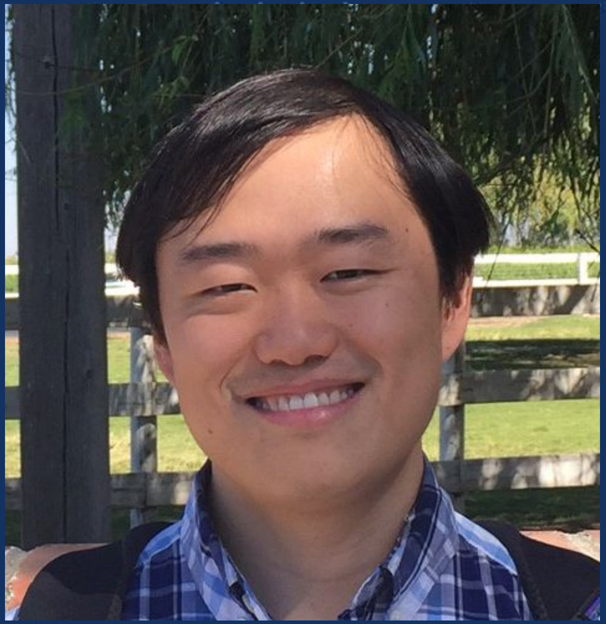
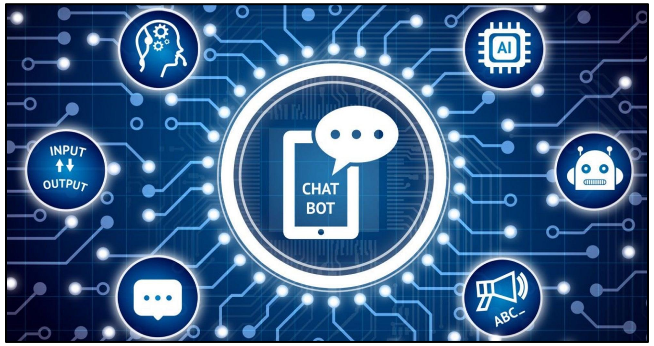
Towards a unified framework of Neural and Symbolic Decision Making

Yuandong Tian
Research Scientist Director

Meta AI (FAIR)

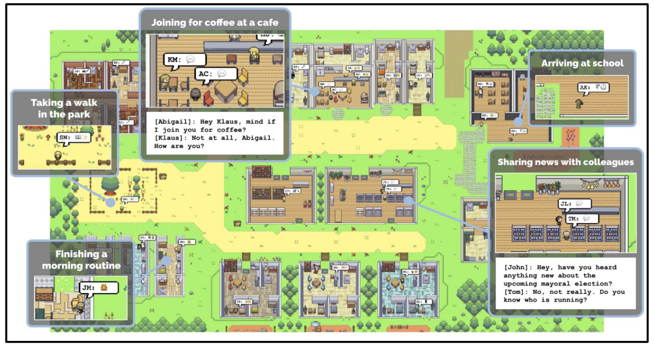


Large Language Models (LLMs)

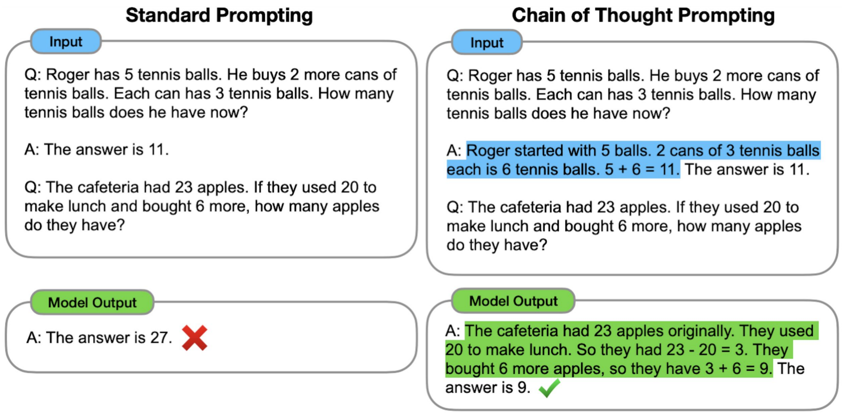


Conversational AI

Content Generation



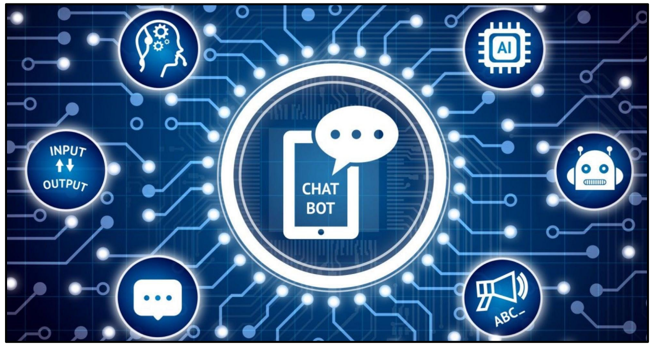
AI Agents



Reasoning

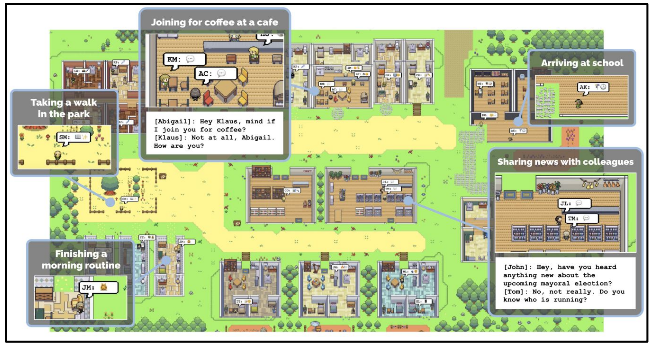
Planning

Large Language Models (LLMs)

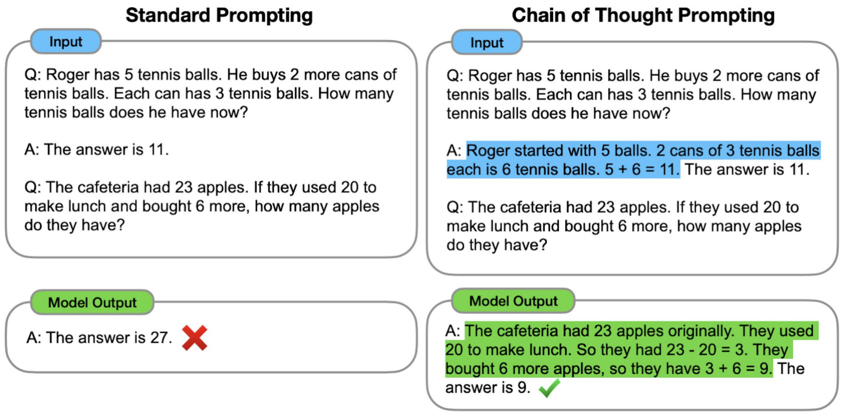


Conversational AI

Content Generation



AI Agents



Reasoning

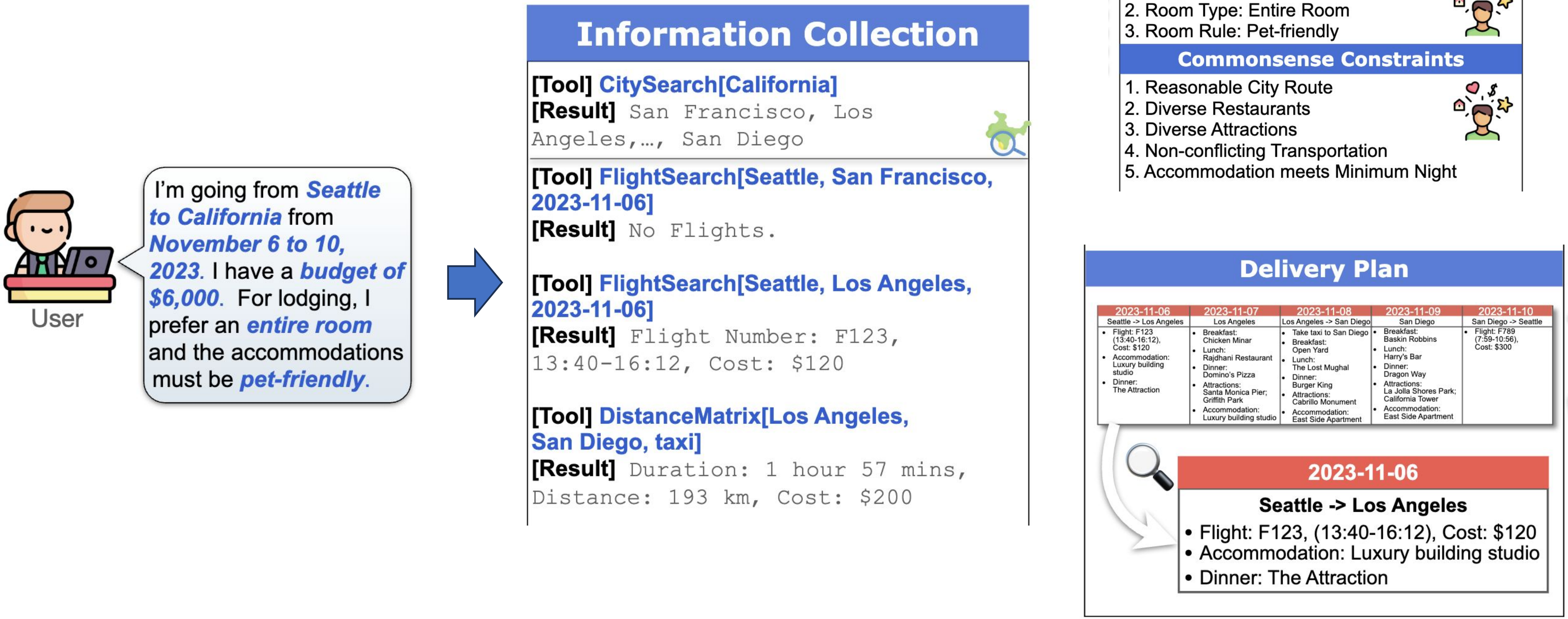
Planning

What LLMs cannot do well yet?

Travel planning

[J. Xie et al, *TravelPlanner: A Benchmark for Real-World Planning with Language Agents*, ICML'24 (**Spotlight**)]

What LLMs cannot do well yet



[J. Xie et al, *TravelPlanner: A Benchmark for Real-World Planning with Language Agents*, ICML'24 (**Spotlight**)]

Using SoTA LLMs for Travel Planning (not great)

Even SoTA LLMs struggle for such hard planning problems

First tool use,
Then plan the travel

Ground-truth tool use,
Then plan the travel

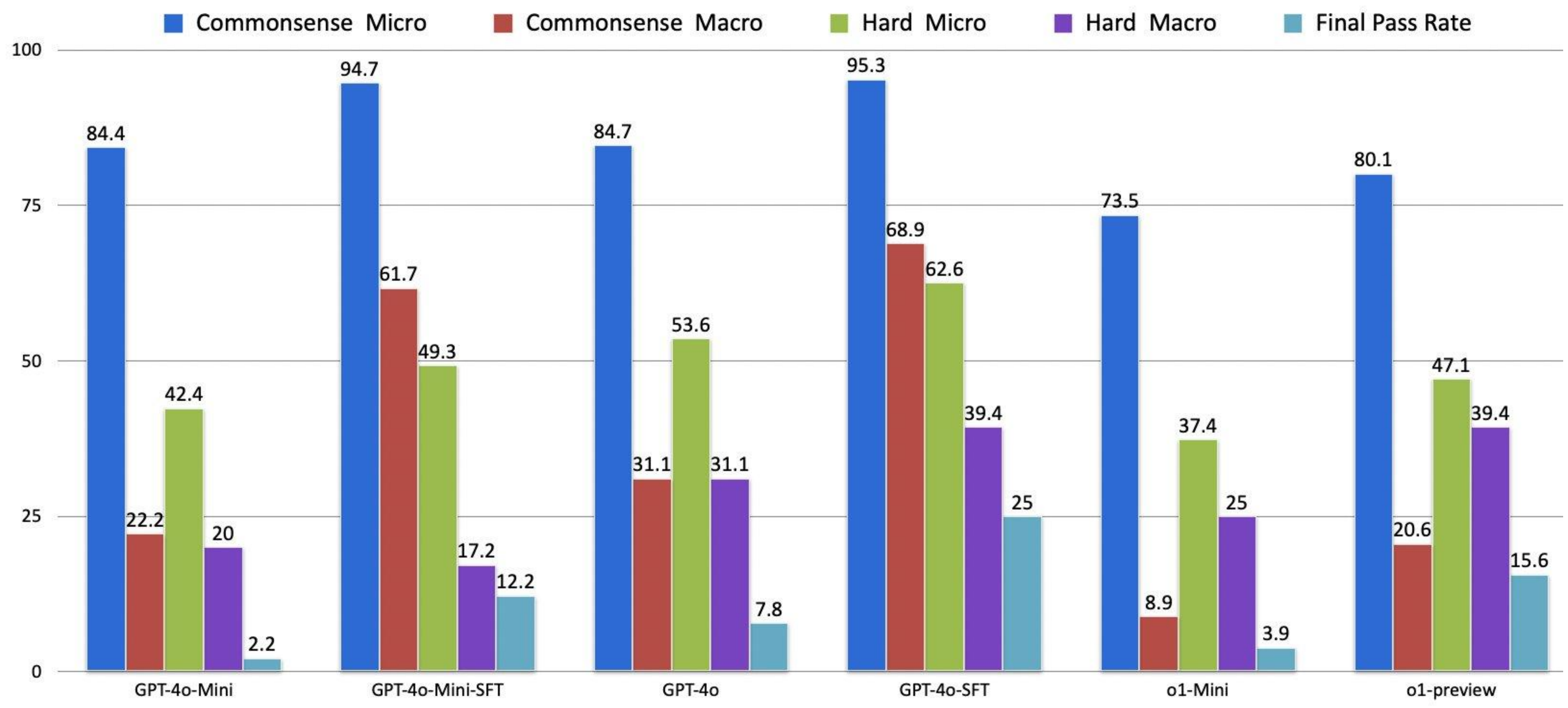
	Validation (#180)						Test (#1,000)					
	Delivery Rate	Commonsense Pass Rate		Hard Constraint Pass Rate		Final Pass Rate	Delivery Rate	Commonsense Pass Rate		Hard Constraint Pass Rate		Final Pass Rate
		Micro	Macro	Micro	Macro			Micro	Macro	Micro	Macro	
Greedy Search	100	74.4	0	60.8	37.8	0	100	72.0	0	52.4	31.8	0
Two-stage												
Mistral-7B-32K (Jiang et al., 2023)	8.9	5.9	0	0	0	0	7.0	4.8	0	0	0	0
Mixtral-8×7B-MoE (Jiang et al., 2024)	49.4	30.0	0	1.2	0.6	0	51.2	32.2	0.2	0.7	0.4	0
Gemini Pro (G Team et al., 2023)	28.9	18.9	0	0.5	0.6	0	39.1	24.9	0	0.6	0.1	0
GPT-3.5-Turbo (OpenAI, 2022)	86.7	54.0	0	0	0	0	91.8	57.9	0	0.5	0.6	0
GPT-4-Turbo (OpenAI, 2023)	**89.4**	**61.1**	**2.8**	**15.2**	**10.6**	**0.6**	**93.1**	**63.3**	**2.0**	**10.5**	**5.5**	**0.6**
Sole-planning												
Direct$_{GPT-3.5-Turbo}$	100	60.2	4.4	11.0	2.8	0	100	59.5	2.7	9.5	4.4	0.6
CoT$_{GPT-3.5-Turbo}$	100	66.3	3.3	11.9	5.0	0	100	64.4	2.3	9.8	3.8	0.4
ReAct$_{GPT-3.5-Turbo}$	82.2	47.6	3.9	11.4	6.7	0.6	81.6	45.9	2.5	10.7	3.1	0.7
Reflexion$_{GPT-3.5-Turbo}$	93.9	53.8	2.8	11.0	2.8	0	92.1	52.1	2.2	9.9	3.8	0.6
Direct$_{Mixtral-8x7B-MoE}$	100	68.1	5.0	3.3	1.1	0	99.3	67.0	3.7	3.9	1.6	0.7
Direct$_{Gemini\ Pro}$	93.9	65.0	8.3	9.3	4.4	0.6	93.7	64.7	7.9	10.6	4.7	2.1
Direct$_{GPT-4-Turbo}$	**100**	**80.4**	**17.2**	**47.1**	**22.2**	**4.4**	**100**	**80.6**	**15.2**	**44.3**	**23.1**	**4.4**

[J. Xie et al, *TravelPlanner: A Benchmark for Real-World Planning with Language Agents*, ICML'24 (**Spotlight**)]

How about o1?

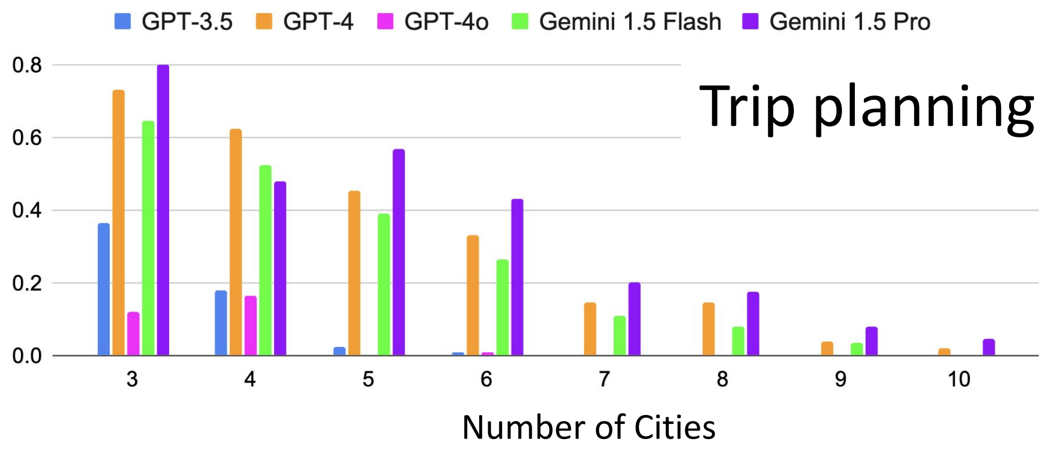
Final Pass Rate
0
0
0
0
0
0.6%

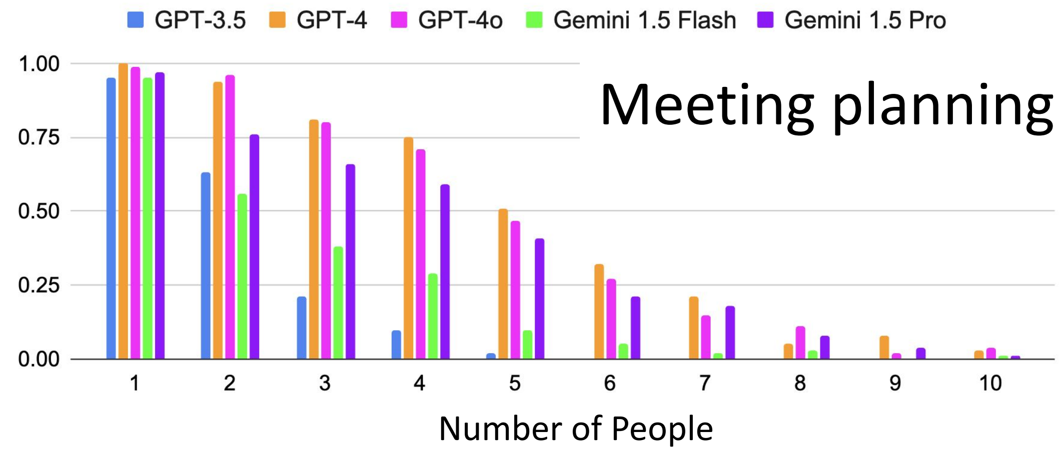
GPT-4-turbo → **0.6%**

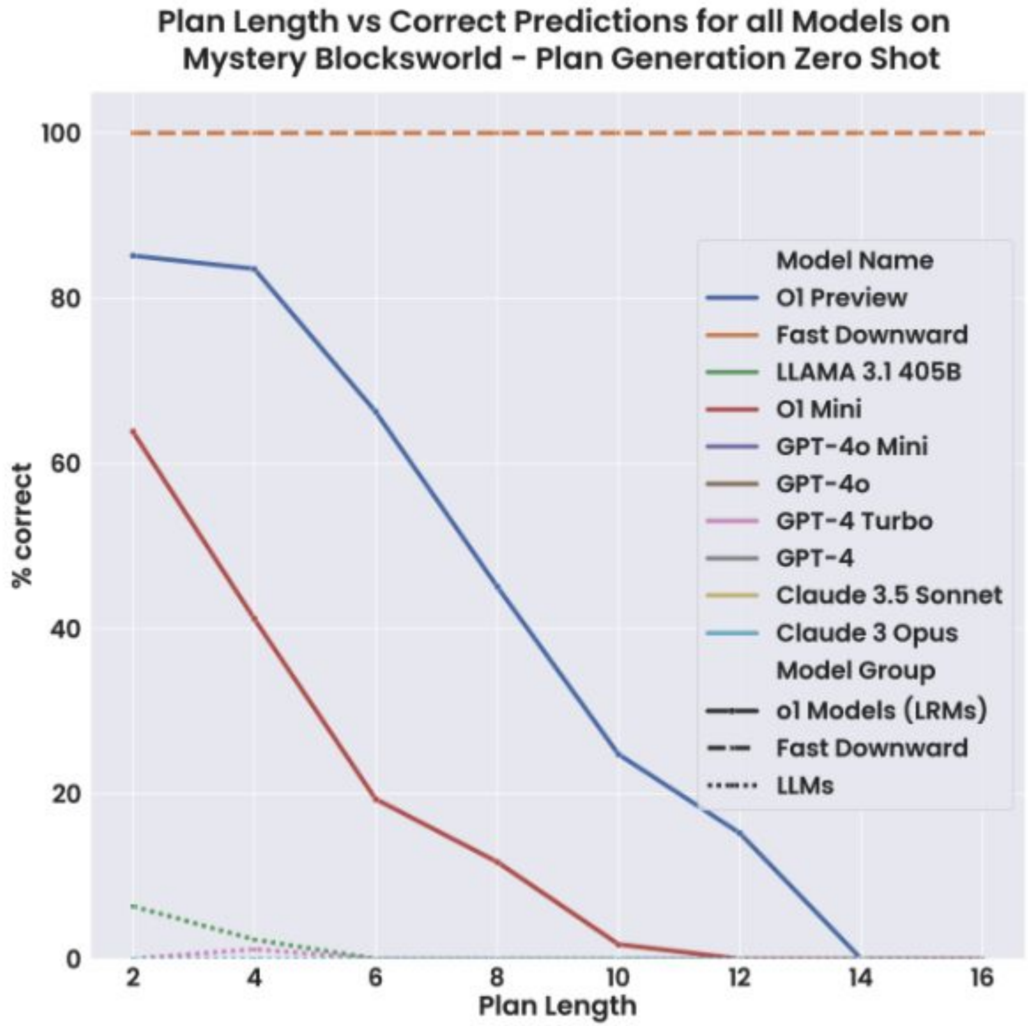


[J. Xie et al, *TravelPlanner: A Benchmark for Real-World Planning with Language Agents*, ICML'24 (**Spotlight**)]

LLM planning is still a hard problem



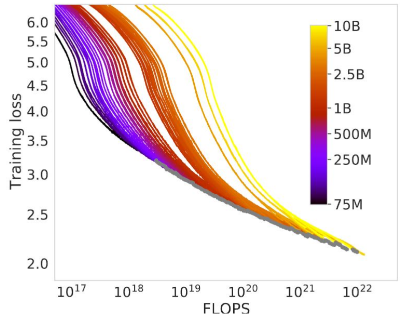




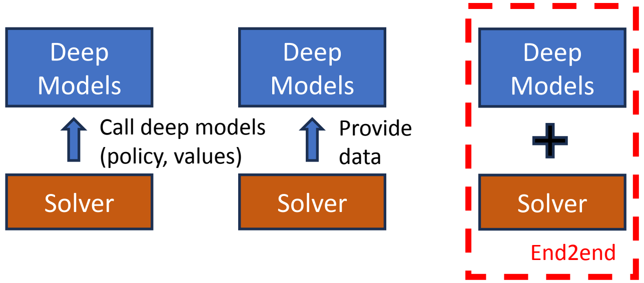
[H. S. Zheng et al, *NATURAL PLAN: Benchmarking LLMs on Natural Language Planning*, arXiv'24]

What are the Solutions?

What are the Solutions?



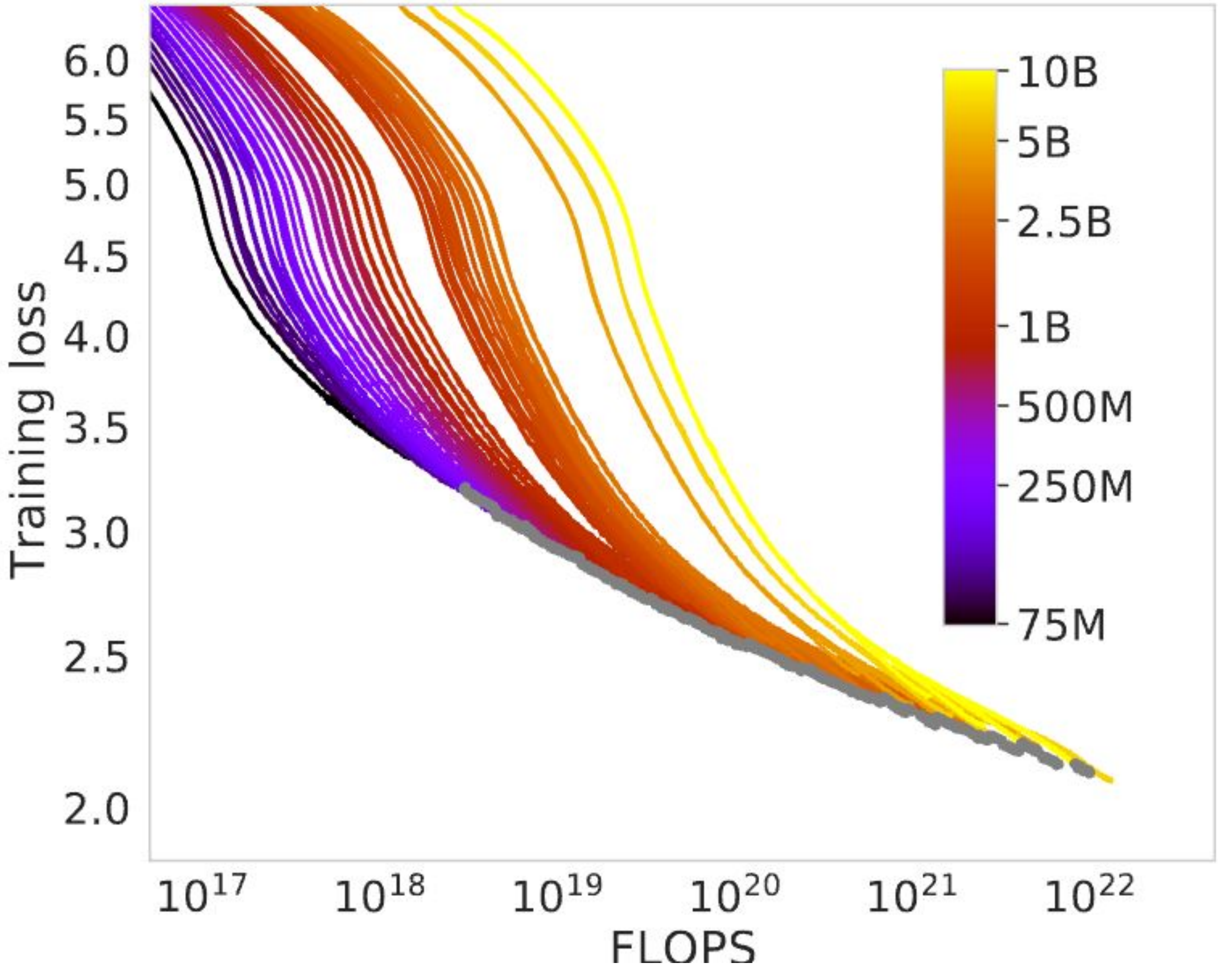
Option **One**: Scaling Law



Option **Two**: Hybrid System

Option **Three**: Emerging Symbolic Structure from Neural network

Option **One**: The Scaling Law



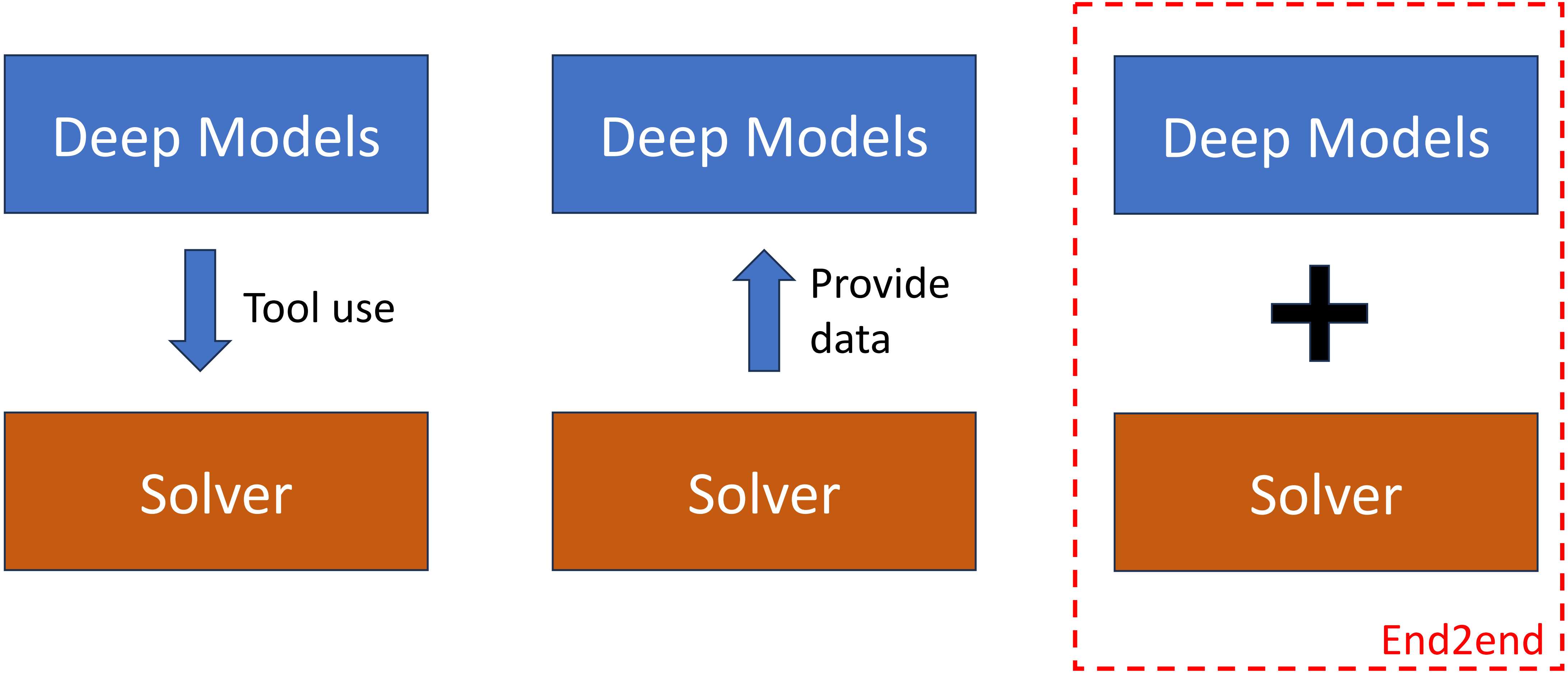
More data
More compute
Larger models

Very expensive

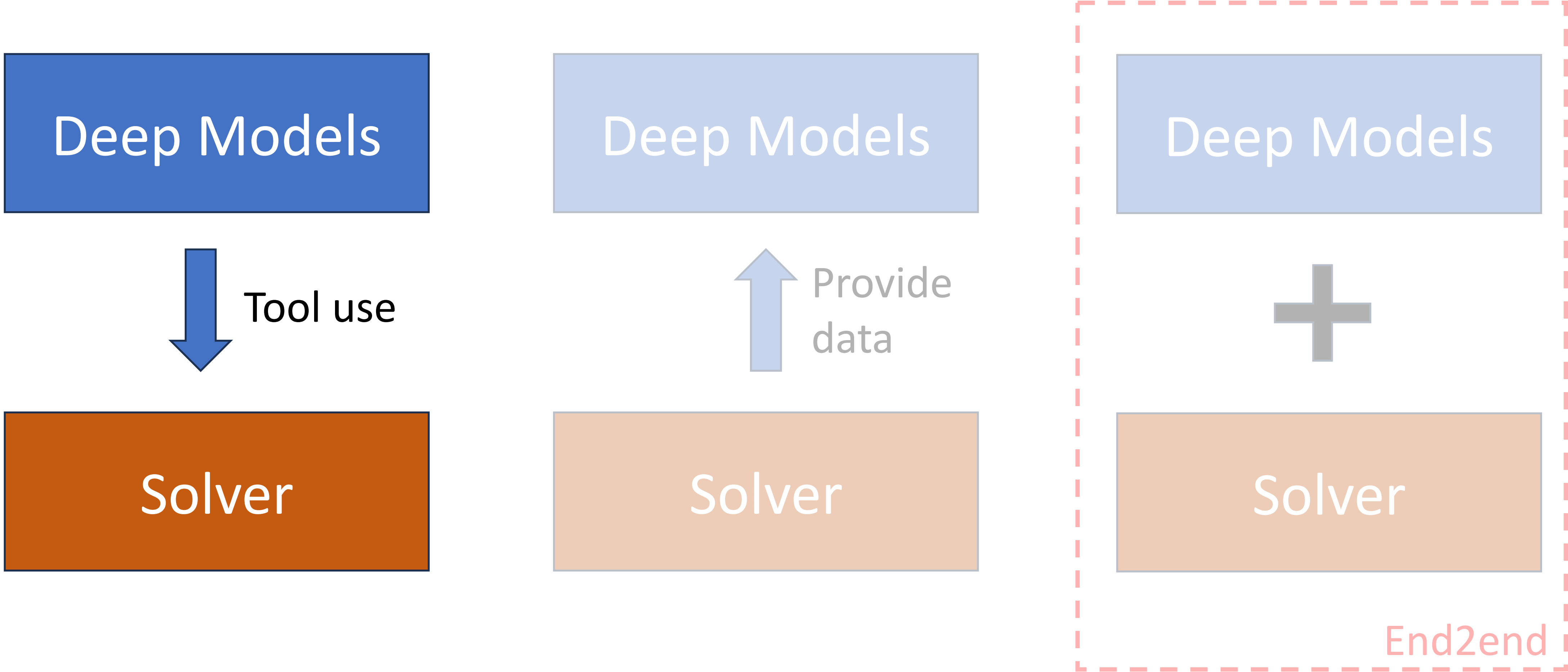
Does that work for reasoning/planning?

[J. Hoffmann*, S. Borgeaud*, A. Mensch* et al, *Training Compute-Optimal Large Language Models*]

Option **Two**: Hybrid Systems



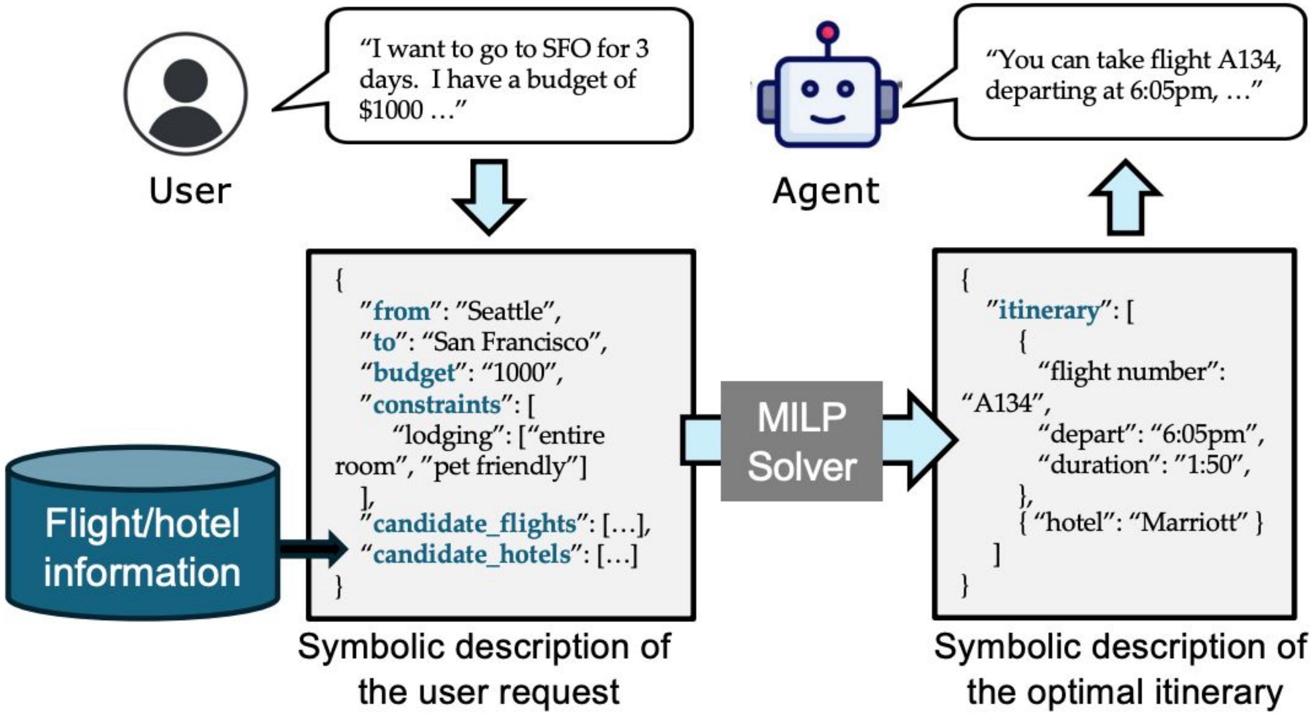
Option **Two**: Hybrid Systems



Language-Driven Guaranteed Travel Planning

LLMs can not handle too many constraints? -> Combinatorial Solvers can!

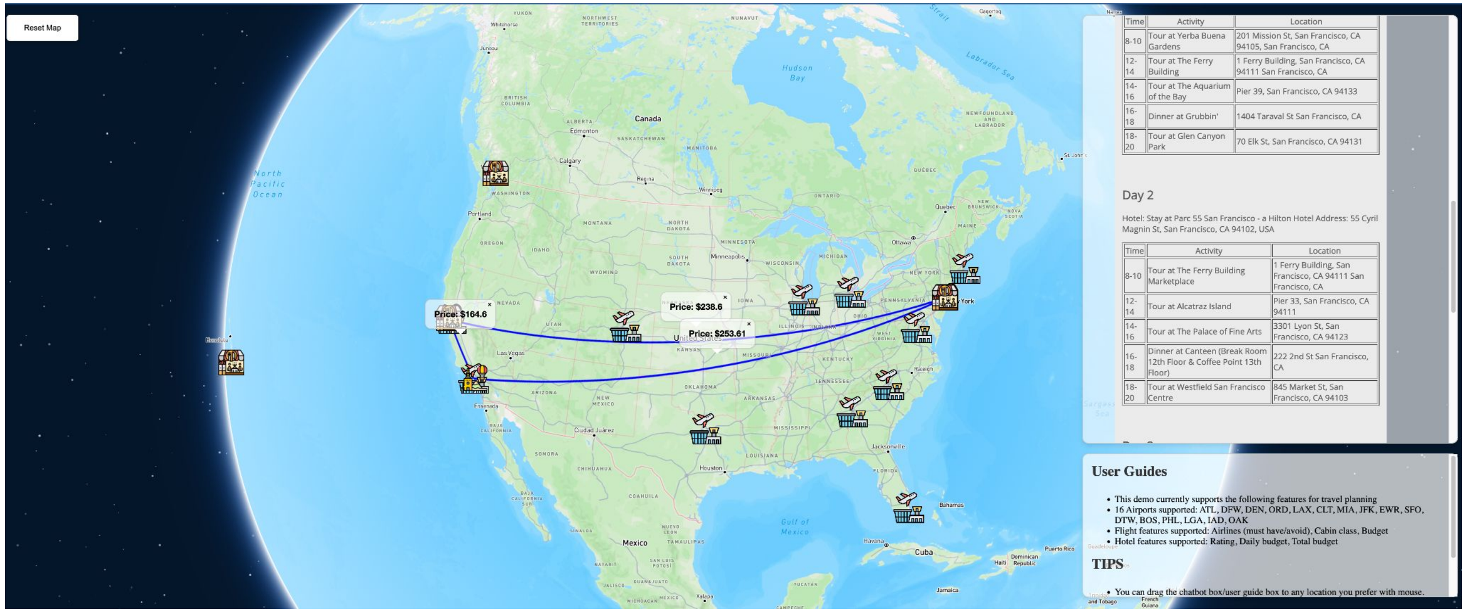
Inference



- Realistic dataset: collect from the real world
- User instruction translator: Fine-tuned LLM to convert user request into symbolic description, augmented by flight/hotel information from database.
- Impose constraints and formalize the travel planning as Mixed Integer Linear Programming (MILP).
- Build a combinatorial solver to give optimal solution.

Ju et al, *To the Globe (TTG): Towards Language-Driven Guaranteed Travel Planning (EMNLP'24 Demo)*

Experiments (End-to-end Human Evaluation)



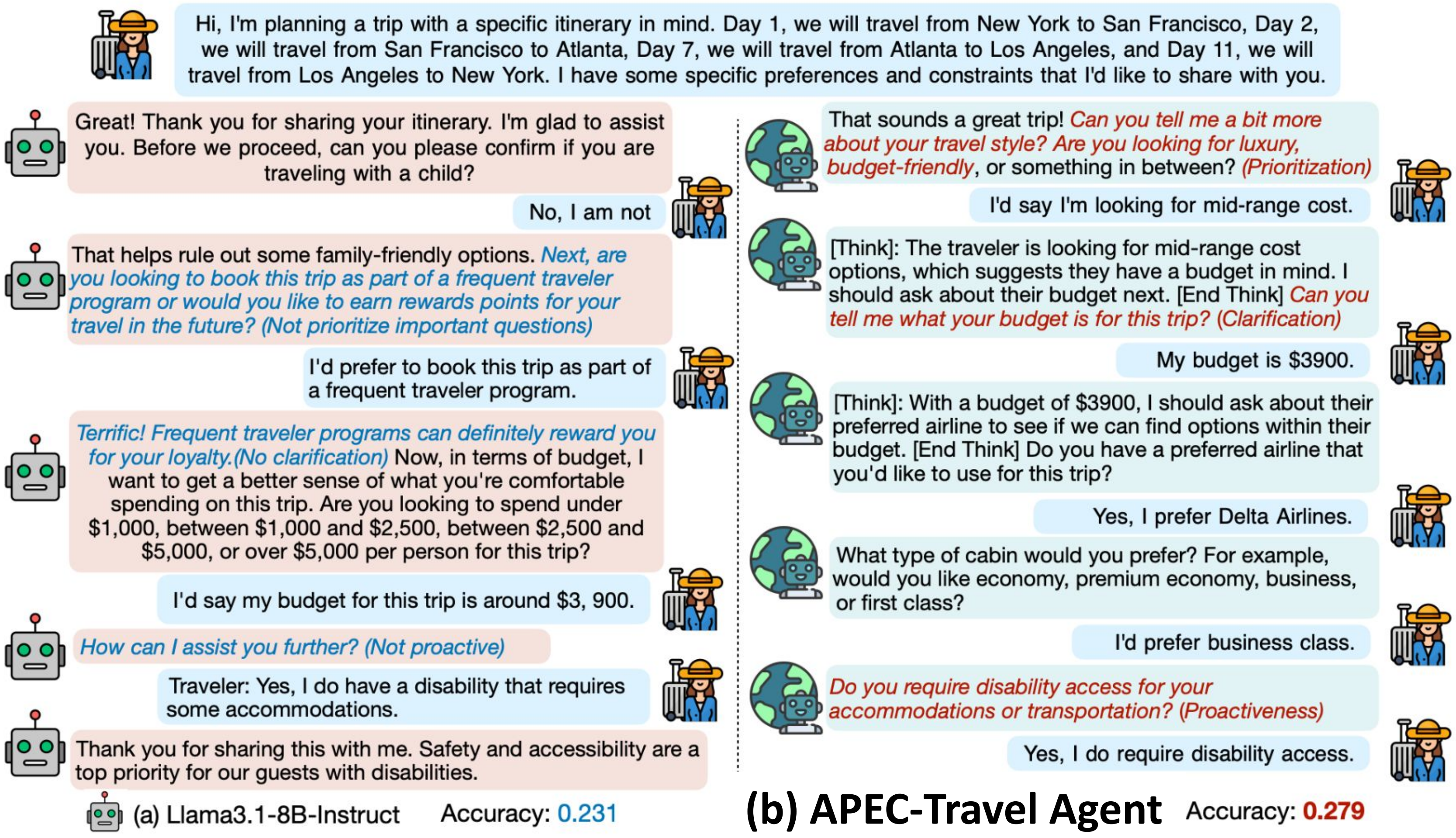
Response phase	Time (s)
Instruction Translator	2.508±0.116
MILP Solver	
- Loading constraints	0.047±0.061
- Solving	0.527±0.457
- Total	0.575±0.507

Question	Detractors %	Promoters %	Net %
...fully satisfies the...request	-13.3	+53.3	+40.0
...offers good value for the money...	-16.8	+52.0	+35.1
...is efficient...	-16.2	+53.1	+36.9

Net Prompter Scores (NPS) and its breakdown in three dimensions: satisfaction, value and efficiency.

Ju et al, *To the Globe (TTG): Towards Language-Driven Guaranteed Travel Planning (EMNLP'24 Demo)*

Multi-round Dialogs to Collect Information

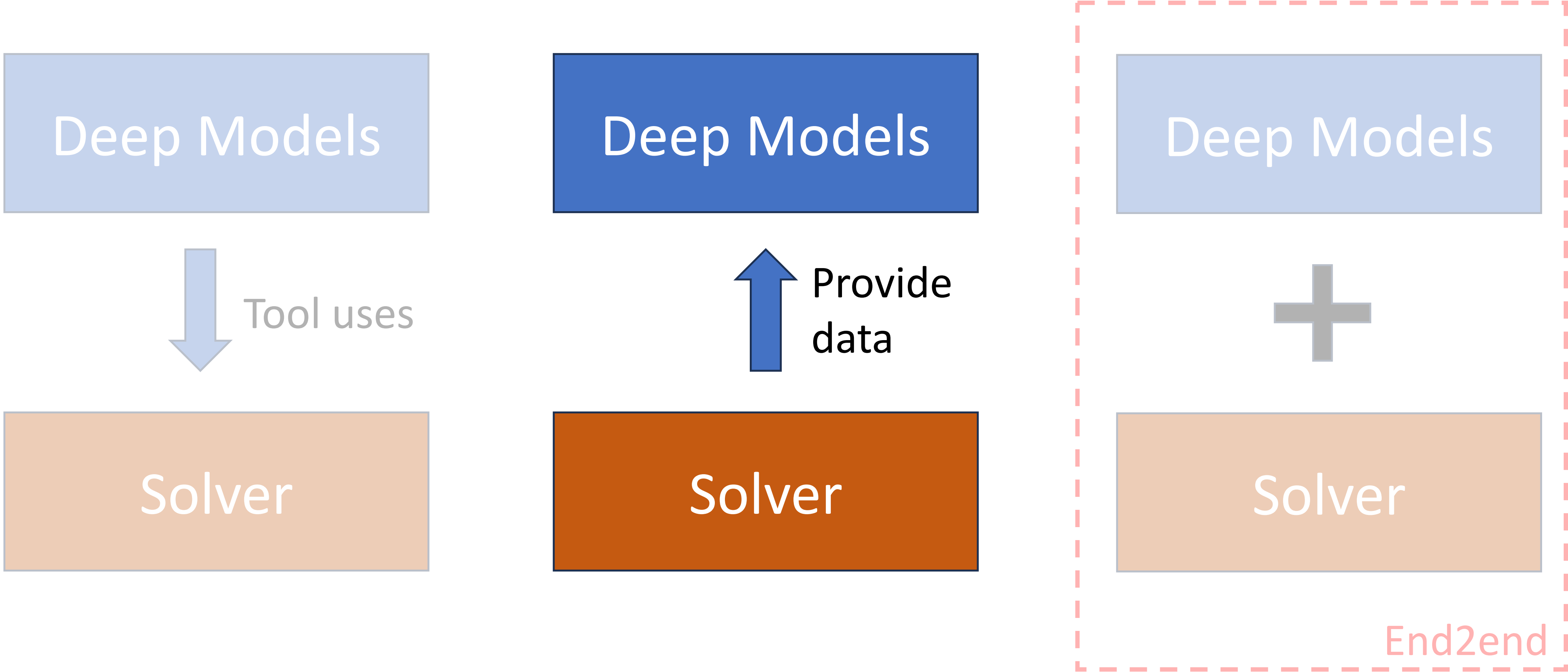


User has hidden constraints, how to figure out?

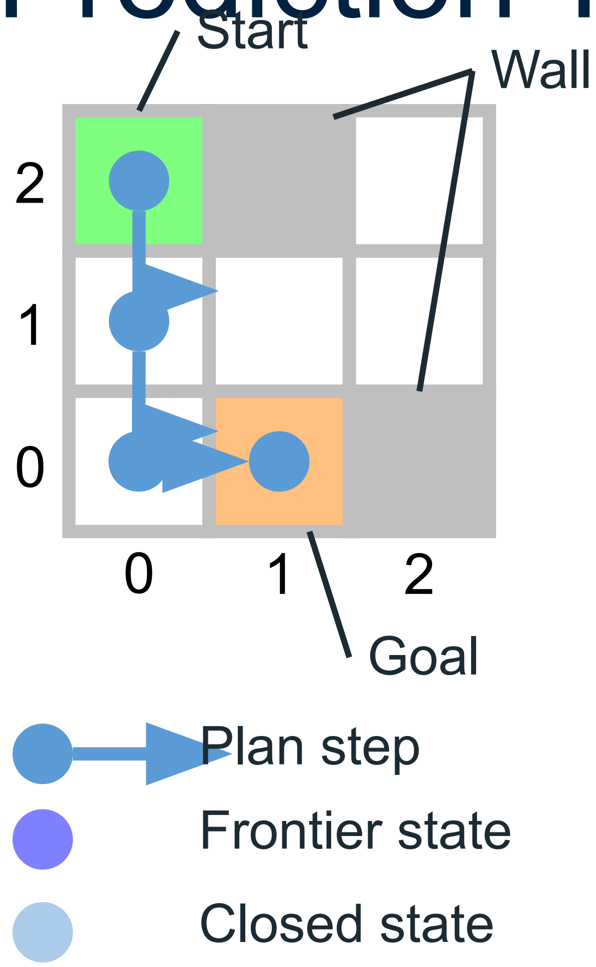
□ **Proactively ask!**

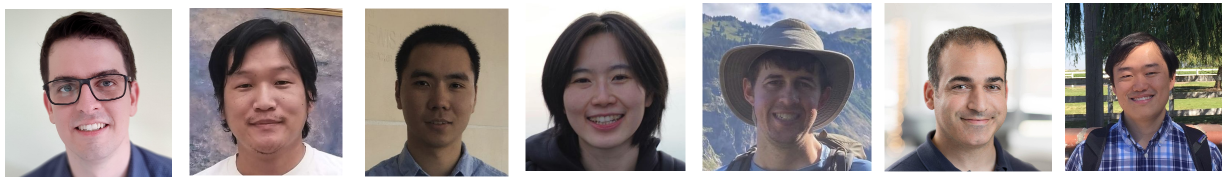
[Jiang et al, *Towards Full Delegation: Designing Ideal Agentic Behaviors for Travel Planning*]

Option **Two**: Hybrid Systems



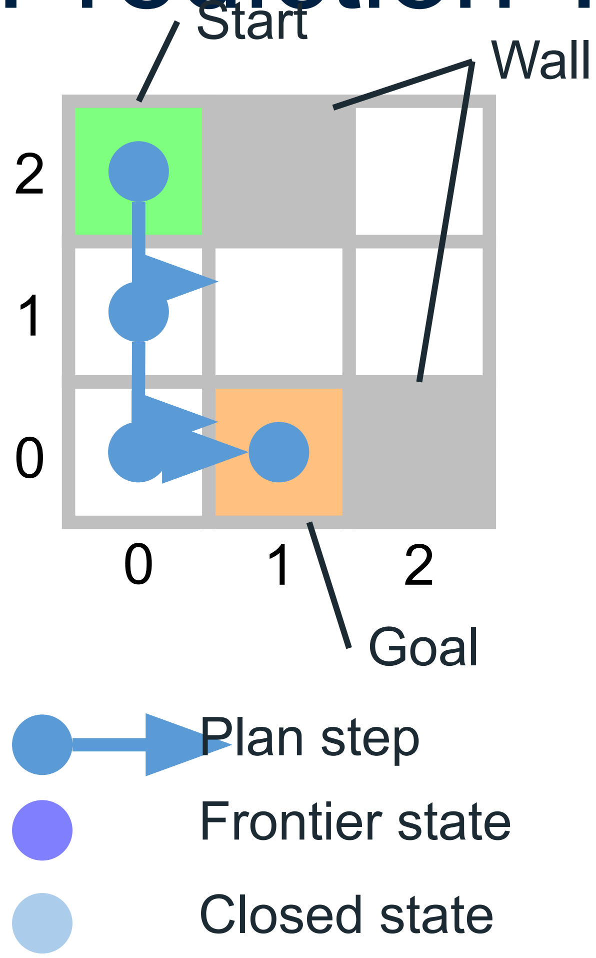
Searchformer: A* Search as a Token Prediction Task





[L. Lehnert, et al, *Beyond A*: Better Planning with Transformers via Search Dynamics Bootstrapping*, COLM'24]

Searchformer: A* Search as a Token Prediction Task



<prompt>

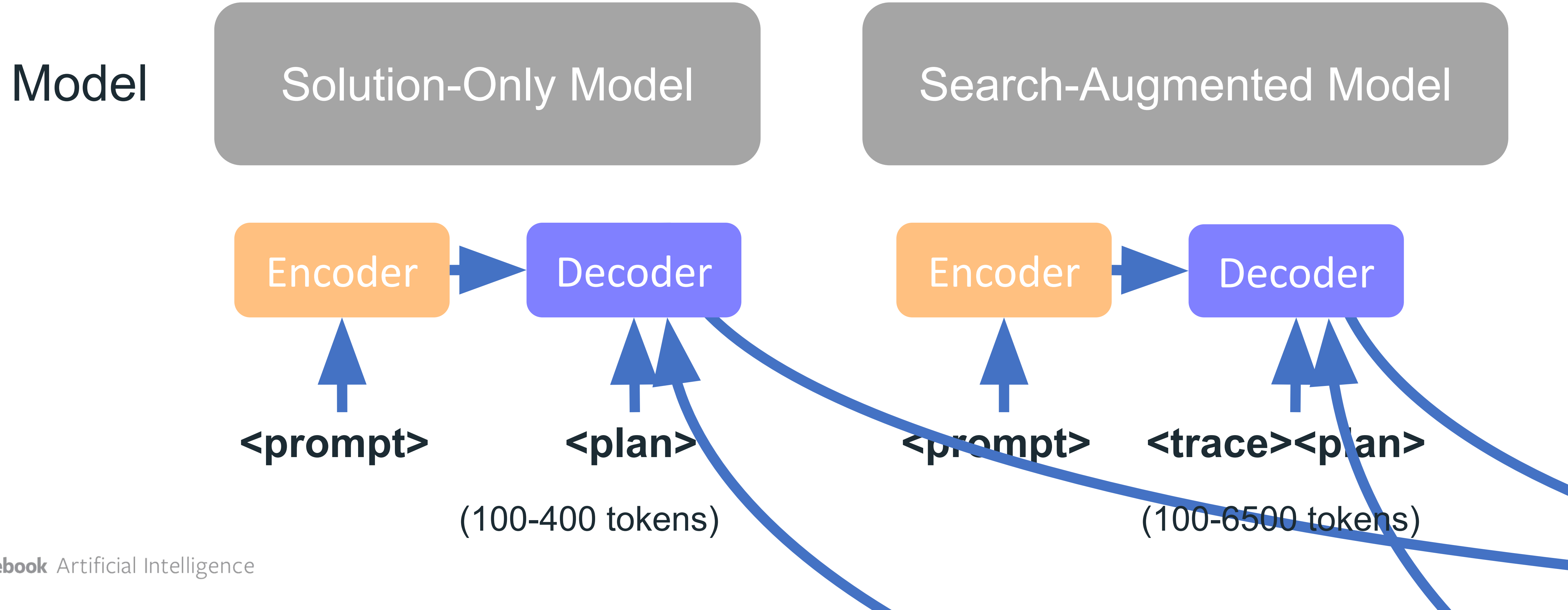
bos
start 0 2
goal 1 0
wall 1 2
wall 2 0
eos

<trace><plan>

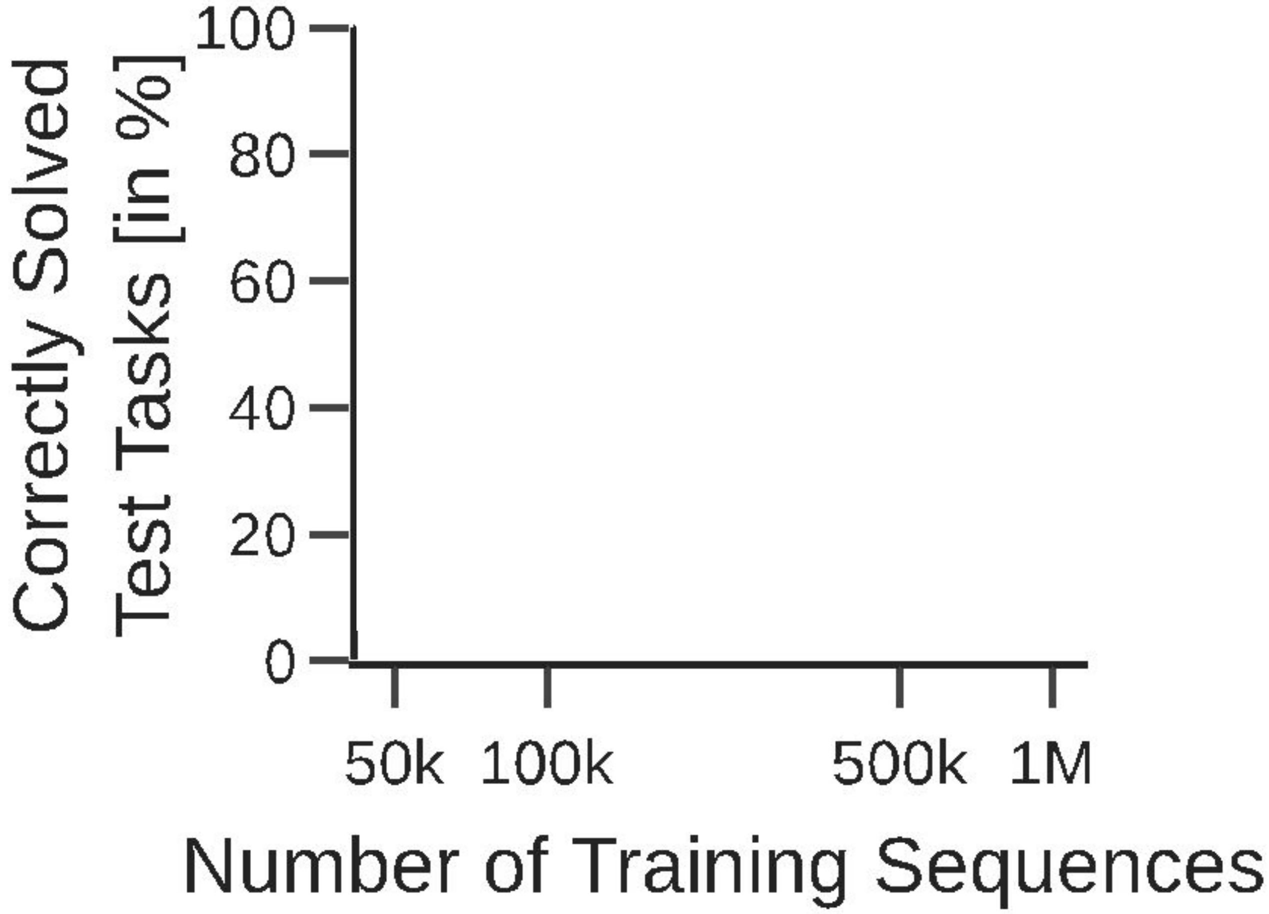
bos
create 0 2 c0 c3 close 0 2 c0
c3 create 0 1 c1 c2 close 0 1
c1 c2 create 0 0 c2 c1 create
1 1 c2 c1 close 0 0 c2 c1
create 1 0 c3 c0 close 1 0 c3
c0
plan 0 2
plan 0 1
plan 0 0
plan 1 0
eos

Training Method

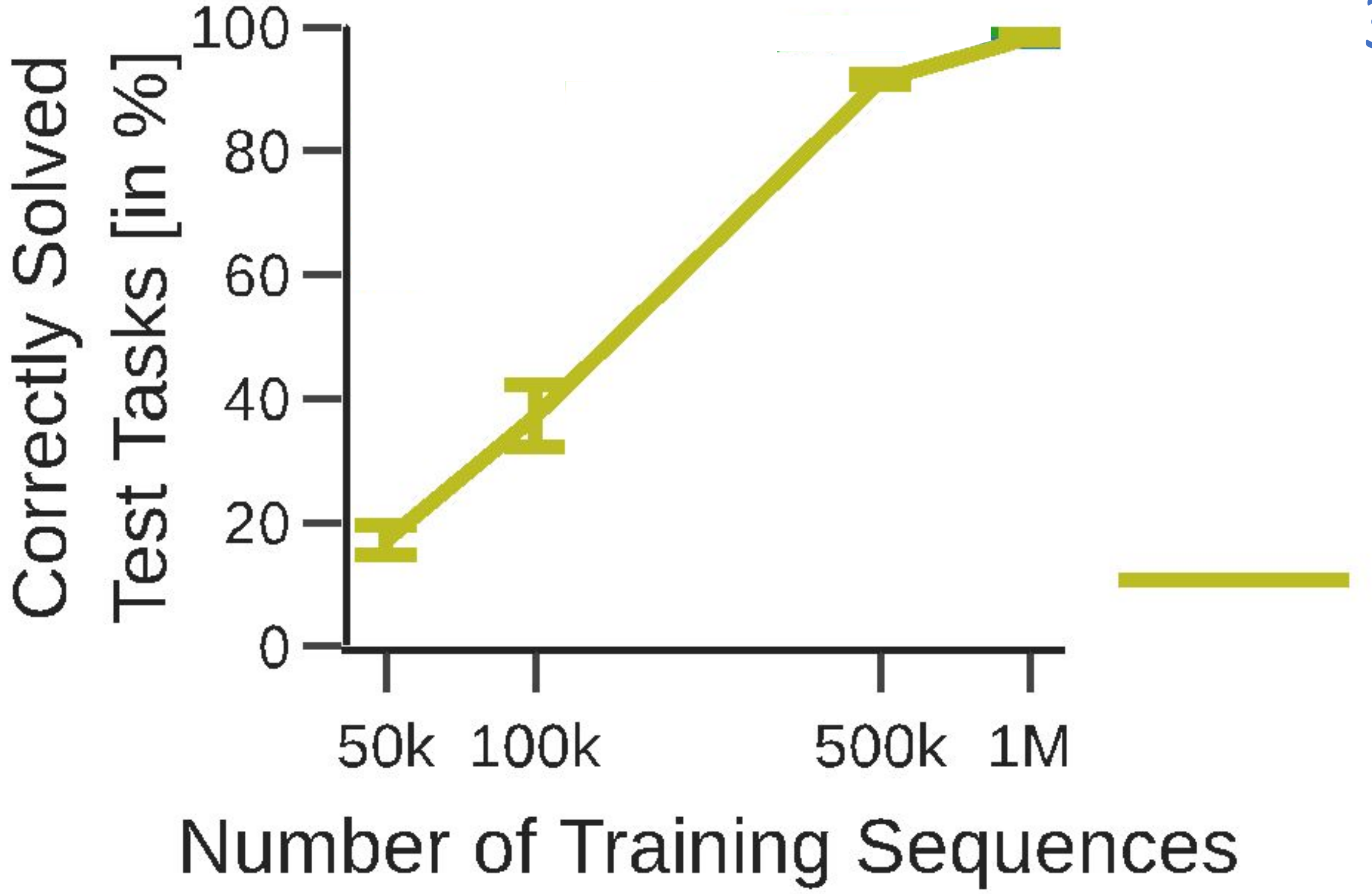
Train a Transformer to predict the next token via teacher forcing.



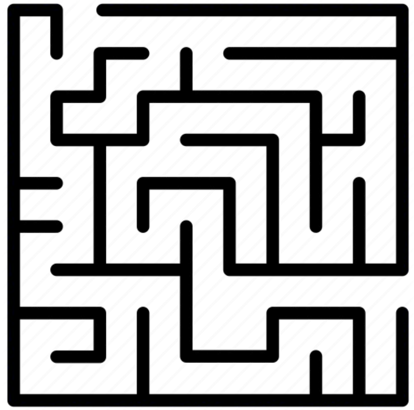
Search-Augmented vs. Solution-Only Models



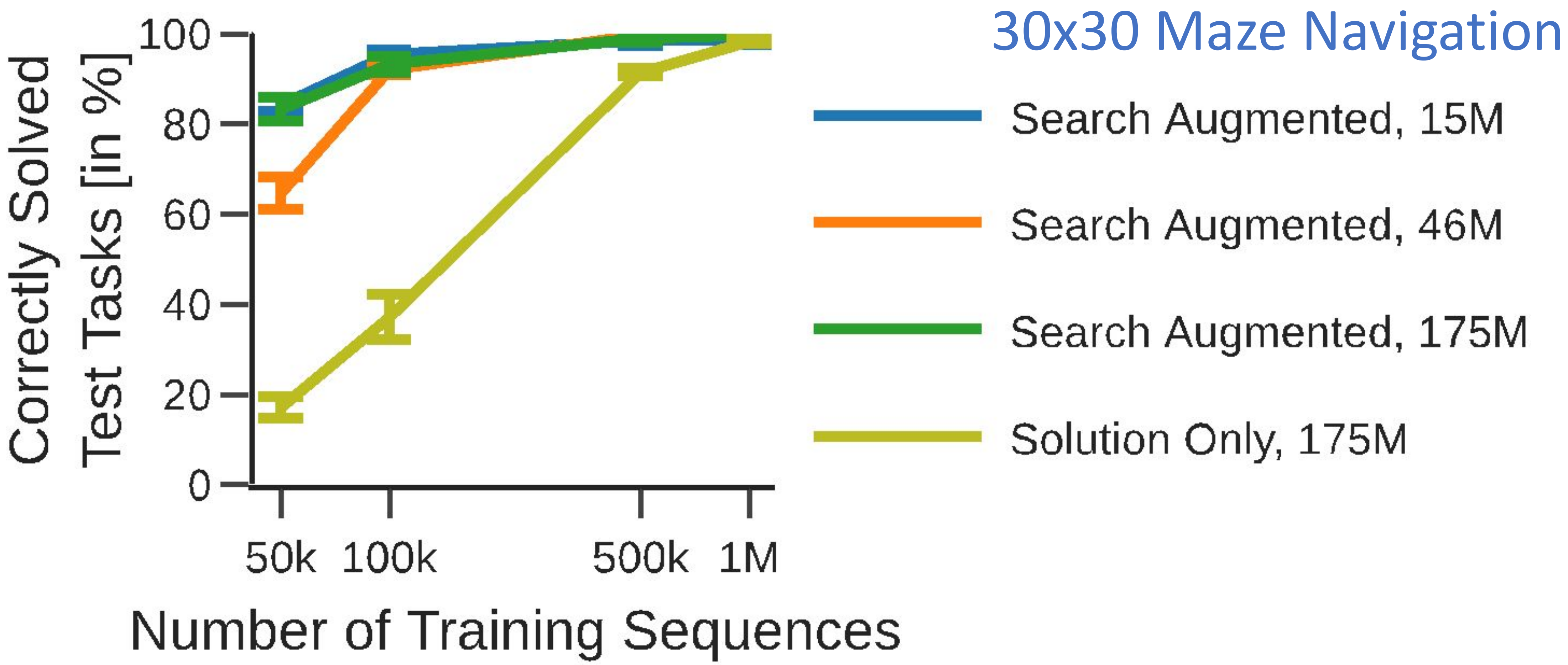
Search-Augmented vs. Solution-Only Models



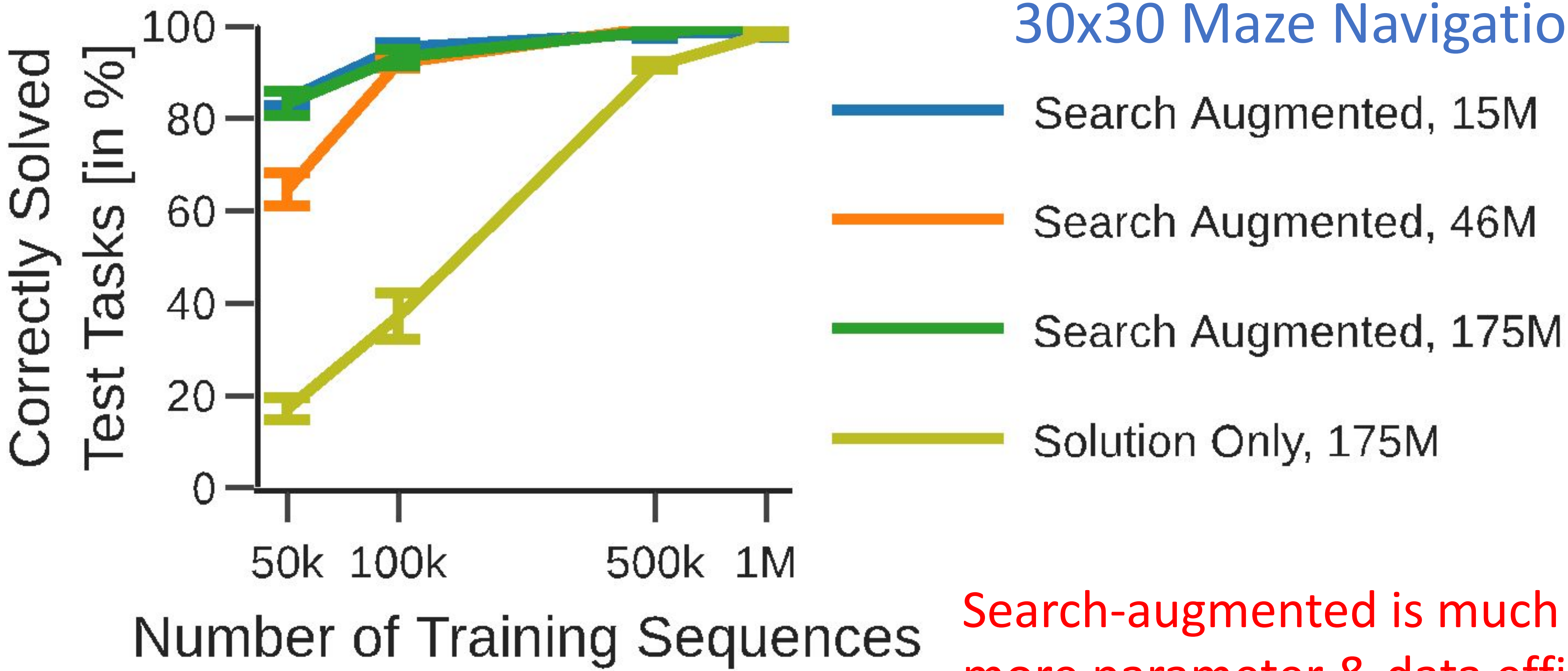
30x30 Maze Navigation



Search-Augmented vs. Solution-Only Models



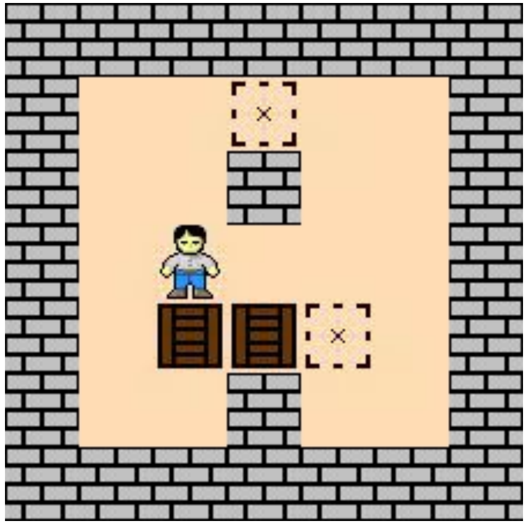
Search-Augmented vs. Solution-Only Models



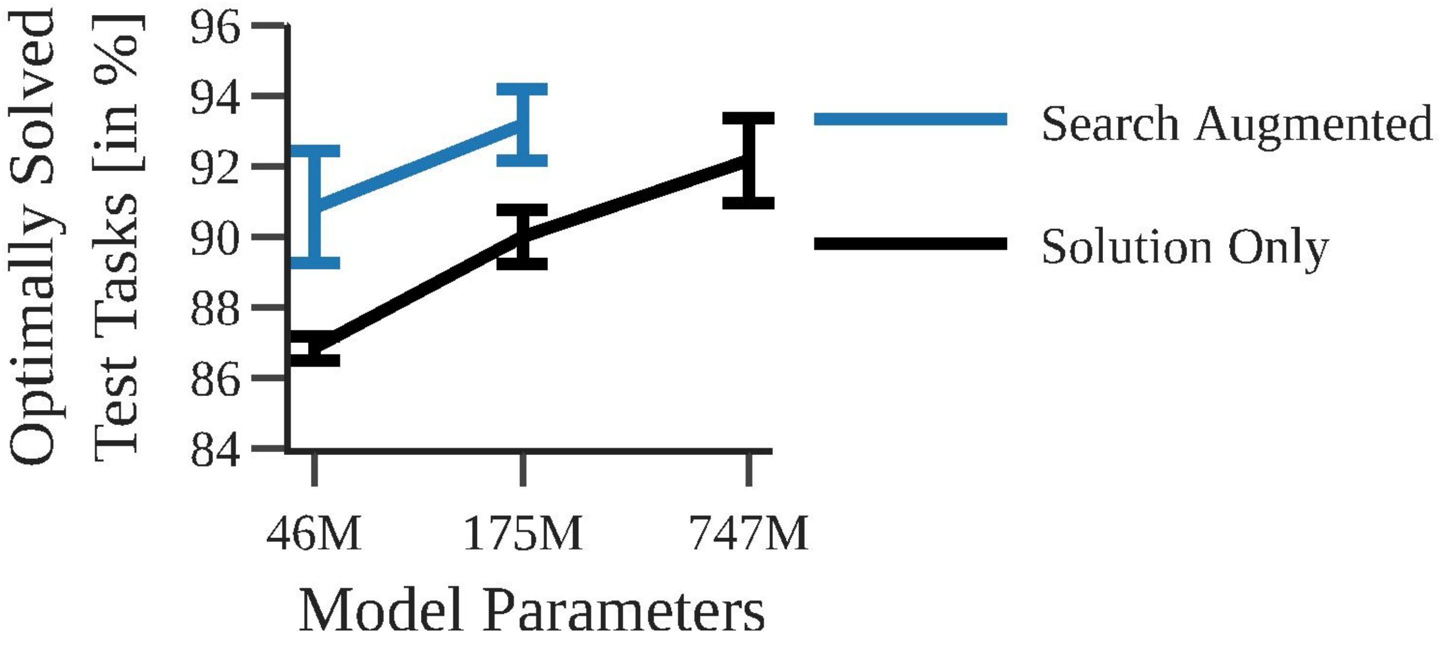
30x30 Maze Navigation

Search-augmented is much more parameter & data efficient!

Search-Augmented vs. Solution-Only Models



Sokoban



Search-augmented is much more parameter & data efficient!

How to go beyond?

Imitation Learning → **Fine-tuning**

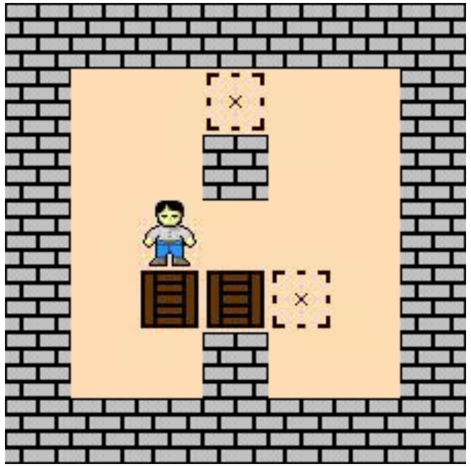
Using solver's trace to train the Transformer with teacher forcing

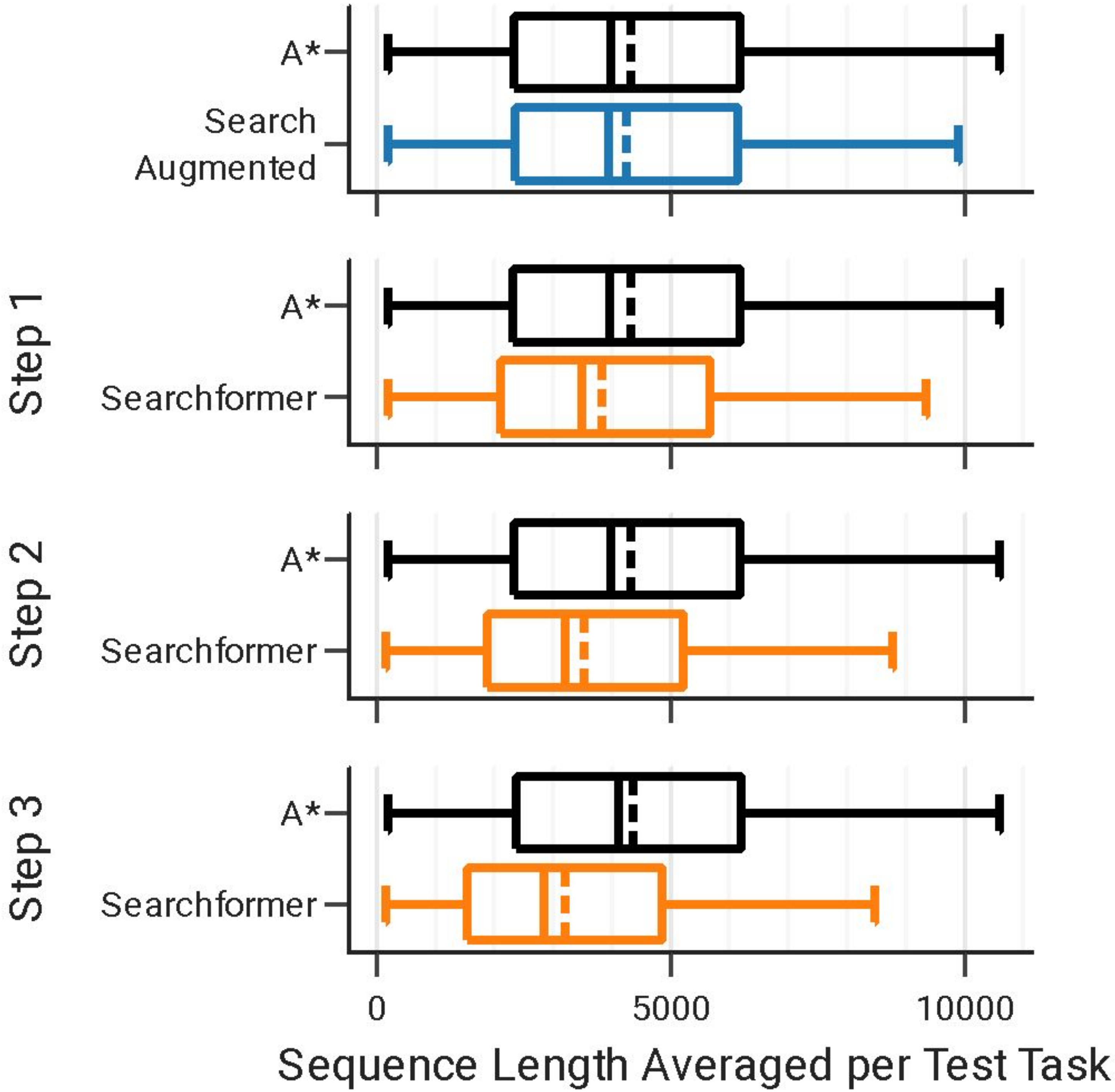
Fine-tune the model to achieve **shorter** trace but still leads to **optimal** plan! (Reinforcement Learning task)

↓ Search-augmented Models

↓ Searchformer

Beyond A*: Improving search dynamics via bootstrapping





facebook Artificial Intelligence

Improving search dynamics via bootstrapping

Params.	Model	ILR-on-solved	ILR-on-optimal
45M	Solution only	–	–
	Search augmented	0.908 ± 0.020	0.919 ± 0.019
	Searchformer, step 1	1.054 ± 0.025	1.062 ± 0.015
	Searchformer, step 2	1.158 ± 0.025	1.181 ± 0.012
	Searchformer, step 3	1.292 ± 0.044	1.343 ± 0.067
175M	Solution only	–	–
	Search augmented	0.925 ± 0.010	0.933 ± 0.011
757M	Solution only	–	–

Repeated bootstrapping increases the *Improved Length Ratio (ILR)*

Improving search dynamics via bootstrapping

Params.	Model	Solved (%)	Optimal (%)
45M	Solution only	90.3 ±1.0	86.8 ±0.3
	Search augmented	92.5 ±1.0	90.8 ±1.6
	Searchformer, step 1	95.5 ±1.0	93.5 ±1.0
	Searchformer, step 2	96.0 ±0.5	93.4 ±0.6
	Searchformer, step 3	95.5 ±0.8	93.7 ±1.6
175M	Solution only	95.7 ±0.2	90.0 ±0.8
	Search augmented	95.2 ±0.9	93.2 ±1.0
757M	Solution only	96.5 ±0.1	92.2 ±1.2

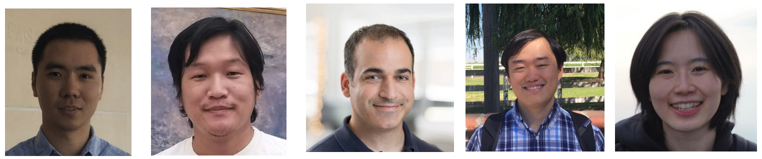
Fine-tuning improves performance initially.

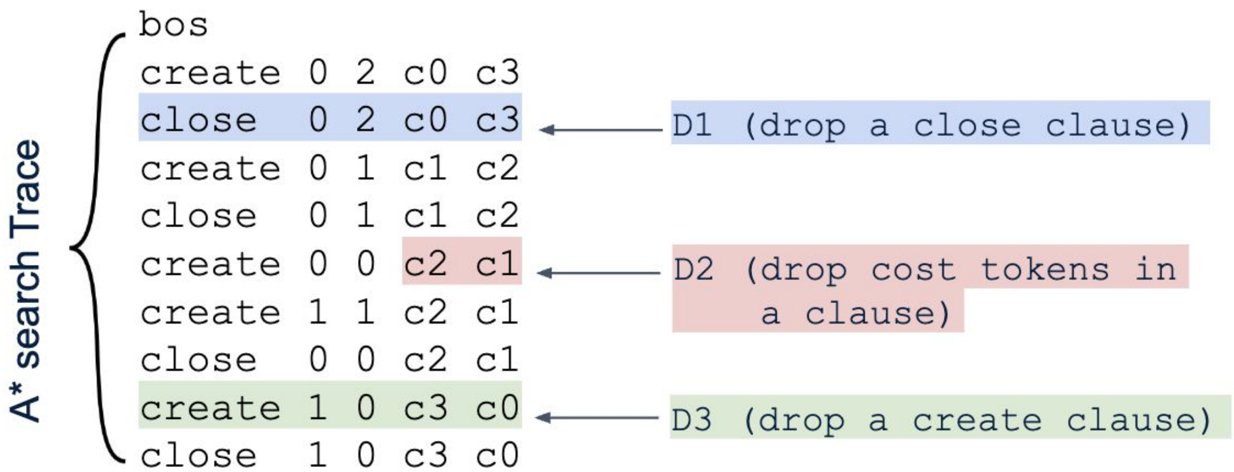
Improving search dynamics via bootstrapping

Params.	Model	Solved (%)	Optimal (%)
45M	Solution only	90.3 ±1.0	86.8 ±0.3
	Search augmented	92.5 ±1.0	90.8 ±1.6
	Searchformer, step 1	95.5 ±1.0	93.5 ±1.0
	Searchformer, step 2	96.0 ±0.5	93.4 ±0.6
	Searchformer, step 3	95.5 ±0.8	93.7 ±1.6
175M	Solution only	95.7 ±0.2	90.0 ±0.8
	Search augmented	95.2 ±0.9	93.2 ±1.0
757M	Solution only	96.5 ±0.1	92.2 ±1.2

Searchformer outperforms largest solution-only model.

DualFormer (Searchformer v2)



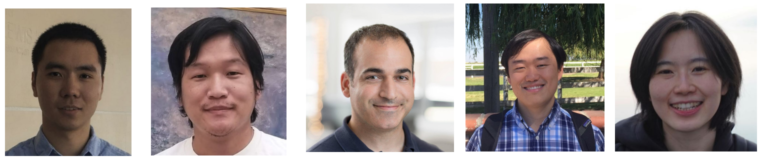


Structured Trace Dropping Strategies

```
LvL 1 = D1                      // drop all the close clauses

LvL 2 = D1 + D2                 // drop all the close clauses
                                          + all the cost tokens

Lvl 3 = D1 + D2 + sampled D3 // Lvl 2 + drop some
                                              create clauses

Lvl 4 = drop the entire trace
```


[D. Su et al, *Dualformer: Controllable Fast and Slow Thinking by Learning with Randomized Reasoning Traces*, arXiv'24]

DualFormer (Searchformer v2)



	Method	Avg Trace Length	1-Optimal-64 / 3-Optimal-64	1-Solved-64 / 3-Solved-64	SWC	Diversity
Maze 15 x 15	Dualformer (auto)	222	99.7 / 99.4	99.9 / 99.8	0.999	12.52
	Complete-Trace	495	94.6 / 90.1	96.7 / 93.0	0.964	7.60
	Solution-Only	-	72.0 / 68.9	82.7 / 80.1	0.610	1.52
Maze 20 x 20	Dualformer (auto)	351	99.5 / 98.6	99.9 / 99.3	0.997	20.28
	Complete-Trace	851	98.3 / 95.5	98.8 / 93.0	0.987	14.53
	Solution-Only	-	56.3 / 52.0	71.9 / 67.5	0.690	1.52
Maze 25 x 25	Dualformer (auto)	427	98.6 / 96.9	99.8 / 99.0	0.998	24.81
	Complete-Trace	1208	95.2 / 85.7	97.0 / 90.4	0.968	18.85
	Solution-Only	-	39.7 / 34.7	60.3 / 55.4	0.570	1.9
Maze 30 x 30	Dualformer (auto)	617	96.6 / 92.1	98.4 / 97.7	0.989	24.42
	Complete-Trace	1538	93.3 / 82.4	95.9 / 88.1	0.964	7.60
	Solution-Only	-	30.0 / 26.0	54.1 / 47.8	0.500	1.86
Sokoban	Dualformer (auto)	494	94.0 / 90.0	97.4 / 94.7	0.979	4.97
	Complete-Trace	3600	92.9 / 84.4	94.7 / 89.0	0.944	2.91
	Solution-Only	-	86.8 / 83.4	92.8 / 90.0	0.919	1.24

Dualformer **automatically** switches between fast mode (System 1) and slow mode (System 2) and works **better** for **dedicated** models on either modes.

Fast mode performance

	Method	1-Optimal-64 / 3-Optimal-64	1-Solved-64 / 3-Solved-64	SWC	Diversity
Maze 15x15	Dualformer (fast)	91.8 / 87.6	97.1 / 94.8	0.960	9.05
	Solution-Only	72.0 / 68.9	82.7 / 80.1	0.610	1.52
Maze 20x20	Dualformer (fast)	90.9 / 84.0	97.0 / 94.0	0.960	17.27
	Solution-Only	56.3 / 52.0	71.9 / 67.5	0.690	1.52
Maze 25x25	Dualformer (fast)	83.9 / 72.9	95.5 / 90.6	0.940	21.23
	Solution-Only	39.7 / 34.7	60.3 / 55.4	0.570	1.9
Maze 30x30	Dualformer (fast)	80.0 / 66.0	91.8 / 85.7	0.906	18.23
	Solution-Only	30.0 / 26.0	54.1 / 47.8	0.500	1.86
Sokoban	Dualformer (fast)	97.3 / 94.4	94.8 / 90.0	0.970	4.92
	Solution-Only	86.8 / 83.4	92.8 / 90.0	0.919	1.24

Slow mode performance

	Method	Avg Trace Length	1-Optimal-64 / 3-Optimal-64	1-Solved-64 / 3-Solved-64	SWC	Diversity
Maze 15 x 15	Dualformer (slow)	278	99.6 / 99.2	99.9 / 99.9	0.999	12.54
	Complete-Trace	495	94.6 / 90.1	96.7 / 93.0	0.964	7.60
Maze 20 x 20	Dualformer (slow)	439	98.9 / 97.8	99.9 / 99.7	0.998	18.86
	Complete-Trace	851	98.3 / 95.5	98.8 / 93.00	0.987	14.53
Maze 25 x 25	Dualformer (slow)	589	99.9 / 97.2	99.7 / 99.3	0.997	25.05
	Complete-Trace	1208	95.2 / 85.7	97.0 / 90.4	0.968	18.85
Maze 30 x 30	Dualformer (slow)	854	97.6 / 93.2	99.5 / 98.2	0.993	25.77
	Complete-Trace	1538	93.3 / 82.4	95.9 / 88.1	0.964	7.60
Sokoban	Dualformer (slow)	1482	94.5 / 87.6	97.4 / 94.1	0.970	4.66
	Complete-Trace	3600	92.9 / 84.4	94.7 / 89.0	0.944	2.91

Math Problems

Question 1

In Pascal's Triangle, the third row has four elements, and the sum of these elements is 8. What is the sum of the elements in the tenth row? (Note that we are counting the top row with a single 1 as the 0th row).

Llama-3-Aug (baseline) Here is a detailed step-by-step explanation of the reasoning:

- Step 1: Understand the problem and the given information. We are asked to find the sum of the elements in the tenth row of Pascal's Triangle, given that the third row has four elements and their sum is 8.
- Step 2: Recall the formula for the sum of the elements in a row of Pascal's Triangle. The sum of the elements in the nth row is given by the formula: $\binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{n} = 2^n$.
- Step 3: Identify the relationship between the sum of the elements in a row and the sum of the elements in the previous row. Since the sum of the elements in the nth row is 2^n, the sum of the elements in the (n-1)th row is 2^{n-1}.
- Step 4: Recognize that the sum of the elements in the third row is 8, which is equal to 2^3. This is because the sum of the elements in the third row is $\binom{3}{0} + \binom{3}{1} + \binom{3}{2} = 1 + 3 + 3 = 7$, and the extra 1 comes from the top row, which is counted as the 0th row.
- Step 5: Use the relationship between the sums of the elements in consecutive rows to find the sum of the elements in the tenth row. Since the sum of the elements in the third row is 2^3, and the sum of the elements in the (n-1)th row is 2^{n-1}, we can set up the equation: $2^3 = 2^{10-1}$.
- Step 6: Solve the equation for n. Taking the logarithm of both sides, we get: $3 \log 2 = (10 - 1) \log 2$. Simplifying, we get: $3 \log 2 = 9 \log 2$, which implies that $\log 2 = 0$. This is a contradiction, since $\log 2$ is a positive number.
- Step 7: Conclude that the equation $2^3 = 2^{10-1}$ has no solution, and therefore, the sum of the elements in the tenth row is not equal to 2^{10-1}.
- Step 8: Recognize that the sum of the elements in the tenth row is still given by the formula: $\binom{10}{0} + \binom{10}{1} + \binom{10}{2} + \cdots + \binom{10}{10} = 2^{10}$.

The answer is: 1024.

Baseline

Llama-3-Aug (p=0.4, ours) Here is a detailed step-by-step explanation of the reasoning:

- Step 1: Understand the problem and the given information. We are asked to find the sum of the elements in the tenth row of Pascal's Triangle, given that the third row has four elements and their sum is 8.
- Step 2: Recall the properties of Pascal's Triangle. The elements in each row of Pascal's Triangle are the binomial coefficients, which represent the number of ways to choose k items from a set of n items. The sum of the elements in each row is equal to 2^n, where n is the row number.
- Step 3: Identify the relationship between the sum of the elements in the third row and the sum of the elements in the tenth row. Since the sum of the elements in each row is equal to 2^n, we can write the sum of the elements in the third row as $2^3 = 8$. Similarly, the sum of the elements in the tenth row is 2^{10}.
- Step 4: Calculate the sum of the elements in the tenth row. $2^{10} = 1024$.

The answer is: 1024.

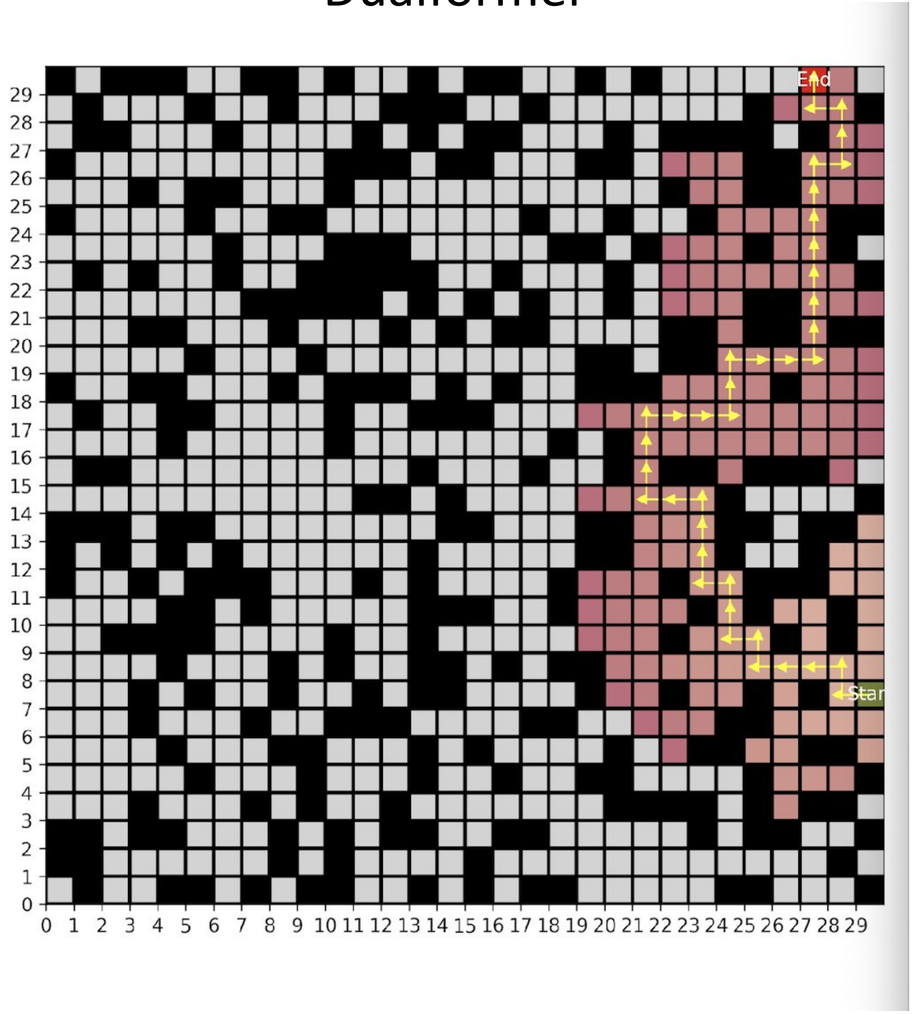
Dualformer

Math Problems

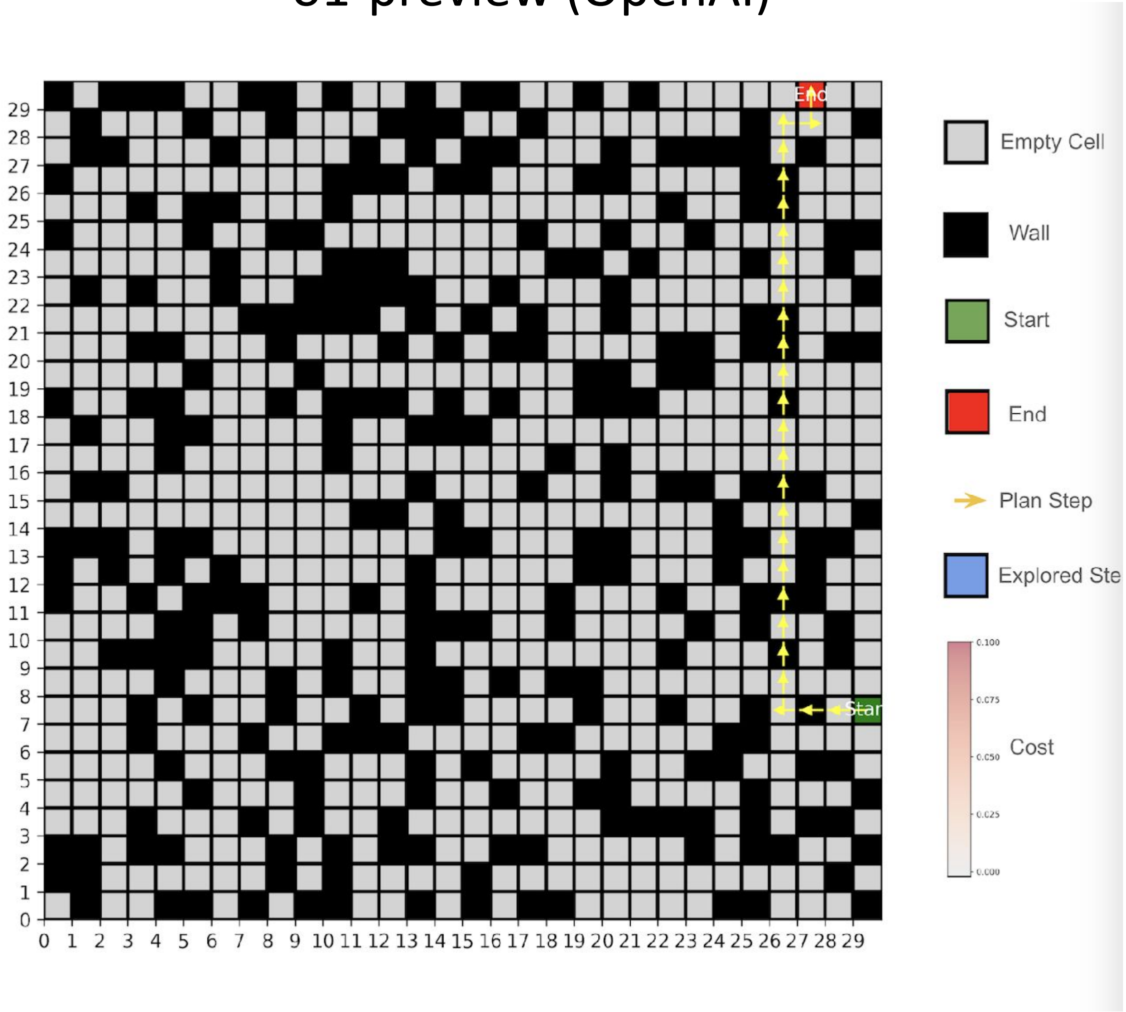
Model	Dataset & Dropping Prob	Greedy@1(%) (slow / fast)	Trace Length	Pass@20(%) (slow / fast)	Trace Length
Mistral-7B	Aug-MATH (baseline)	16.9 / 9.6	527 / -	59.6 / 29.8	521 / -
	Aug-MATH (p=0.1)	18.6 / 11.3	508 / -	61.6 / 32.0	479 / -
	Aug-MATH (p=0.2)	17.8 / 11.2	477 / -	61.4 / 31.9	470 / -
	Aug-MATH (p=0.3)	17.8 / 11.8	497 / -	61.9 / 31.7	466 / -
	Aug-MATH (p=0.4)	17.0 / 11.0	434 / -	56.4 / 28.9	397 / -
	MATH	13.1 / 8.5	290 / -	53.0 / 29.4	227 / -
Llama-3-8B	Aug-MATH (baseline)	19.7 / 13.1	548 / -	62.7 / 35.6	535 / -
	Aug-MATH (p=0.1)	20.1 / 13.3	544 / -	63.4 / 36.2	522 / -
	Aug-MATH (p=0.2)	20.5 / 13.8	525 / -	63.9 / 36.7	497 / -
	Aug-MATH (p=0.3)	20.5 / 13.5	515 / -	63.4 / 37.5	474 / -
	Aug-MATH (p=0.4)	20.4 / 13.5	490 / -	63.4 / 37.2	450 / -
	MATH	13.3 / 12.6	432 / -	52.8 / 35.5	332 / -

DualFormer

Dualformer

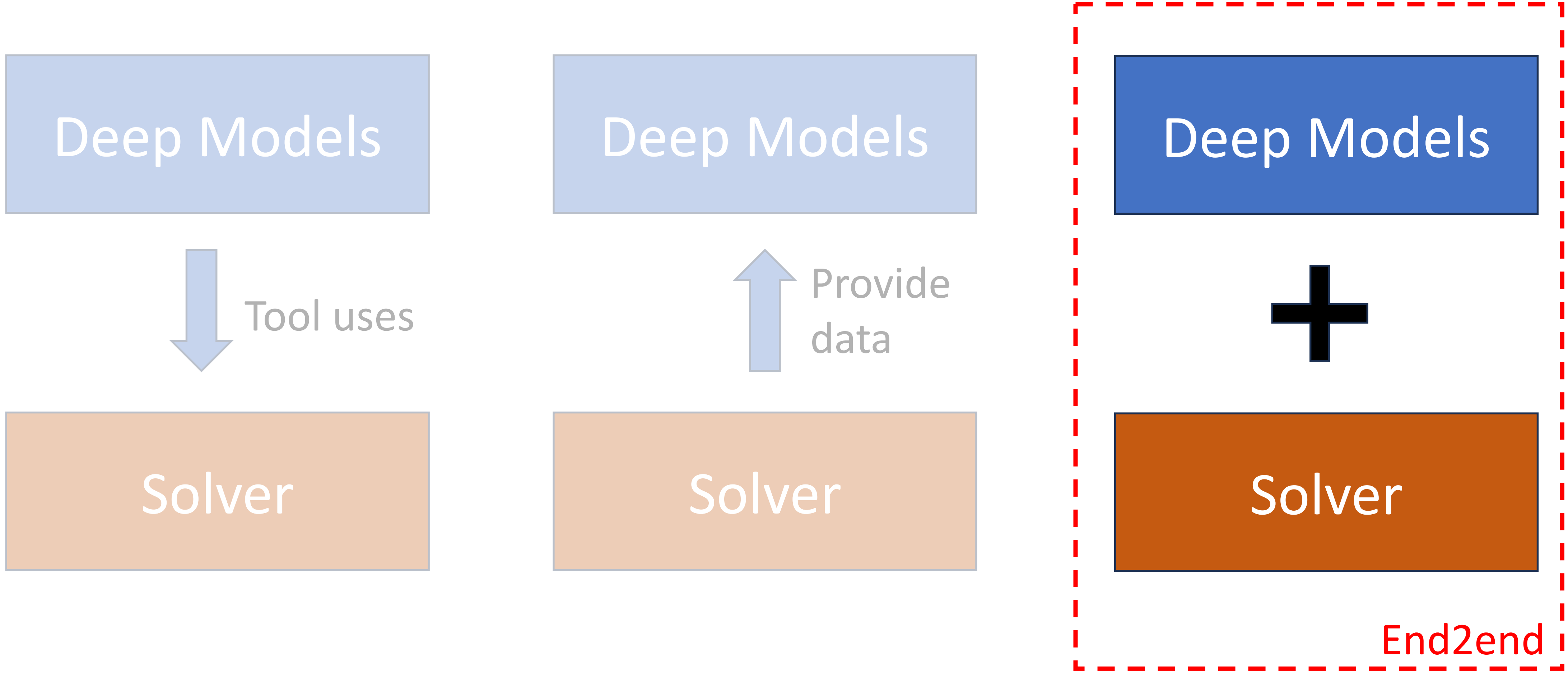


o1-preview (OpenAI)

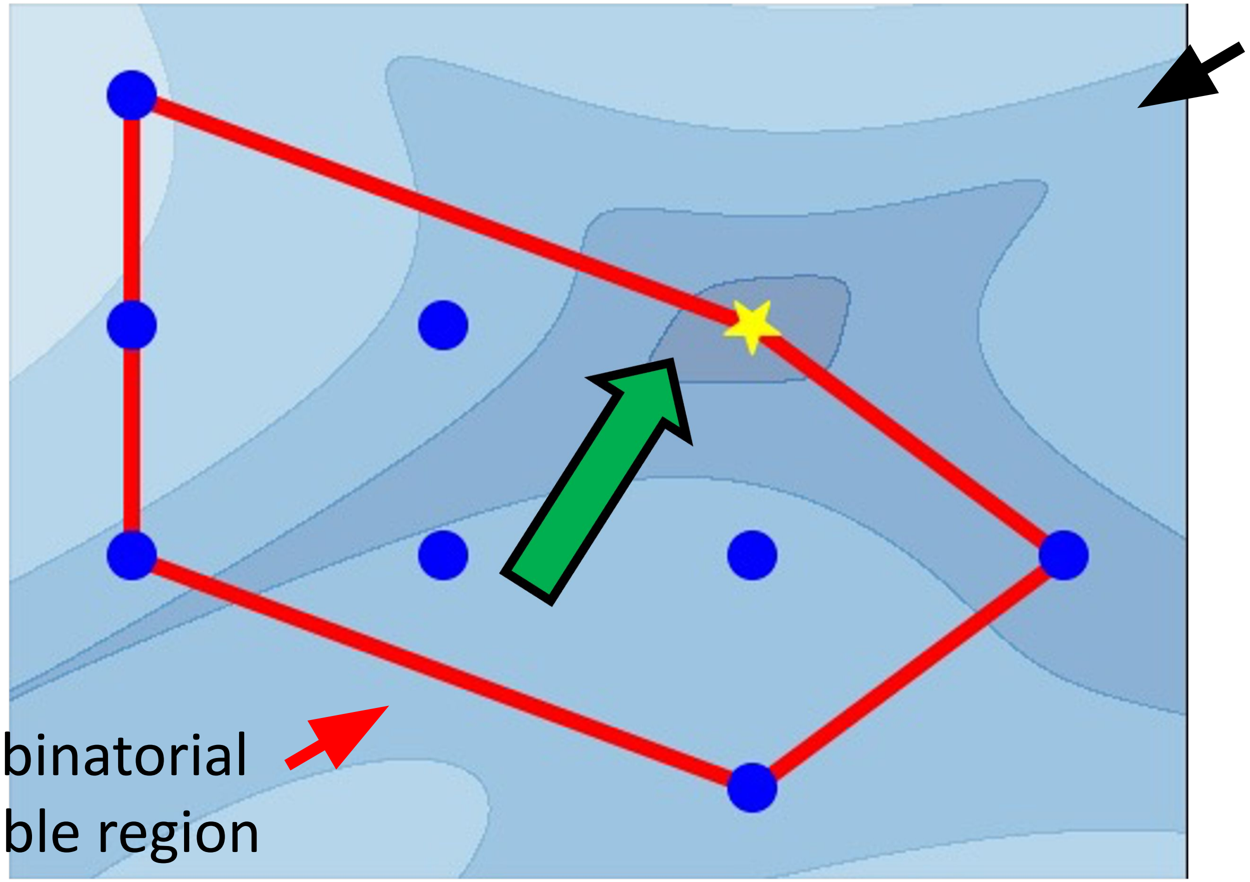


facebc

Option **Two**: Hybrid Systems



Nonlinear objective with combinatorial constraints

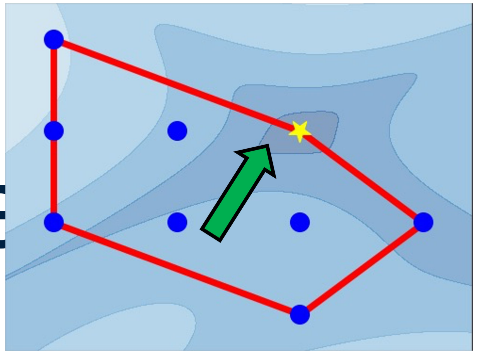


Nonlinear + differentiable objective $f(\boldsymbol{x})$

Combinatorial feasible region

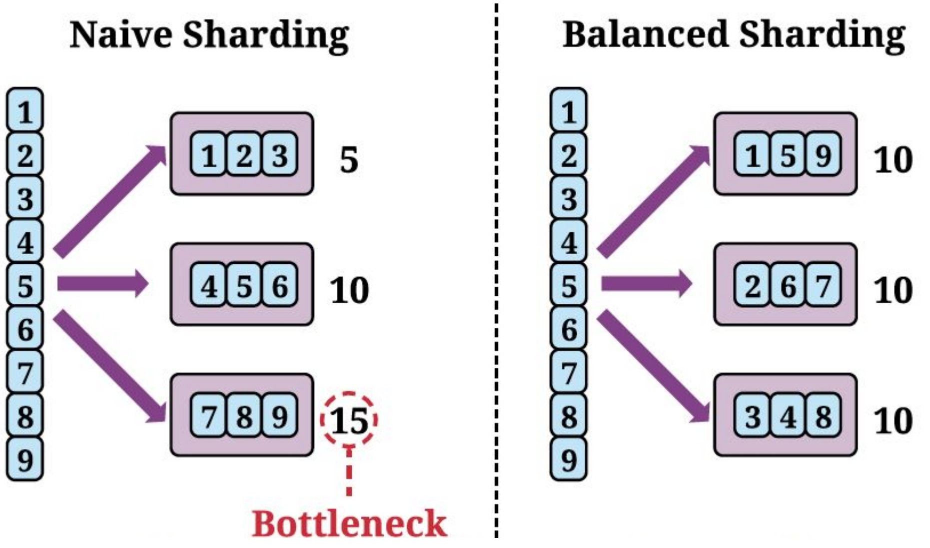
- Real-world domains:
  - Computer system planning
  - Designing photonic devices
  - Throughput optimization
  - Antenna design
  - Energy grid

Example: Embedding Table Placeme



Given:

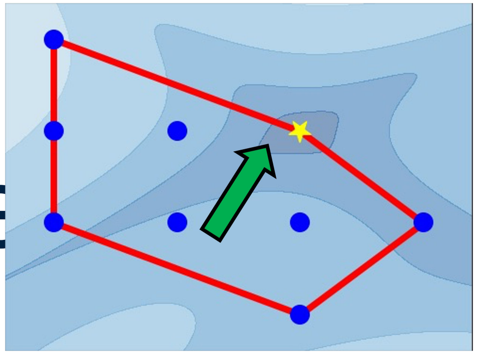
- k tables
- n identical devices
- Table i has memory requirement m_i
- Device j has memory capacity M_j



Find

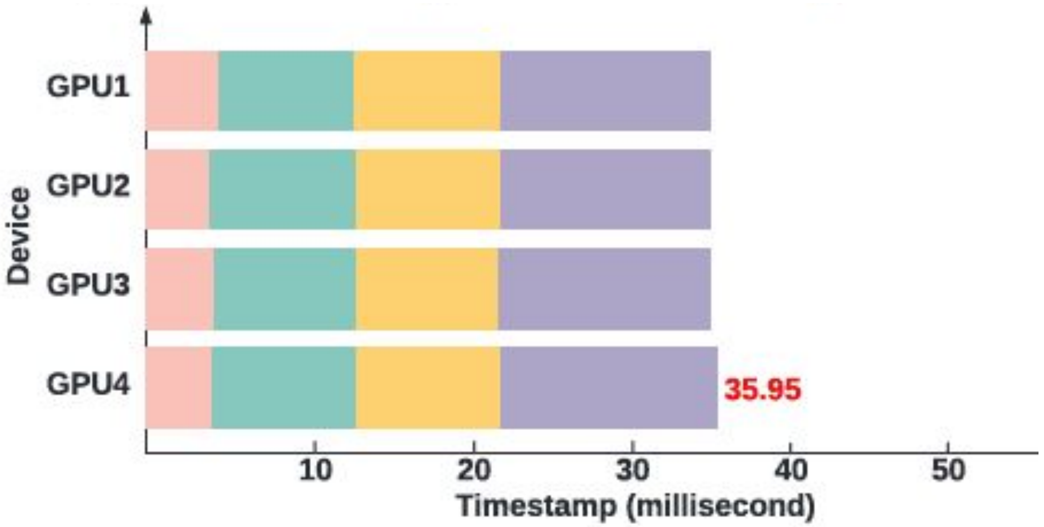
- Allocation of tables to devices observing device memory limits
- Minimize latency which is estimated by a neural network (capturing nonlinear interactions)

Example: Embedding Table Placement



Given:

- k tables
- n identical devices
- Table i has memory requirement m_i
- Device j has memory capacity M_j

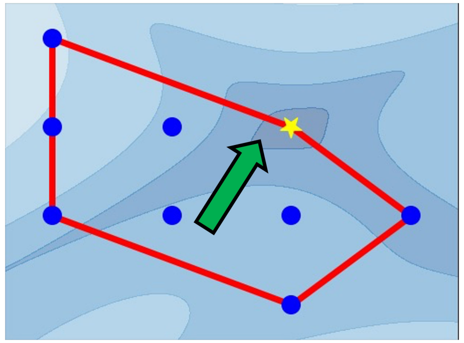


Formulation

$$\mathrm{Min}_x\ \boldsymbol{L}(\{x_{ij}\}) \quad \text{s.t.}\ \sum_i x_{ij} m_i \le M_j, \quad \sum_j x_{ij} = 1, \quad x_{ij} \in \{0,1\}$$

$\boldsymbol{L}$ is nonlinear due to system issues (e.g., batching, communication, etc)

Solve the Combinatorial Problem in the Latent Space

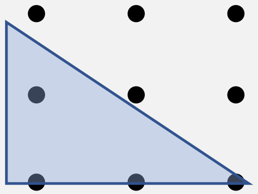


Original Space

Nonlinear optimization with combinatorial constraints

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}; \boldsymbol{y})$$

$$\text{s.t } \boldsymbol{x} \in \Omega =$$



combinatorial constraints

Predict surrogate cost $\boldsymbol{c} = \boldsymbol{c}(\boldsymbol{y})$

Latent Space

Surrogate optimization

$$\boldsymbol{x}^*(\boldsymbol{y}) = \arg\min_{\boldsymbol{x}} \boldsymbol{c}(\boldsymbol{y})^T \boldsymbol{x}$$

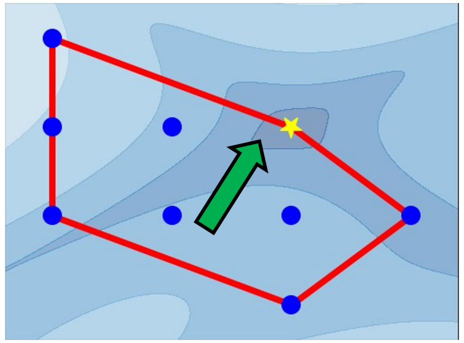
$$\text{s.t } \boldsymbol{x} \in \Omega$$

solved by existing combinatorial solvers

$\boldsymbol{x}^*(\boldsymbol{y})$ optimizes $f(\boldsymbol{x}; \boldsymbol{y})$ as much as possible

[A. Ferber et al, *SurCo: Learning Linear Surrogates For Combinatorial Nonlinear Optimization Problems*, ICML'23 and **outstanding paper** in SODS workshop]

Solve the Combinatorial Problem in the Latent Space

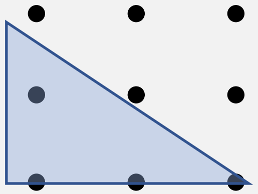


Original Space

Nonlinear optimization with combinatorial constraints

$$\min_{\boldsymbol{x}} f(\boldsymbol{x};\boldsymbol{y})$$

$$\text{s.t } \boldsymbol{x} \in \Omega =$$



combinatorial constraints

Predict surrogate cost $\boldsymbol{c} = \boldsymbol{c}(\boldsymbol{y})$

Latent Space

Surrogate optimization

$$\boldsymbol{x}^*(\boldsymbol{y}) = \operatorname*{argmin}_{\boldsymbol{x}} \boldsymbol{c}(\boldsymbol{y})^T \boldsymbol{x}$$

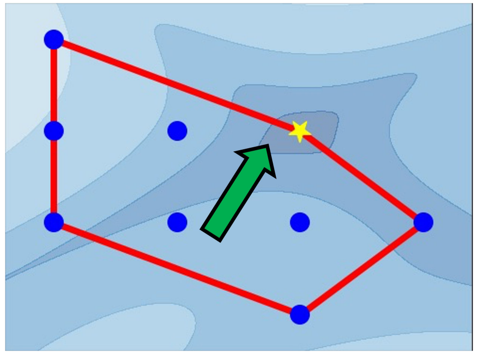
$$\text{s.t } \boldsymbol{x} \in \Omega$$

solved by existing combinatorial solvers

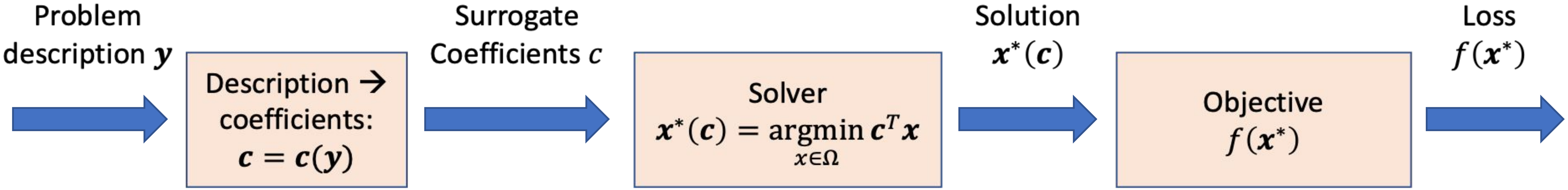
$\boldsymbol{x}^*(\boldsymbol{y})$ optimizes $f(\boldsymbol{x};\boldsymbol{y})$ as much as possible

Proposal: gradient-based optimization

SurCo: Surrogate combinatorial opt

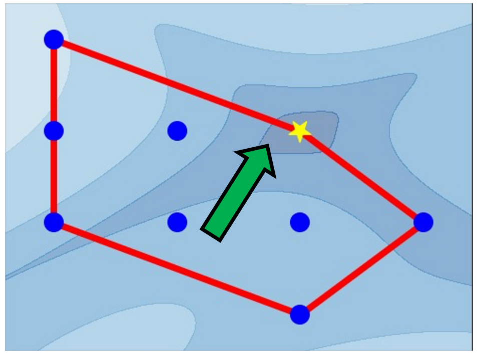


- Use surrogate MILP to solve original problem
- Find linear coefficients c such that $\arg\min_{x \in \Omega} f(x) = \arg\min_{x \in \Omega} c^T x$

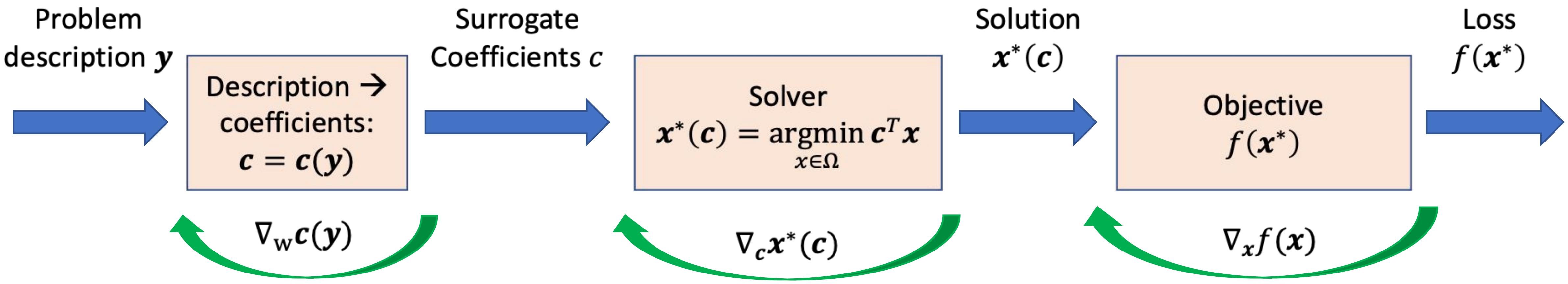


[A. Ferber et al, *SurCo: Learning Linear Surrogates For Combinatorial Nonlinear Optimization Problems*, ICML'23 and **outstanding paper** in SODS workshop]

Gradient-based Optimization



- Use surrogate MILP to solve original problem
- Find linear coefficients c such that $\underset{x \in \Omega}{\operatorname{argmin}} f(x) = \underset{x \in \Omega}{\operatorname{argmin}} c^T x$



Assumed differentiable

Recent work on differentiable optimization

Differentation of blackbox optimizers

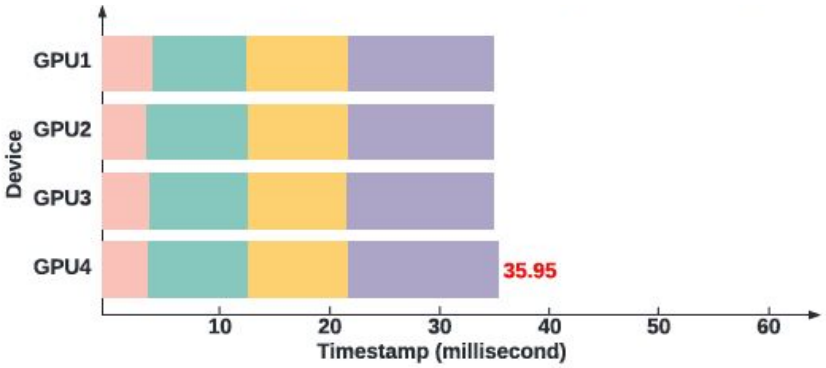
CVXPYLayers

MIPaaL

Etc.

Assumed differentiable

Embedding Table Sharding

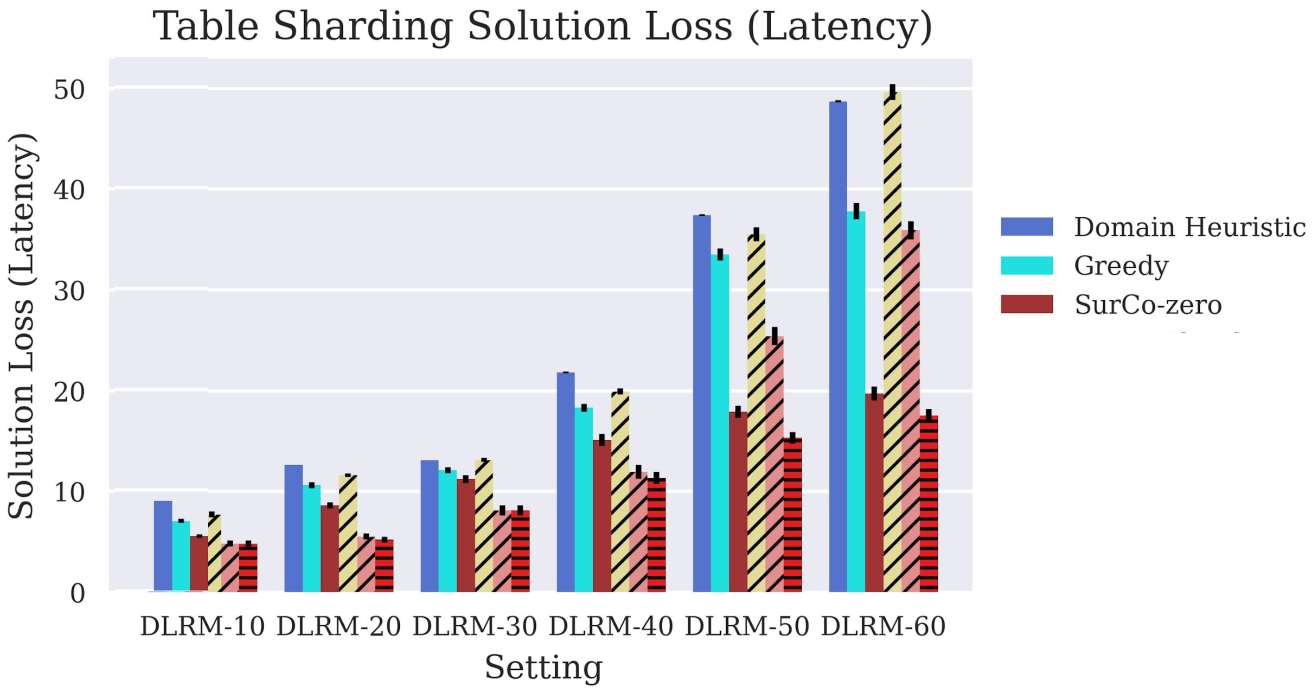


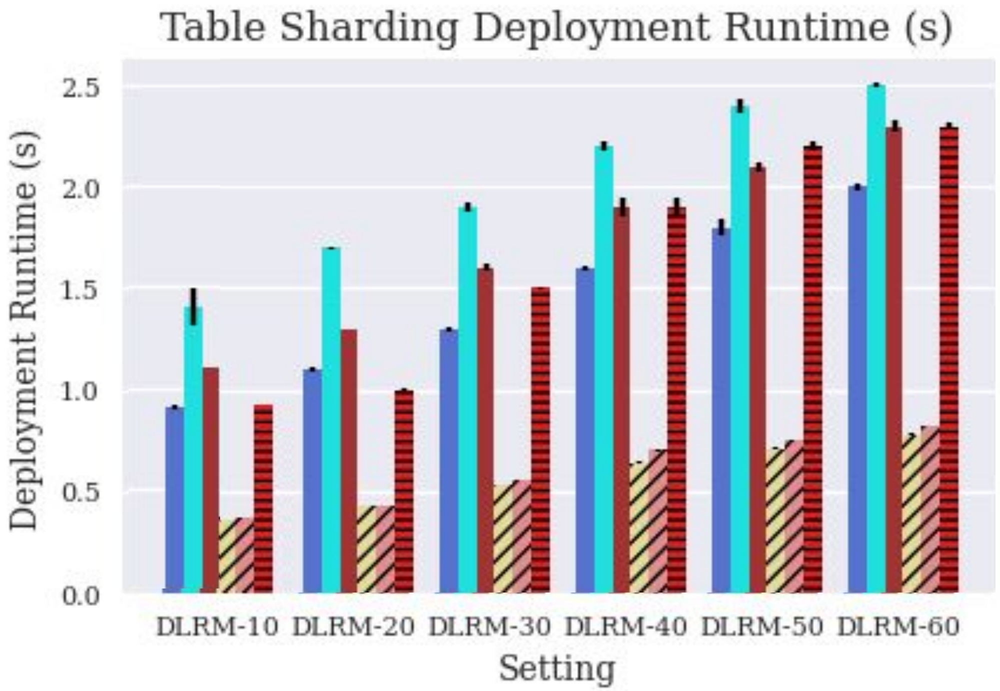
- Public **D**eep **L**earning **R**ecommendation **M**odel (DLRM dataset) placing between 10 to 60 tables on 4 GPUs

- Baseline: Greedy
- SoTA: RL approach Dreamshard[1]
- SurCo: Surrogate NN model learned via CVXPYLayers (differentiable LP Solver)

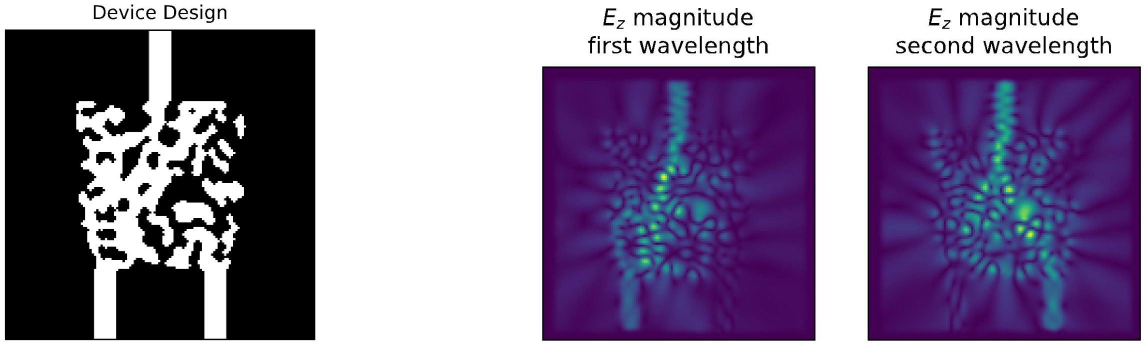
[1] Zha et al. NeurIPS 2022
Dataset: https://github.com/facebookresearch/dlrm_datasets

Results – Table Sharding





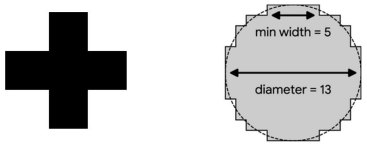
Inverse Photonic Desig



- Design physically-viable devices that take light waves and routes different wavelengths to correct locations

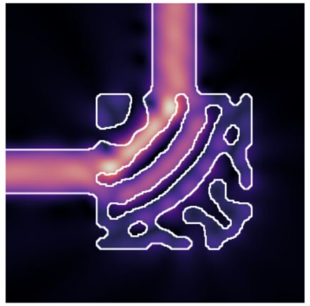
$$\mathcal{L}(S) = \left(\left\| \text{softplus}\left(g \frac{|S|^2 - |S_{\text{cutoff}}|^2}{\min(w_{\text{valid}})} \right) \right\|_2 \right)^2$$

- Device design misspecification loss $f(\boldsymbol{x})$ computed by differentiable electromagnetic simulator
- Feasible solution: the design must be the union of brush pattern
  - x = binary_opening(x, brush)
  - x = ~binary_opening(~x, brush)

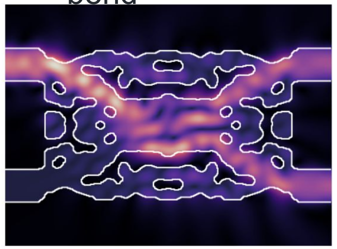


Inverse Photonic Design

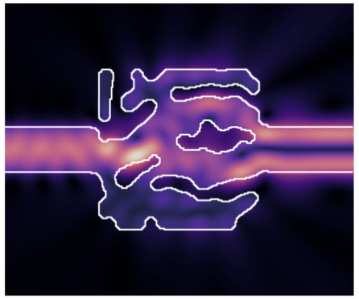
- Dataset: Ceviche Challenges[1]
- Most baselines don't work here due to combinatorial constraints
- SoTA: Brush-based algorithm [1]
- SurCo: Surrogate learned via blackbox differentiation[2] of brush solver



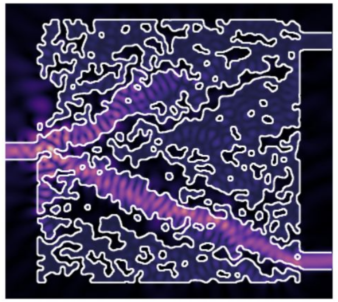
Waveguide bend



Beam splitter



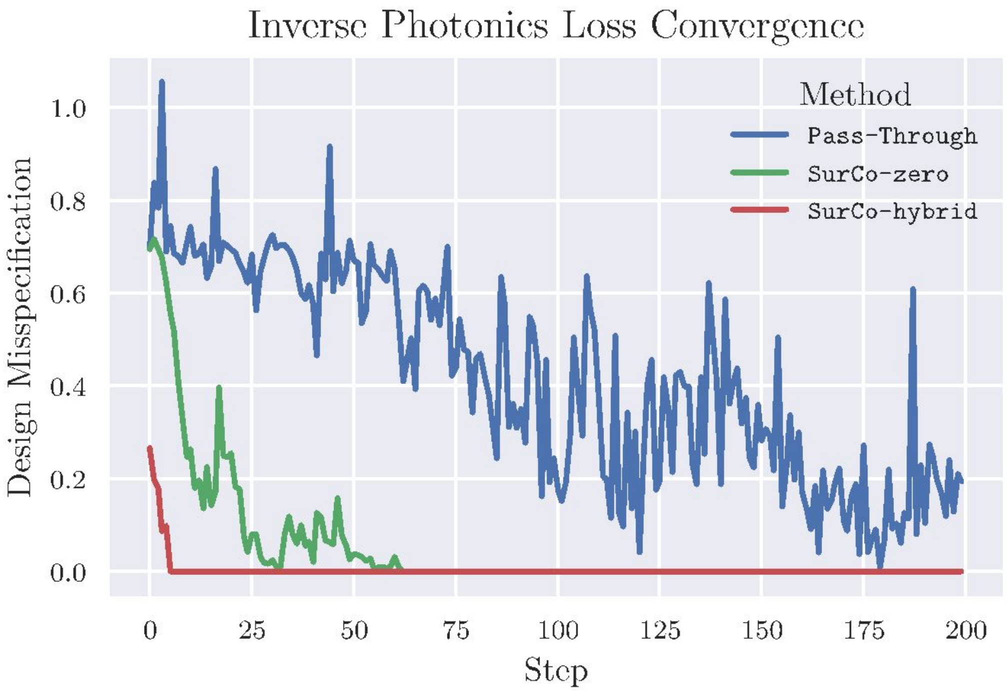
Mode converter



Wavelength division multiplexe

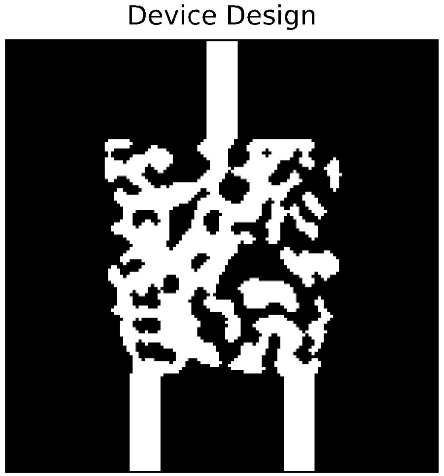
[1]Schubert et al. ACS Photonics 2022
[2]Vlastelica et al. ICLR 2019
Dataset: https://github.com/google/ceviche-challenges

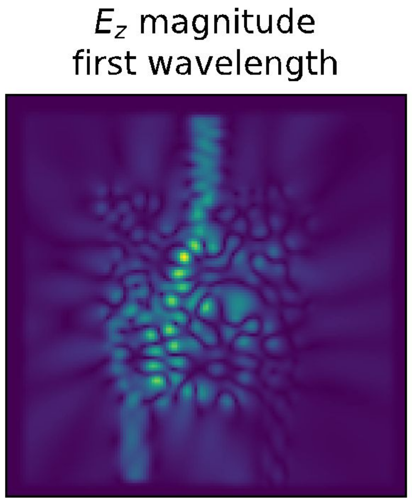
Inverse photonics Convergence comparison + Solution example

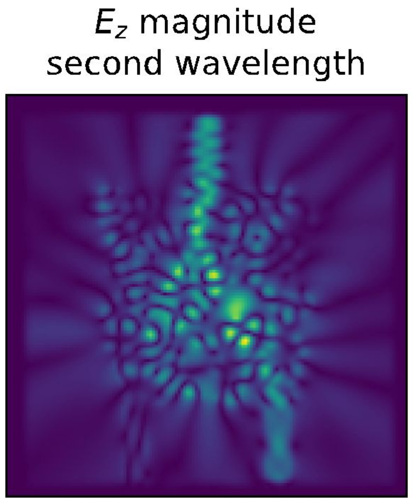


Takeaways:

- SurCo-Zero finds loss-0 solutions quickly
- SurCo-Hybrid uses offline training data to get a head start







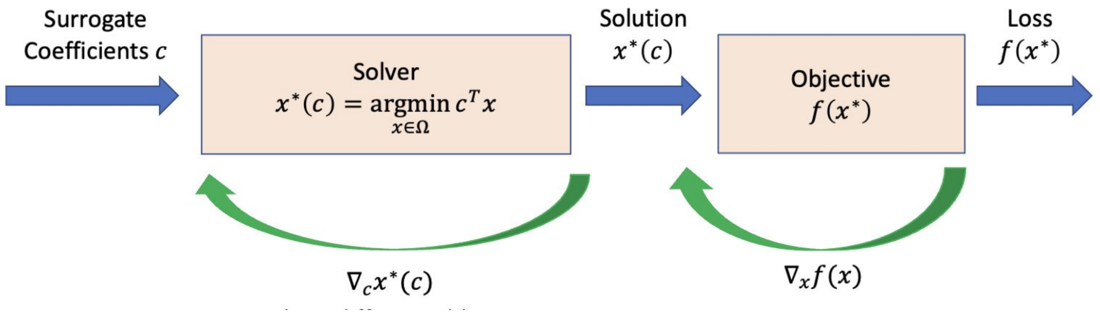
Wavelength division multiplexer

Limitation of SurCo

Recall **SurCo**: Update linear coefficients $\boldsymbol{c}$ such that $x^*(\boldsymbol{c})$ improves objective $f(x^*(\boldsymbol{c}))$

$$\min_{\boldsymbol{\theta}} \mathcal{L}(Y, Z) := \sum_{i=1}^{N} f\left(\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{y}_i); \mathbf{z}_i\right)$$

$$\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{y}) = \arg\min_{\mathbf{x}\in\Omega} \mathbf{x}^\top \mathbf{c}_{\boldsymbol{\theta}}(\mathbf{y})$$



- Requires $\nabla_x f(x)$ → Does not apply to nondifferentiable functions
- Requires $\nabla_c \boldsymbol{g_\theta}(c)$ → Solver is backpropagatable

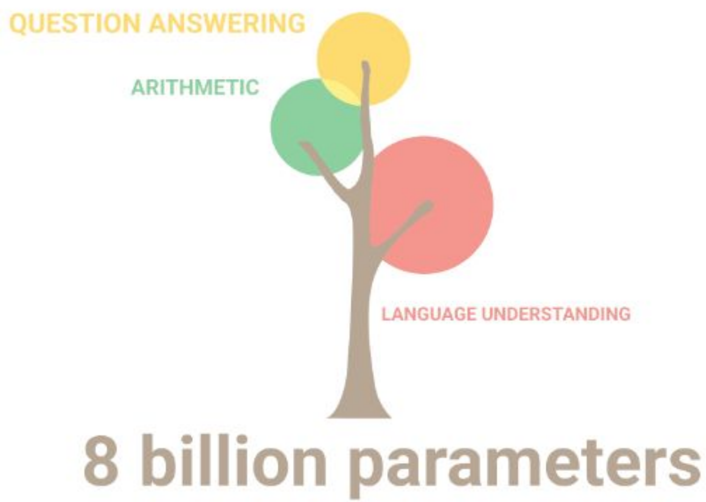
[A. Zharmagambetov et al, *Landscape Surrogate: Learning Decision Losses for Mathematical Optimization Under Partial Information*, NeurIPS'23]
[A. Ferber et al, GenCO: Generating Diverse Solutions to Design Problems with Combinatorial Nature, ICML'24]

Option **Three**: Does Deep Model Actually Converge to Anything Symbolic?

Emerging Symbolic Structure

Deep Models

Debate: Is LLM doing retrieval or true reasoning?



LLM shows emergent behaviors!!

https://medium.com/@fenjiro/large-language-models-llms-emergent-abilities-chatgpt-talks-moroccan-dialect-as-an-example-c945f93aa63a

Debate: Is LLM doing retrieval or true reasoning?

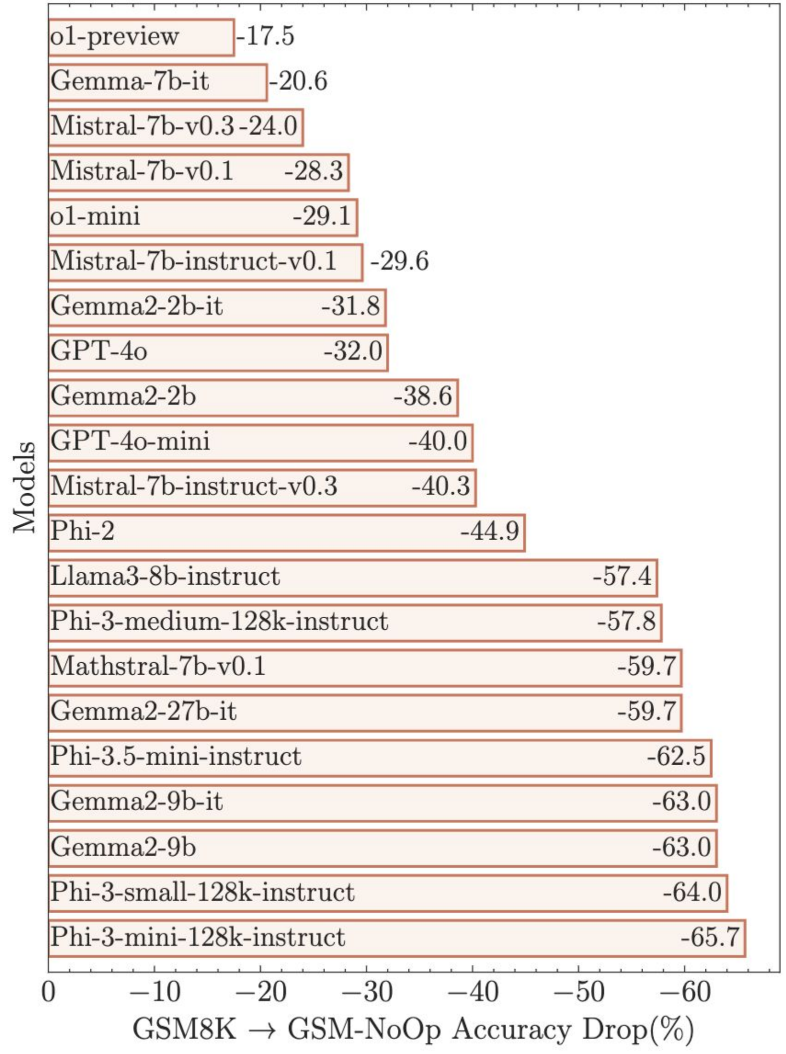
Yann LeCun @ylecun

Do LLMs perform reasoning or approximate retrieval?
There is a continuum between the two, and Auto-Regressive LLMs are largely on the retrieval side.

Subbarao Kambhampati (కంభంపాటి సుబ్బారావు) @rao2z

Emergent Abilities (noun): The preferred euphemism for what your LLM does, when saying "approximate retrieval" sounds too unsexy.

#AIAphorisms



LLM is just doing retrievals!!

Concrete Example: **Modular Addition**

$$a + b = c \mod d$$

Does neural network have an *implicit table* to do retrieval?

[T. Zhou et al, Pre-trained Large Language Models Use Fourier Features to Compute Addition]

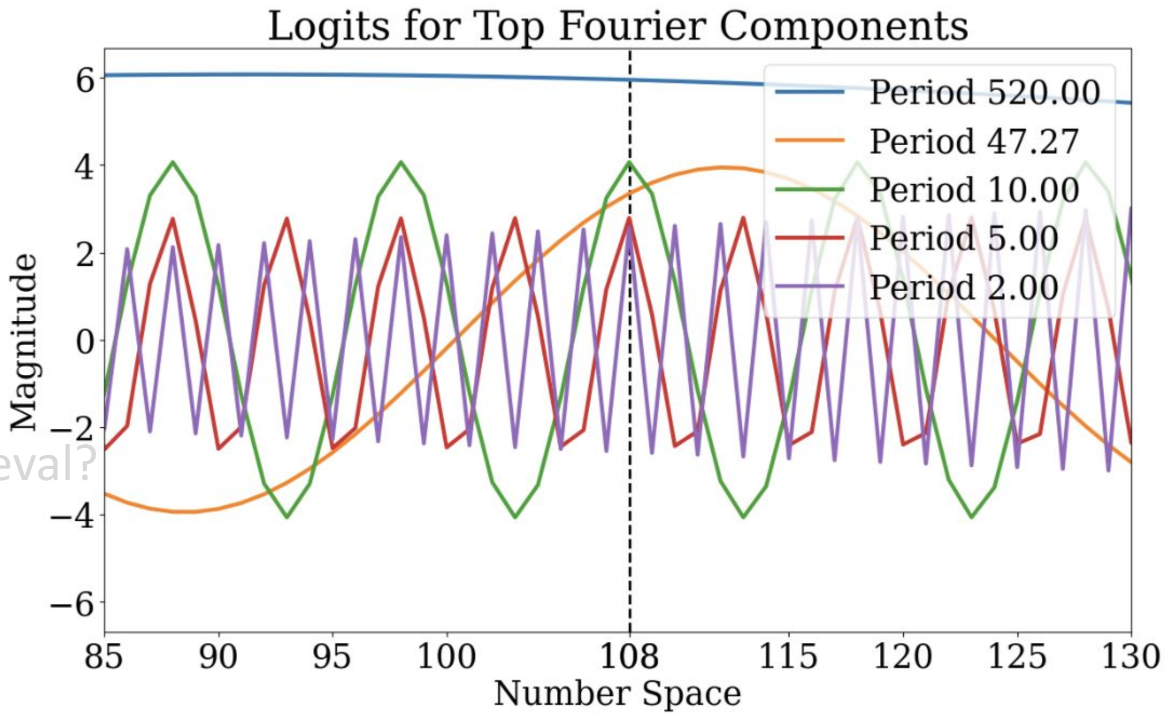
Concrete Example: **Modular Addition**

$$a + b = c \bmod d$$

Does neural network have an *implicit table* to do retrieval?

Learned representation = Fourier basis 🤯

Why? 🤔

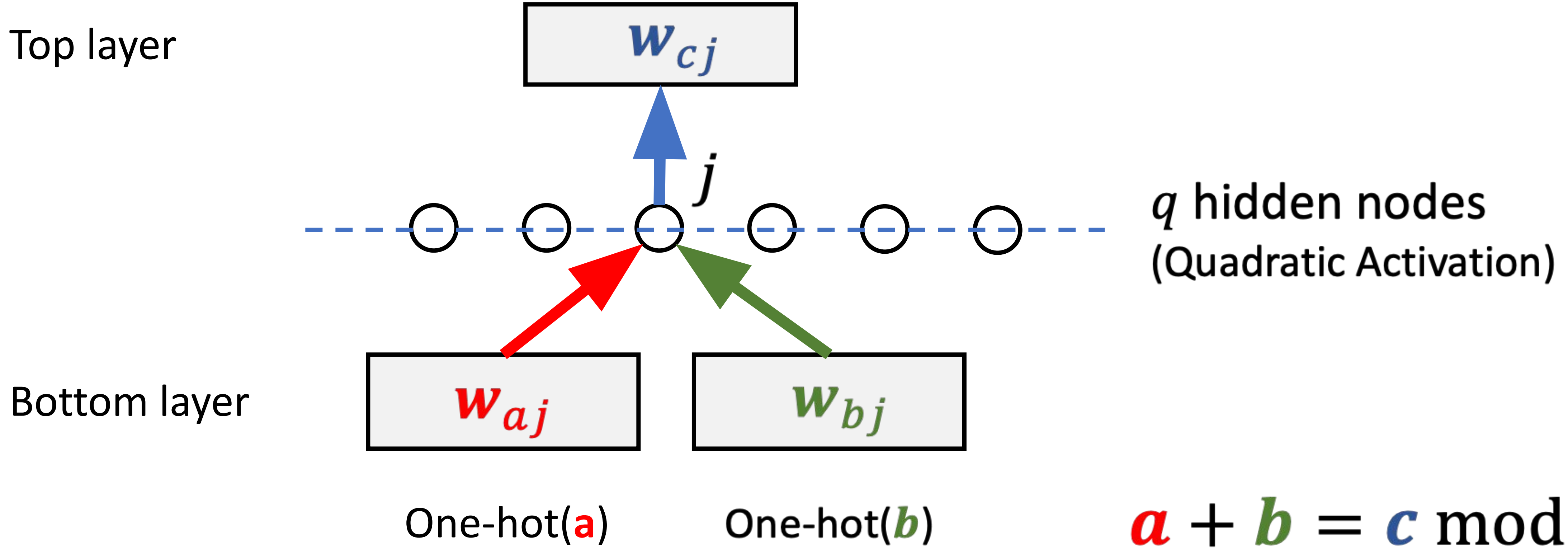


(a) Final logits for top Fourier components

[T. Zhou et al, Pre-trained Large Language Models Use Fourier Features to Compute Addition]

Problem Setup

MSE Loss: $Min\ \|\text{Output} - \text{one-hot}(c)\|_2$



$$a + b = c \bmod d$$

[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

(Scaled) Fourier Transform

$$z_{akj} = \sum_{m=0}^{d-1} w_{amj} e^{imk/d}$$

$$z_{bkj} = \sum_{m=0}^{d-1} w_{bmj} e^{imk/d}$$

$$z_{ckj} = \sum_{m=0}^{d-1} w_{cmj} e^{imk/d}$$

k: frequency

$\{W_a, W_b, W_c\}$ are real

↓

Hermitian condition holds

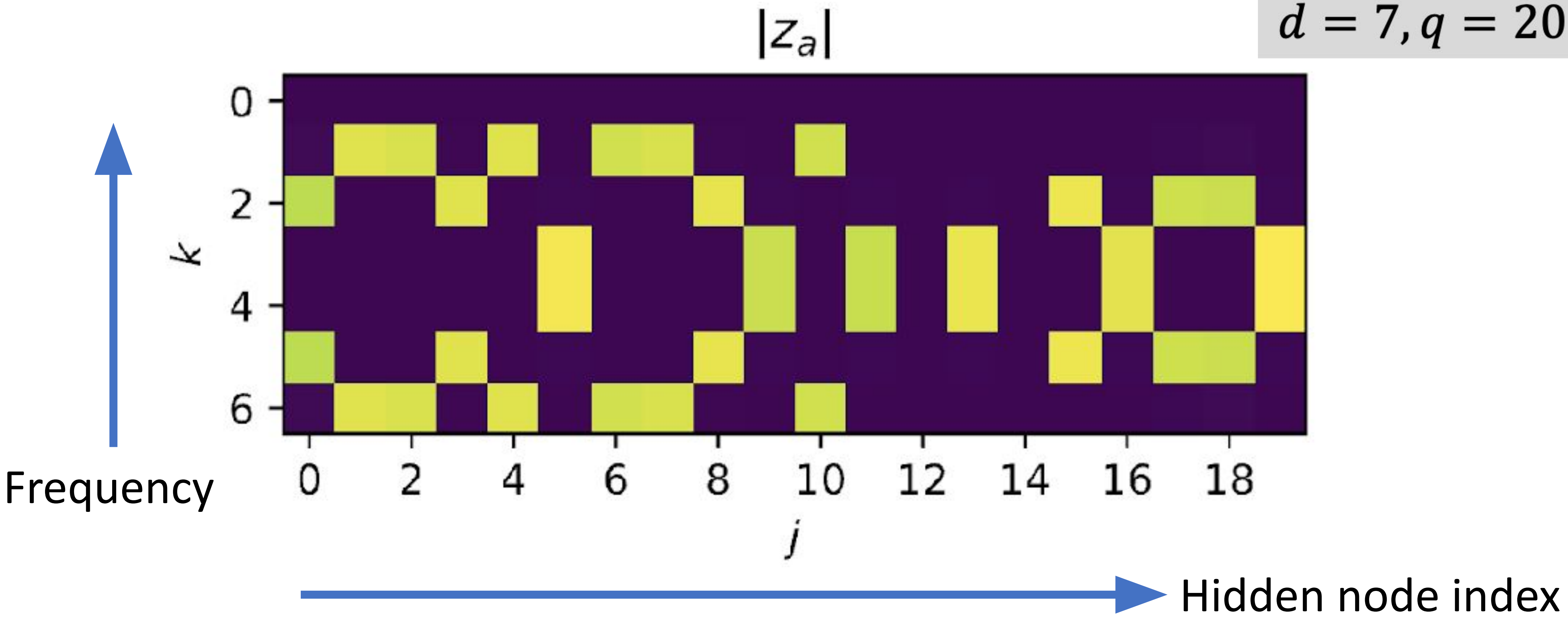
$$z_{akj} = \overline{z_{a,-k,j}}$$

$$z_{bkj} = \overline{z_{b,-k,j}}$$

$$z_{ckj} = \overline{z_{c,-k,j}}$$

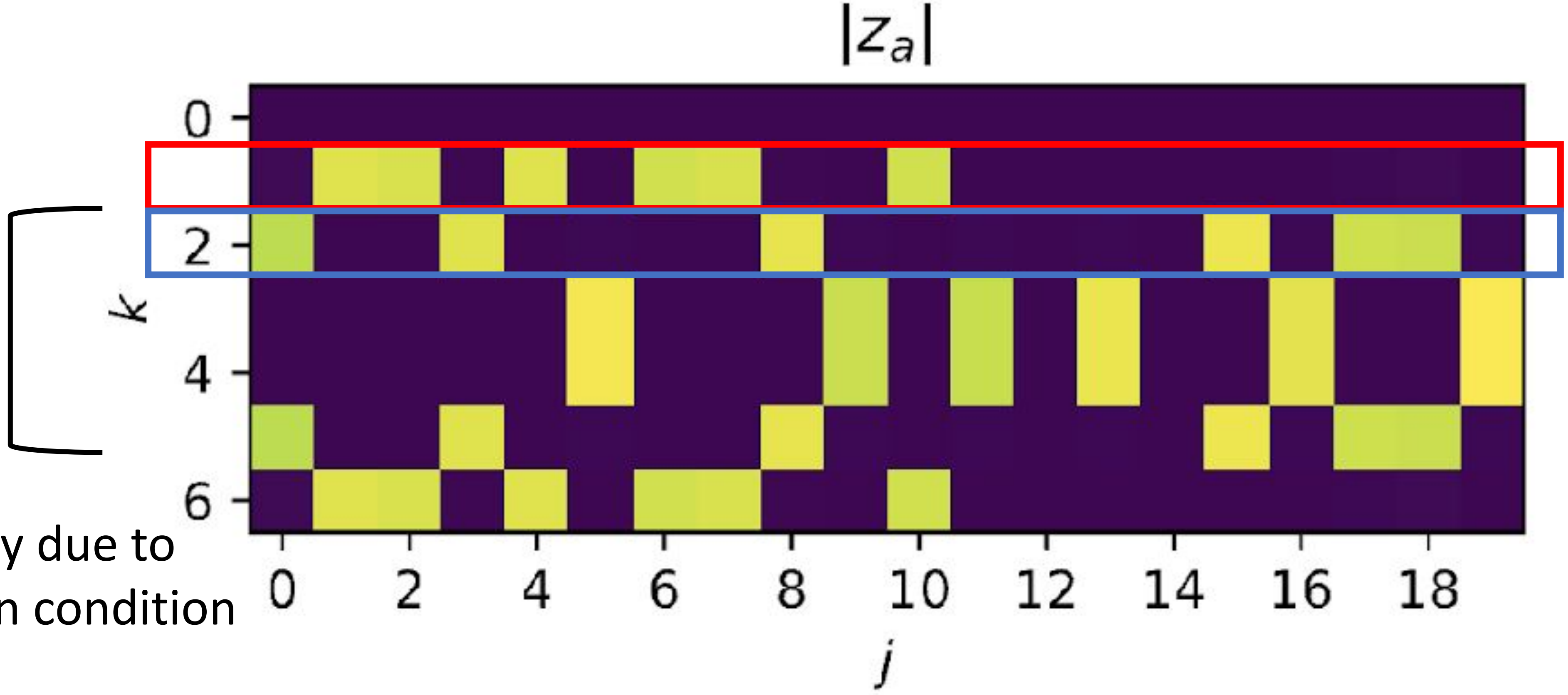
What a Gradient Descent Solution look like?

$d = 7, q = 20$



[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

What a Gradient Descent Solution look like?

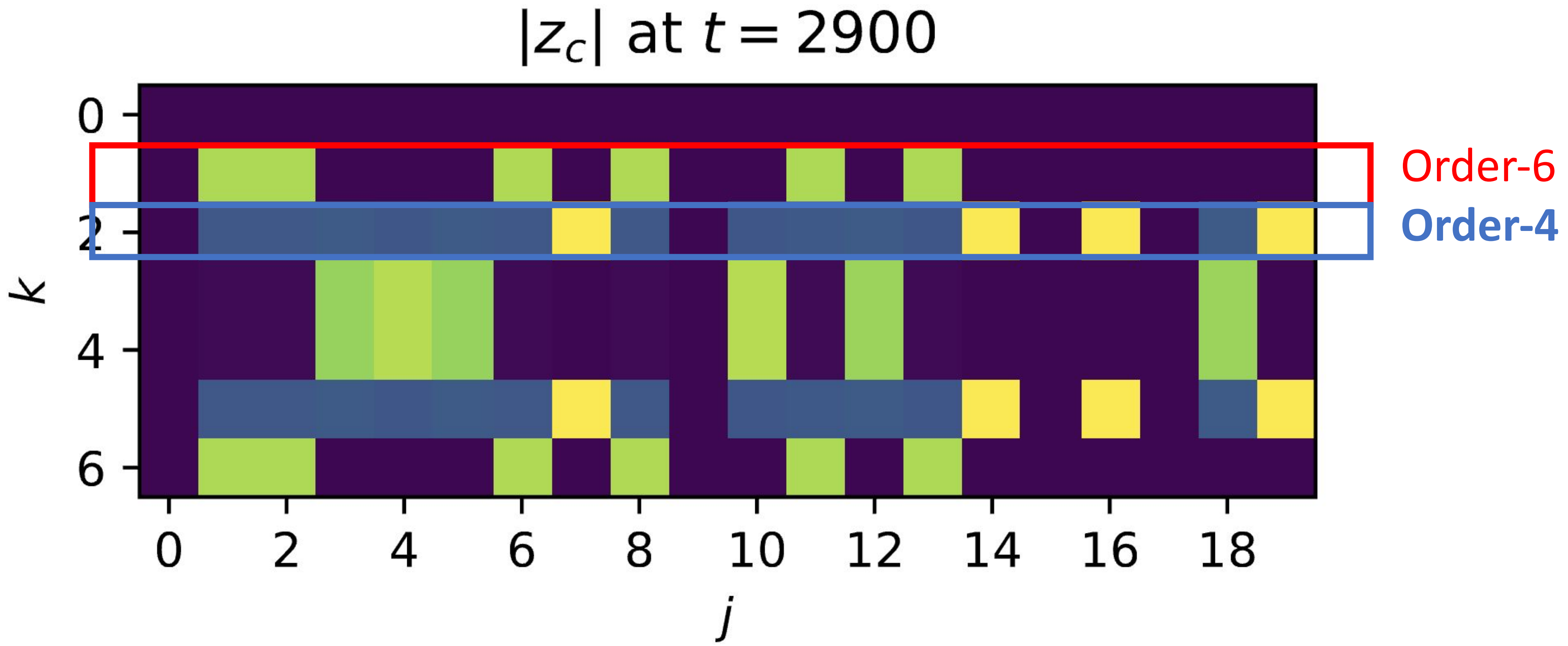


Order-6 solutions

Symmetry due to Hermitian condition

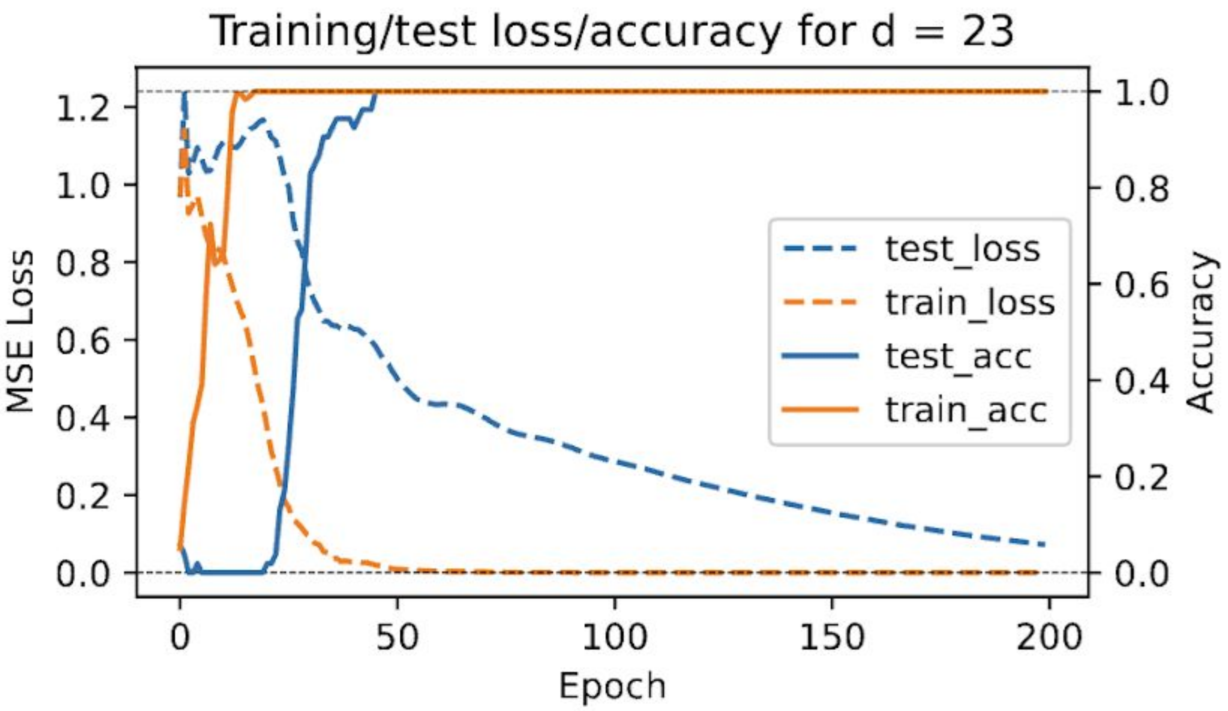
[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv’24]

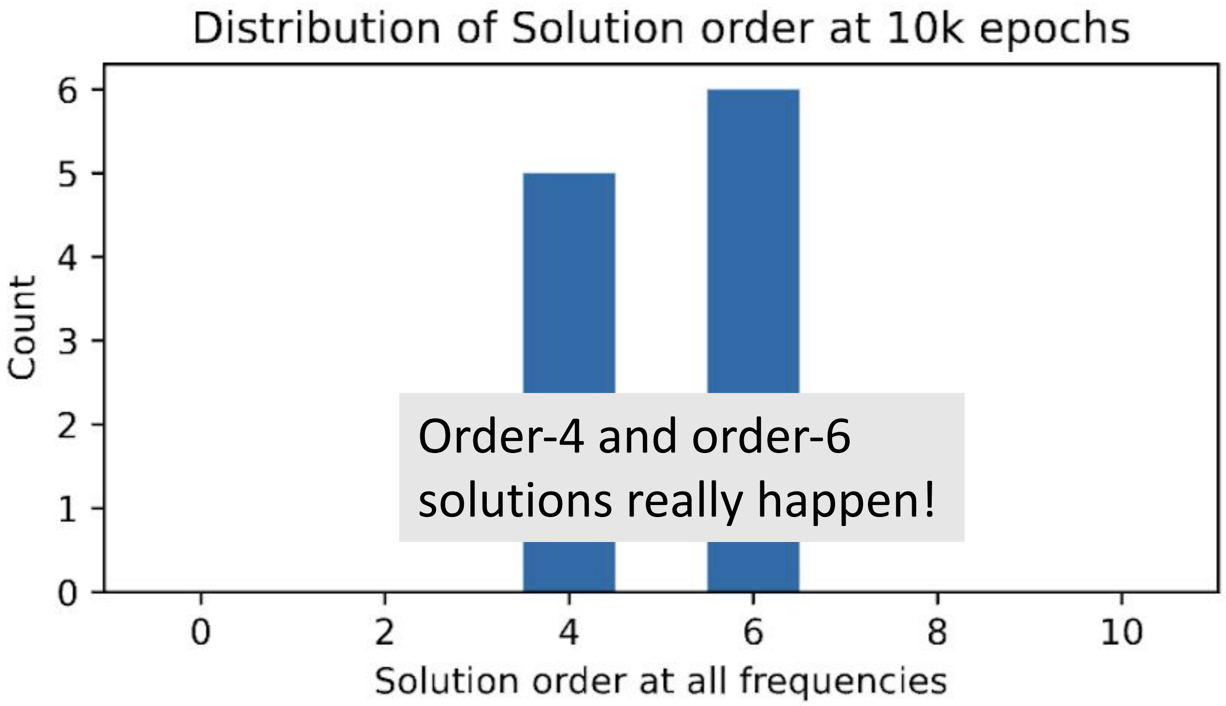
What a Gradient Descent Solution look like?



[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

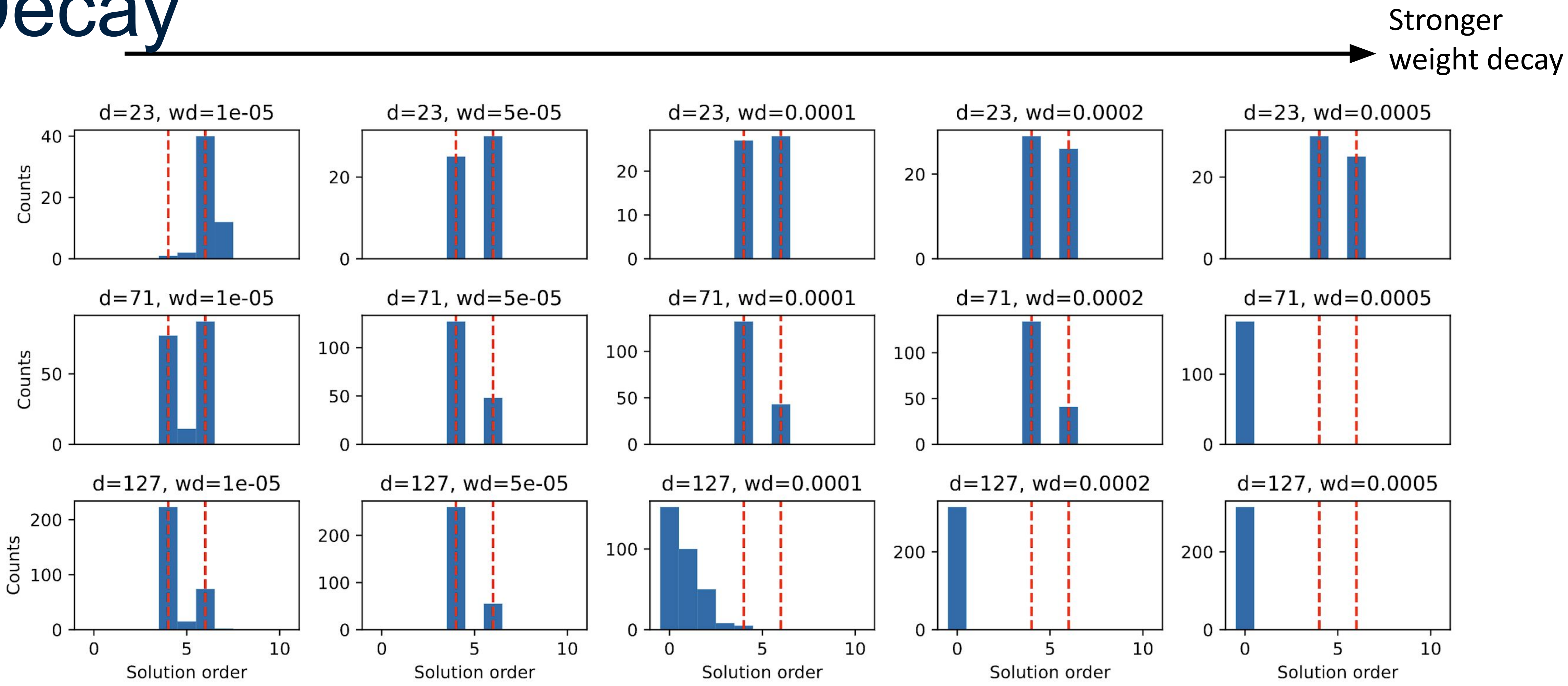
More Statistics on Gradient Descent Solutions





[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

Effect of Weight Decay



[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

Why?

🤔

Structure of Loss Functions

MSE loss $\ell(\boldsymbol{z}) = d^{-1}\sum_{k\neq 0}\ell_k(\boldsymbol{z}) + 1 - 1/d$

$$\ell_k(\boldsymbol{z}) = -2r_{kkk} + \sum_{k_1 k_2}\left|r_{k_1 k_2 k}\right|^2 + \frac{1}{4}\left|\sum_{p\in\{a,b\}}\sum_{k'} r_{p,k',-k',k}\right|^2 + \frac{1}{4}\sum_{m\neq 0}\sum_{p\in\{a,b\}}\left|\sum_{k'} r_{p,k',m-k',k}\right|^2$$

Term $r_{k_1 k_2 k}(\boldsymbol{z}) := \sum_j z_{ak_1 j} z_{bk_2 j} z_{ckj}$ and $r_{pk_1 k_2 k}(\boldsymbol{z}) := \sum_j z_{pk_1 j} z_{pk_2 j} z_{ckj}$

[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

Structure of Loss Functions

MSE loss $\ell(\mathbf{z}) = d^{-1} \sum_{k \neq 0} \ell_k(\mathbf{z}) + 1 - 1/d$

$$\ell_k(\mathbf{z}) = -2 r_{kkk} + \sum_{k_1 k_2} \left| r_{k_1 k_2 k} \right|^2 + \frac{1}{4} \left| \sum_{p \in \{a,b\}} \sum_{k'} r_{p,k',-k',k} \right|^2 + \frac{1}{4} \sum_{m \neq 0} \sum_{p \in \{a,b\}} \left| \sum_{k'} r_{p,k',m-k',k} \right|^2$$

Term $r_{k_1 k_2 k}(\mathbf{z}) := \sum_j z_{a k_1 j} z_{b k_2 j} z_{c k j}$ and $r_{p k_1 k_2 k}(\mathbf{z}) := \sum_j z_{p k_1 j} z_{p k_2 j} z_{c k j}$

Sufficient conditions of Global Optimizers:

[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

How to Optimize?

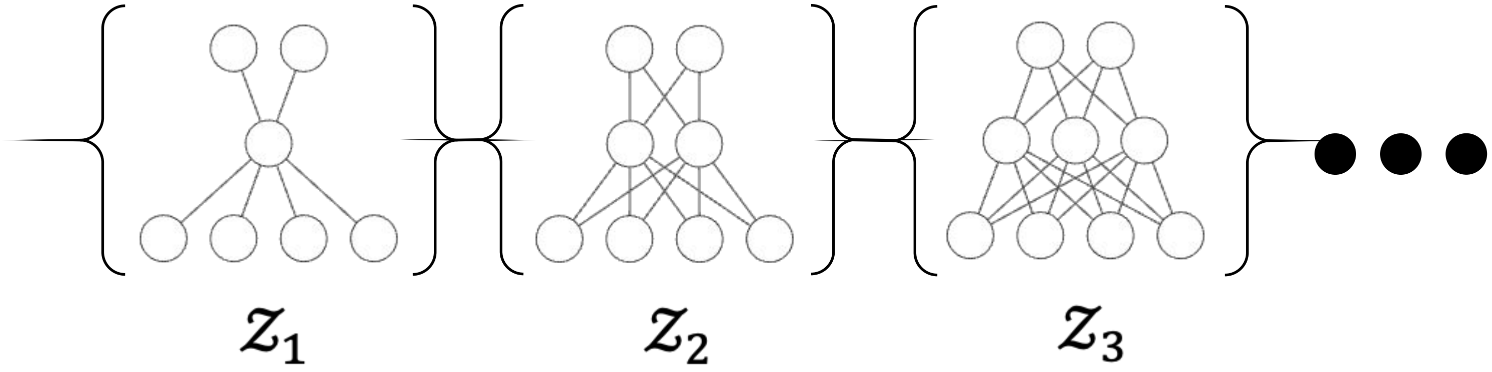
The objective is highly nonlinear !!

However, nice *algebraic structures* exist!

How to Optimize?

The objective is highly nonlinear !!

However, nice *algebraic structures* exist!

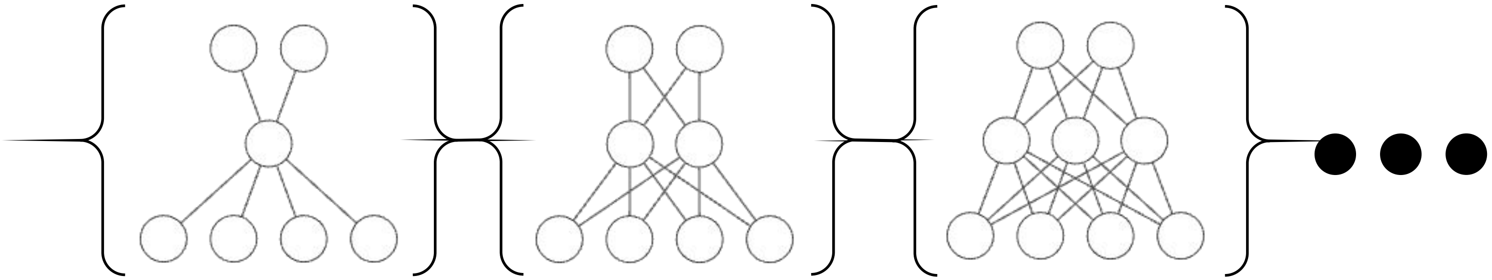


$\mathcal{Z} = \cup_{q \geq 0} \mathcal{Z}_q$: All 2-layer networks with different number of hidden nodes

How to Optimize?

The objective is highly nonlinear !!

However, nice *algebraic structures* exist!



$\mathcal{Z}_1$ $\mathcal{Z}_2$ $\mathcal{Z}_3$

$\mathcal{Z} = \cup_{q \geq 0} \mathcal{Z}_q$: All 2-layer networks with different number of hidden nodes

Ring addition $+$: Concatenate hidden nodes

Ring multiplication $*$: Kronecker production along the hidden dimensions

$\langle \mathcal{Z}, +, * \rangle$ is a ***semi-ring***

Ring Homomorphism

A function $r(\boldsymbol{z}): \mathcal{Z} \mapsto \mathbb{C}$ is a *ring homomorphism*, if

- $r(\mathbf{1}) = 1$
- $r(\boldsymbol{z}_1 + \boldsymbol{z}_2) = r(\boldsymbol{z}_1) + r(\boldsymbol{z}_2)$
- $r(\boldsymbol{z}_1 * \boldsymbol{z}_2) = r(\boldsymbol{z}_1) r(\boldsymbol{z}_2)$

Ring Homomorphism

A function $r(\mathbf{z}): \mathcal{Z} \mapsto \mathbb{C}$ is a *ring homomorphism*, if

- $r(\mathbf{1}) = 1$
- $r(\mathbf{z}_1 + \mathbf{z}_2) = r(\mathbf{z}_1) + r(\mathbf{z}_2)$
- $r(\mathbf{z}_1 * \mathbf{z}_2) = r(\mathbf{z}_1) r(\mathbf{z}_2)$

🎯 $r_{k_1 k_2 k}(\mathbf{z})$ and $r_{p k_1 k_2 k}(\mathbf{z})$ are ***ring homomorphisms***!

Ring Homomorphism

A function $r(\mathbf{z}): \mathcal{Z} \mapsto \mathbb{C}$ is a *ring homomorphism*, if

- $r(\mathbf{1}) = 1$
- $r(\mathbf{z}_1 + \mathbf{z}_2) = r(\mathbf{z}_1) + r(\mathbf{z}_2)$
- $r(\mathbf{z}_1 * \mathbf{z}_2) = r(\mathbf{z}_1) r(\mathbf{z}_2)$

🎯 $r_{k_1 k_2 k}(\mathbf{z})$ and $r_{p k_1 k_2 k}(\mathbf{z})$ are ***ring homomorphisms***!

MSE Loss

$$\ell_k(\mathbf{z}) = -2 r_{kkk} + \sum_{k_1 k_2} \left| r_{k_1 k_2 k} \right|^2 + \frac{1}{4} \left| \sum_{p \in \{a,b\}} \sum_{k'} r_{p,k',-k',k} \right|^2 + \frac{1}{4} \sum_{m \neq 0} \sum_{p \in \{a,b\}} \left| \sum_{k'} r_{p,k',m-k',k} \right|^2$$

Ring Homomorphism

A function $r(\mathbf{z}): \mathcal{Z} \mapsto \mathbb{C}$ is a *ring homomorphism*, if

- $r(\mathbf{1}) = 1$
- $r(\mathbf{z}_1 + \mathbf{z}_2) = r(\mathbf{z}_1) + r(\mathbf{z}_2)$
- $r(\mathbf{z}_1 * \mathbf{z}_2) = r(\mathbf{z}_1) r(\mathbf{z}_2)$

$r_{k_1 k_2 k}(\mathbf{z})$ and $r_{p k_1 k_2 k}(\mathbf{z})$ are ***ring homomorphisms***!

MSE Loss

$$\ell_k(\mathbf{z}) = -2 r_{kkk} + \sum_{k_1 k_2} \left| r_{k_1 k_2 k} \right|^2 + \frac{1}{4} \left| \sum_{p \in \{a,b\}} \sum_{k'} r_{p,k',-k',k} \right|^2 + \frac{1}{4} \sum_{m \neq 0} \sum_{p \in \{a,b\}} \left| \sum_{k'} r_{p,k',m-k',k} \right|^2$$

Partial solution $\mathbf{z}_1$ satisfies $r_{k_1 k_2 k}(\mathbf{z}_1) = 0$

Partial solution $\mathbf{z}_2$ satisfies $r_{pk',-k',k}(\mathbf{z}_2) = 0$

Ring Homomorphism

A function $r(\mathbf{z}): \mathcal{Z} \mapsto \mathbb{C}$ is a *ring homomorphism*, if

- $r(\mathbf{1}) = 1$
- $r(\mathbf{z}_1 + \mathbf{z}_2) = r(\mathbf{z}_1) + r(\mathbf{z}_2)$
- $r(\mathbf{z}_1 * \mathbf{z}_2) = r(\mathbf{z}_1) r(\mathbf{z}_2)$

$r_{k_1 k_2 k}(\mathbf{z})$ and $r_{p k_1 k_2 k}(\mathbf{z})$ are ***ring homomorphisms***!

MSE Loss

$$\ell_k(\mathbf{z}) = -2 r_{kkk} + \sum_{k_1 k_2} \left| r_{k_1 k_2 k} \right|^2 + \frac{1}{4} \left| \sum_{p \in \{a,b\}} \sum_{k'} r_{p,k',-k',k} \right|^2 + \frac{1}{4} \sum_{m \neq 0} \sum_{p \in \{a,b\}} \left| \sum_{k'} r_{p,k',m-k',k} \right|^2$$

Partial solution $\mathbf{z}_1$ satisfies $r_{k_1 k_2 k}(\mathbf{z}_1) = 0$

Partial solution $\mathbf{z}_2$ satisfies $r_{pk',-k',k}(\mathbf{z}_2) = 0$

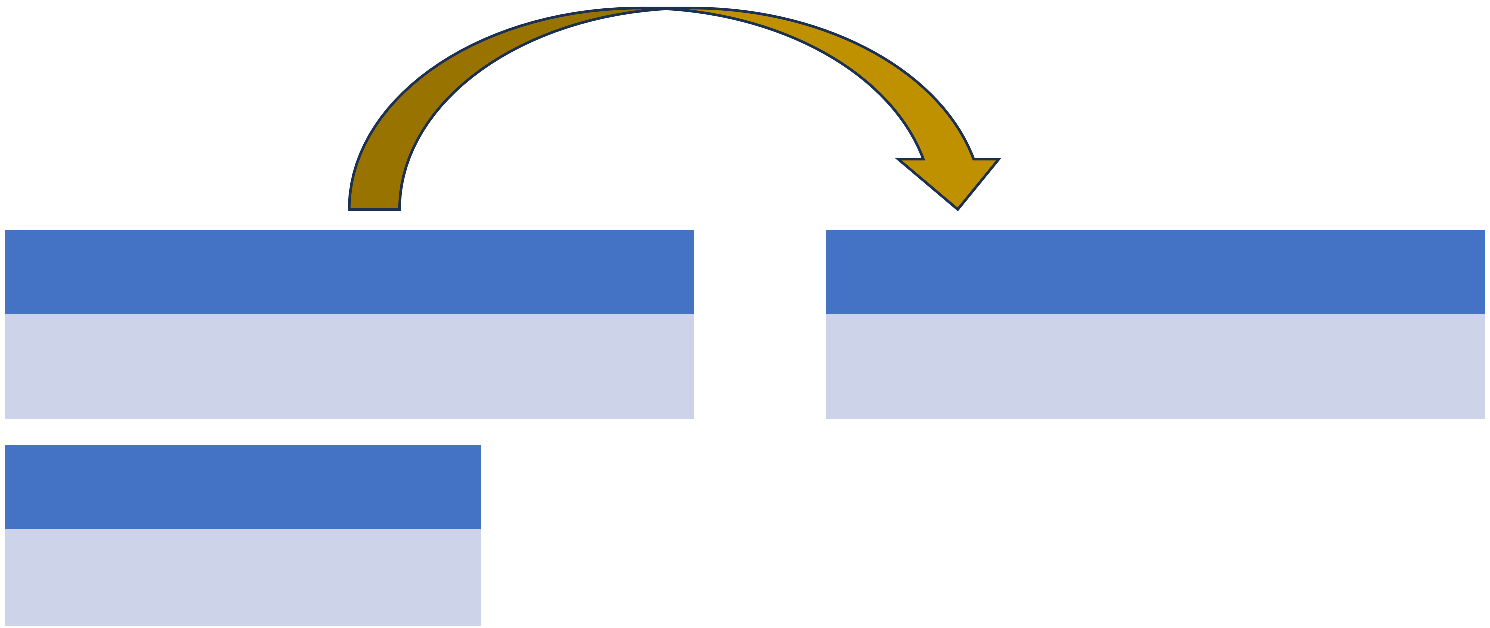
$\mathbf{z} = \mathbf{z}_1 * \mathbf{z}_2$ satisfies both $r_{k_1 k_2 k}(\mathbf{z}) = r_{pk',-k',k}(\mathbf{z}) = 0$

Composing Global Optimizers from Partial Ones

[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

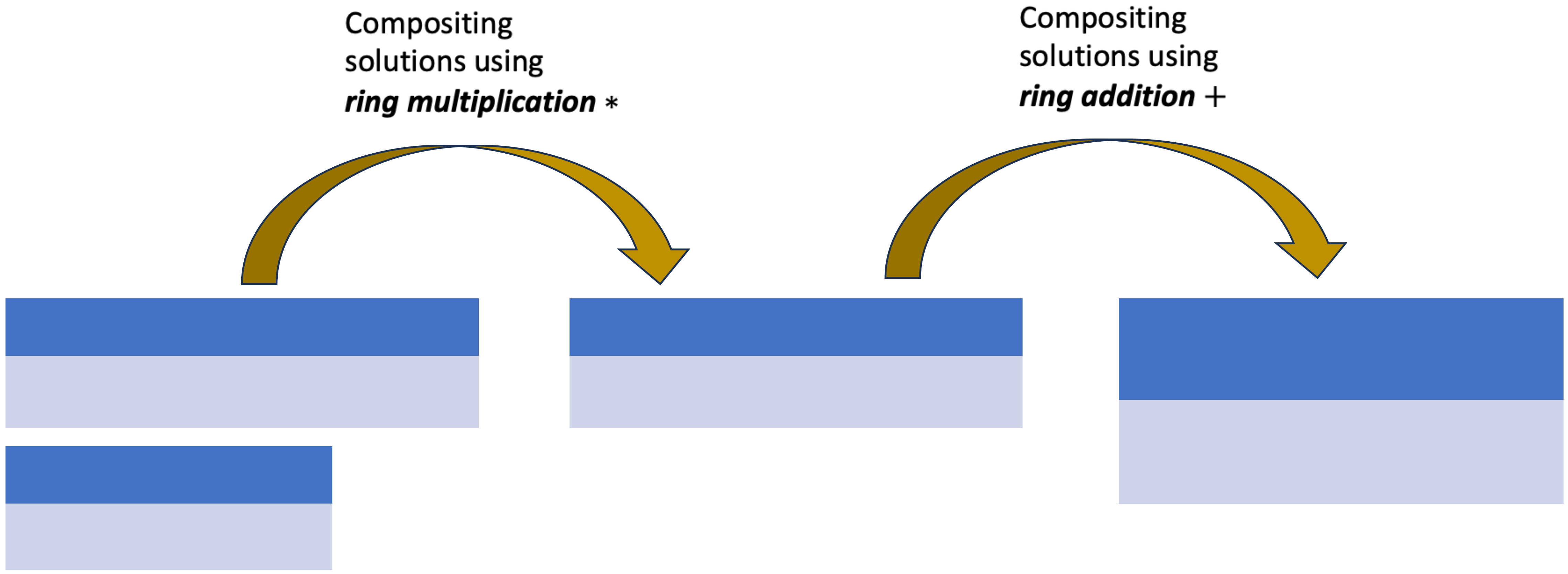
Composing Global Optimizers from Partial Ones

Compositing
solutions using
ring multiplication $*$



[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

Composing Global Optimizers from Partial Ones



[Y. Tian, *Composing Global Optimizers to Reasoning Tasks via Algebraic Objects in Neural Nets*, arXiv'24]

Exemplar constructed global optimizers

Order-6 $\boldsymbol{z}_{F6}$ (2*3)

$$\boldsymbol{z}_{F6} = \frac{1}{\sqrt[3]{6}} \sum_{k=1}^{(d-1)/2} \boldsymbol{z}_{\text{syn}}^{(k)} * \boldsymbol{z}_{\nu}^{(k)} * \boldsymbol{y}_k$$

Exemplar constructed global optimizers

Order-6 $\boldsymbol{z}_{F6}$ (2*3)

$$\boldsymbol{z}_{F6} = \frac{1}{\sqrt[3]{6}} \sum_{k=1}^{(d-1)/2} \boldsymbol{z}_{\mathrm{syn}}^{(k)} * \boldsymbol{z}_{\nu}^{(k)} * \boldsymbol{y}_k$$

Order-4 $\boldsymbol{z}_{F4/6}$ (2*2)
(mixed with order-6)

$$\boldsymbol{z}_{F4/6} = \frac{1}{\sqrt[3]{6}} \hat{\boldsymbol{z}}_{F6}^{(k_0)} + \frac{1}{\sqrt[3]{4}} \sum_{k=1, k\neq k_0}^{(d-1)/2} \boldsymbol{z}_{F4}^{(k)}$$

Exemplar constructed global optimizers

Order-6 $\boldsymbol{z}_{F6}$ (2*3)

$$\boldsymbol{z}_{F6} = \frac{1}{\sqrt[3]{6}} \sum_{k=1}^{(d-1)/2} \boldsymbol{z}_{\text{syn}}^{(k)} * \boldsymbol{z}_{\nu}^{(k)} * \boldsymbol{y}_k$$

Order-4 $\boldsymbol{z}_{F4/6}$ (2*2) (mixed with order-6)

$$\boldsymbol{z}_{F4/6} = \frac{1}{\sqrt[3]{6}} \hat{\boldsymbol{z}}_{F6}^{(k_0)} + \frac{1}{\sqrt[3]{4}} \sum_{k=1, k\neq k_0}^{(d-1)/2} \boldsymbol{z}_{F4}^{(k)}$$

Perfect memorization (order-d per frequency)

$$\boldsymbol{z}_a = \sum_{j=0}^{d-1} \boldsymbol{u}_a^j, \qquad \boldsymbol{z}_b = \sum_{j=0}^{d-1} \boldsymbol{u}_b^j$$

$$\boldsymbol{z}_M = d^{-2/3} \boldsymbol{z}_a * \boldsymbol{z}_b$$

Gradient Descent solutions matches with construction

d	%not	%non-factorable		error ($\times 10^{-2}$)		solution distribution (%) in factorable ones			
	order-4/6	order-4	order-6	order-4	order-6	$z^{(k)}_{\nu=\mathrm{i}} * z^{(k)}_{\xi}$	$z^{(k)}_{\nu=\mathrm{i}} * z^{(k)}_{\mathrm{syn},\alpha\beta}$	$z^{(k)}_{\nu} * z^{(k)}_{\mathrm{syn}}$	others
23	0.0 ± 0.0	0.00 ± 0.00	5.71 ± 5.71	0.05 ± 0.01	4.80 ± 0.96	47.07 ± 1.88	11.31 ± 1.76	39.80 ± 2.11	1.82 ± 1.82
71	0.0 ± 0.0	0.00 ± 0.00	0.00 ± 0.00	0.03 ± 0.00	5.02 ± 0.25	72.57 ± 0.70	4.00 ± 1.14	21.14 ± 2.14	2.29 ± 1.07
127	0.0 ± 0.0	1.50 ± 0.92	0.00 ± 0.00	0.26 ± 0.14	0.93 ± 0.18	82.96 ± 0.39	2.25 ± 0.64	14.13 ± 0.87	0.66 ± 0.66

$q = 512, wd = 5 \cdot 10^{-5}$

Gradient Descent solutions matches with construction

d	%not	%non-factorable		error ($\times 10^{-2}$)		solution distribution (%) in factorable ones			
	order-4/6	order-4	order-6	order-4	order-6	$\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\xi}$	$\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\mathrm{syn},\alpha\beta}$	$\boldsymbol{z}^{(k)}_{\nu} * \boldsymbol{z}^{(k)}_{\mathrm{syn}}$	others
23	$0.0_{\pm 0.0}$	$0.00_{\pm 0.00}$	$5.71_{\pm 5.71}$	$0.05_{\pm 0.01}$	$4.80_{\pm 0.96}$	$47.07_{\pm 1.88}$	$11.31_{\pm 1.76}$	$39.80_{\pm 2.11}$	$1.82_{\pm 1.82}$
71	$0.0_{\pm 0.0}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.03_{\pm 0.00}$	$5.02_{\pm 0.25}$	$72.57_{\pm 0.70}$	$4.00_{\pm 1.14}$	$21.14_{\pm 2.14}$	$2.29_{\pm 1.07}$
127	$0.0_{\pm 0.0}$	$1.50_{\pm 0.92}$	$0.00_{\pm 0.00}$	$0.26_{\pm 0.14}$	$0.93_{\pm 0.18}$	$82.96_{\pm 0.39}$	$2.25_{\pm 0.64}$	$14.13_{\pm 0.87}$	$0.66_{\pm 0.66}$

100% of the per-freq solutions are order-4/6

Gradient Descent solutions matches with construction

d	%not	%non-factorable		error ($\times 10^{-2}$)		solution distribution (%) in factorable ones			
	order-4/6	order-4	order-6	order-4	order-6	$\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\xi}$	$\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\mathrm{syn},\alpha\beta}$	$\boldsymbol{z}^{(k)}_{\nu} * \boldsymbol{z}^{(k)}_{\mathrm{syn}}$	others
23	0.0 ± 0.0	0.00 ± 0.00	5.71 ± 5.71	0.05 ± 0.01	4.80 ± 0.96	47.07 ± 1.88	11.31 ± 1.76	39.80 ± 2.11	1.82 ± 1.82
71	0.0 ± 0.0	0.00 ± 0.00	0.00 ± 0.00	0.03 ± 0.00	5.02 ± 0.25	72.57 ± 0.70	4.00 ± 1.14	21.14 ± 2.14	2.29 ± 1.07
127	0.0 ± 0.0	1.50 ± 0.92	0.00 ± 0.00	0.26 ± 0.14	0.93 ± 0.18	82.96 ± 0.39	2.25 ± 0.64	14.13 ± 0.87	0.66 ± 0.66

95% of the solutions are factorizable into “2*3” or “2*2”

Gradient Descent solutions matches with construction

d	%not	%non-factorable		error ($\times 10^{-2}$)		solution distribution (%) in factorable ones			
	order-4/6	order-4	order-6	order-4	order-6	$\boldsymbol{z}_{\nu=\mathrm{i}}^{(k)} * \boldsymbol{z}_{\xi}^{(k)}$	$\boldsymbol{z}_{\nu=\mathrm{i}}^{(k)} * \boldsymbol{z}_{\mathrm{syn},\alpha\beta}^{(k)}$	$\boldsymbol{z}_{\nu}^{(k)} * \boldsymbol{z}_{\mathrm{syn}}^{(k)}$	others
23	$0.0_{\pm 0.0}$	$0.00_{\pm 0.00}$	$5.71_{\pm 5.71}$	$0.05_{\pm 0.01}$	$4.80_{\pm 0.96}$	$47.07_{\pm 1.88}$	$11.31_{\pm 1.76}$	$39.80_{\pm 2.11}$	$1.82_{\pm 1.82}$
71	$0.0_{\pm 0.0}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.03_{\pm 0.00}$	$5.02_{\pm 0.25}$	$72.57_{\pm 0.70}$	$4.00_{\pm 1.14}$	$21.14_{\pm 2.14}$	$2.29_{\pm 1.07}$
127	$0.0_{\pm 0.0}$	$1.50_{\pm 0.92}$	$0.00_{\pm 0.00}$	$0.26_{\pm 0.14}$	$0.93_{\pm 0.18}$	$82.96_{\pm 0.39}$	$2.25_{\pm 0.64}$	$14.13_{\pm 0.87}$	$0.66_{\pm 0.66}$

Factorization error is very small

Gradient Descent solutions matches with construction

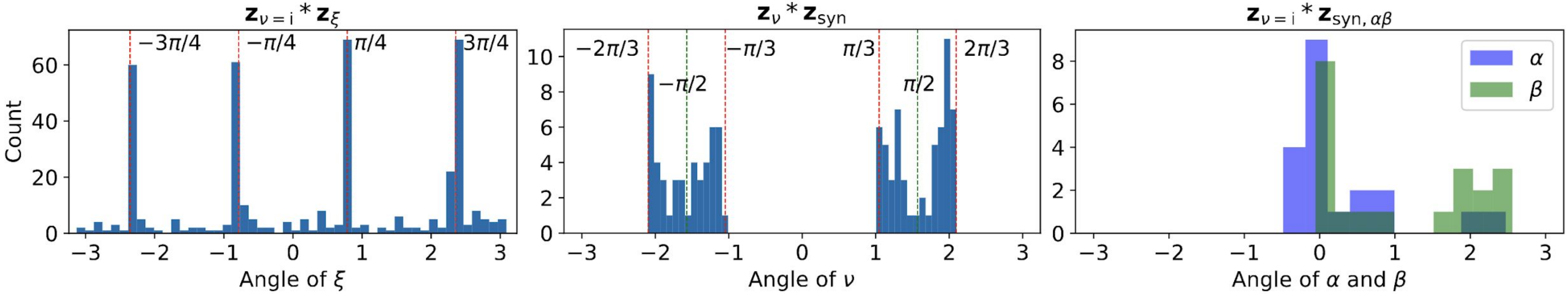
d	%not	%non-factorable		error ($\times 10^{-2}$)		solution distribution (%) in factorable ones			
	order-4/6	order-4	order-6	order-4	order-6	$\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\xi}$	$\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\mathrm{syn},\alpha\beta}$	$\boldsymbol{z}^{(k)}_{\nu} * \boldsymbol{z}^{(k)}_{\mathrm{syn}}$	others
23	0.0±0.0	0.00±0.00	5.71±5.71	0.05±0.01	4.80±0.96	47.07±1.88	11.31±1.76	39.80±2.11	1.82±1.82
71	0.0±0.0	0.00±0.00	0.00±0.00	0.03±0.00	5.02±0.25	72.57±0.70	4.00±1.14	21.14±2.14	2.29±1.07
127	0.0±0.0	1.50±0.92	0.00±0.00	0.26±0.14	0.93±0.18	82.96±0.39	2.25±0.64	14.13±0.87	0.66±0.66

98% of the solutions can be factorizable into the constructed forms

Gradient Descent solutions matches with construction

d	%not	%non-factorable		error ($\times 10^{-2}$)		solution distribution (%) in factorable ones			
	order-4/6	order-4	order-6	order-4	order-6	$z^{(k)}_{\nu=\mathrm{i}} * z^{(k)}_{\xi}$	$z^{(k)}_{\nu=\mathrm{i}} * z^{(k)}_{\mathrm{syn},\alpha\beta}$	$z^{(k)}_{\nu} * z^{(k)}_{\mathrm{syn}}$	others
23	$0.0_{\pm 0.0}$	$0.00_{\pm 0.00}$	$5.71_{\pm 5.71}$	$0.05_{\pm 0.01}$	$4.80_{\pm 0.96}$	$47.07_{\pm 1.88}$	$11.31_{\pm 1.76}$	$39.80_{\pm 2.11}$	$1.82_{\pm 1.82}$
					5	$72.57_{\pm 0.70}$	$4.00_{\pm 1.14}$	$21.14_{\pm 2.14}$	$2.29_{\pm 1.07}$
					8	$82.96_{\pm 0.39}$	$2.25_{\pm 0.64}$	$14.13_{\pm 0.87}$	$0.66_{\pm 0.66}$

Distribution of the parameters in the solutions



Possible Implications

Do neural networks end up learning more efficient symbolic representations that we don’t know?

Does gradient descent lead to a solution that can be reached by advanced algebraic operations?

Will gradient descent become obsolete, eventually?

Thanks!