CS 639: Foundation Models
More on Reasoning

Fred Sala

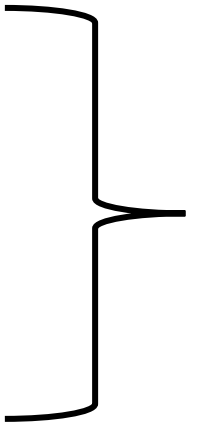
University of Wisconsin-Madison

March 24, 2026

Announcements

- **Exam:** grades are out
  - Talk to us if anything is wrong
- **Homework 3**: due!
  - Bonus OH: tomorrow (Weds) 3:30-5:00 pm
- **Project information**: here
  https://pages.cs.wisc.edu/~fredsala/cs639/files/project_info_639.pdf
- Class outline:

Tuesday March 24	Reasoning I: More CoT
Thursday March 26	Reasoning II: RLVR



Outline

- **Review: Alignment and RLHF**
  - Basic alignment idea, goals, mechanisms, RL review, RLHF steps
- **Reasoning: Back to Chain-of-Thought**
  - Reasoning intro, chain-of-thought for reasoning, generalizations, CoT challenges and functionality
- **RL for Reasoning**
  - Notion of “verifiers”, using RL for reasoning rather than alignment

Outline

- **Review: Alignment and RLHF**
  - Basic alignment idea, goals, mechanisms, RL review, RLHF steps
- **Reasoning: Back to Chain-of-Thought**
  - Reasoning intro, chain-of-thought for reasoning, generalizations, CoT challenges and functionality
- **RL for Reasoning**
  - Notion of “verifiers”, using RL for reasoning rather than alignment

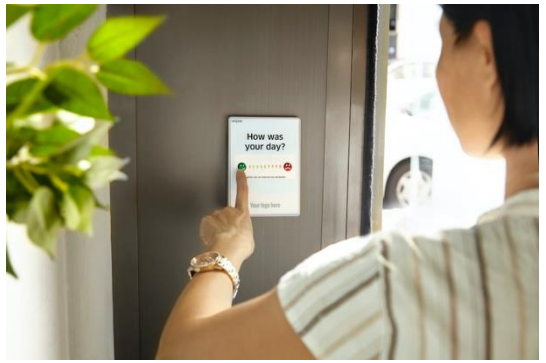
Alignment: **Basic Motivation**

Goal: produce language model outputs that **users like better…**

- **Challenge: Hard** to specify exactly what this means
- We can do fine-tuning/instruction-tuning to make model more likely to produce certain outputs
  - But we don't necessarily know that these are "preferred" outputs
  - Plus, the model may not produce desirable outputs more generally

Instead, let's use feedback.

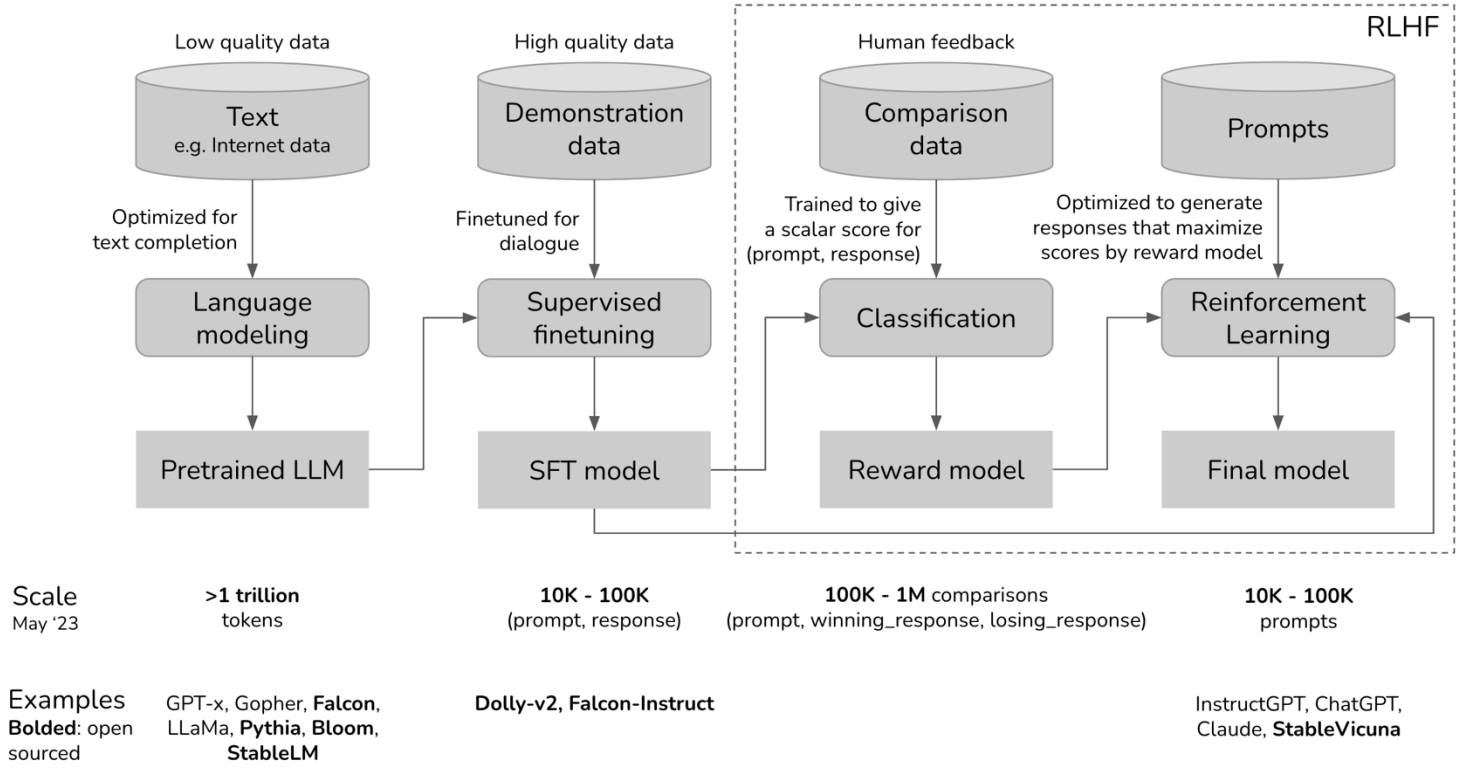
- **Why? Easy** to query users
- Not the only way to do alignment, but we'll focus on it---will lead to alignment via RLHF



RLHF: **Setup**

Goal: produce language model outputs that users like better…

- Will use **R**einforcement **L**earning from **H**uman **F**eedback
- Via RL with trained reward model (Ouyang et al ‘22)



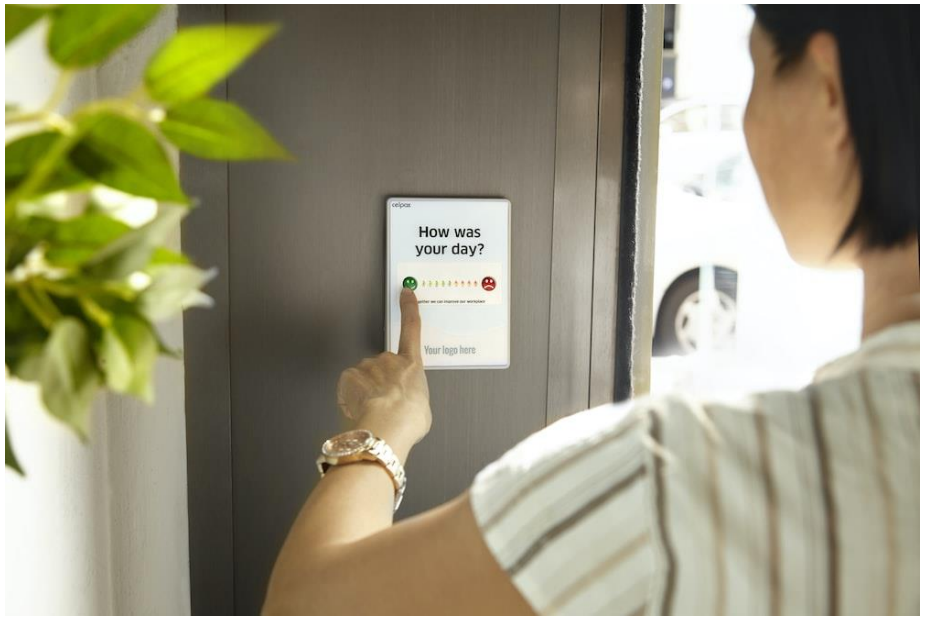
Chip Huyen

RLHF: **Feedback**

First stage: get **human feedback** to train reward model

- Fix a set of prompts
- Produce multiple outputs for each prompt
  - Can get them from the original model post-SFT, or otherwise

- Ask human users **which is better**
  - **Binary output**
  - Can do more, but don't have to
    - Rank more questions, get feedback, etc.



RLHF: **Reward/Preference Model**

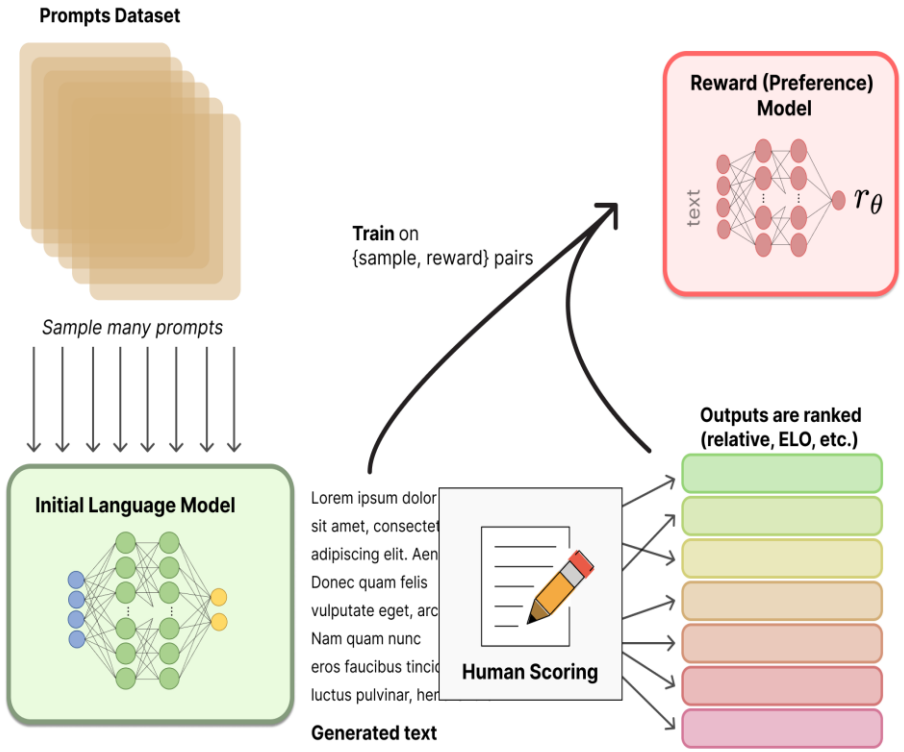
Second stage: train reward model

- Use the human feedback to train/fine-tune another model to reproduce the preferences
- **"Preference" model**

- **Why?** Reward model can tell us,

in general, how "liked" any input is.

  - Use to tell how good outputs are
- Note: **not generative!**
  - Input: token sequence
  - Output: scalar reward value



https://huggingface.co/blog/rlhf

RLHF: **Reward/Preference Model**

Second stage: train reward model

- Use the human feedback to train/fine-tune another model to reproduce the preferences
- Mismatch between training data and what model produces
  - Training data: pairs (x, y_l), (x, y_w)

Prompt and **Preferred (winner)** response

Prompt and **Dispreferred (loser)** response

  - But the reward model produces just **one** value: r(x,y).
  - So must somehow form an objective that makes r(x,y) consistent with the winning and losing pairs → softmax!

RLHF: **Reward/Preference Model**

Second stage: train reward model

- Mismatch between training data and what model produces
  - Training data: pairs (x, y_l), (x, y_w)
  - But the reward model produces just **one** value: r(x,y).
  - So must somehow form an objective that makes r(x,y) consistent with the winning and losing pairs → softmax!
  - Bradley-Terry preference model (famous in social choice theory)

$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(r^*(x, y_1)\right)}{\exp\left(r^*(x, y_1)\right) + \exp\left(r^*(x, y_2)\right)}.$$

  - Now we can relate winning and losing responses to scalar rewards
  - And can learn a reward model maximally consistent with our data

RLHF: **Reward/Preference Model**

Second stage: train reward model

- Bradley-Terry preference model (famous in social choice theory)
  - Now we can relate winning and losing responses to scalar rewards
  - And can learn a reward model maximally consistent with our data

$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(r^*(x, y_1)\right)}{\exp\left(r^*(x, y_1)\right) + \exp\left(r^*(x, y_2)\right)}.$$

  - Actual training objective: log likelihood over our dataset,

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))\right]$$

RLHF: **Reward/Preference Model**

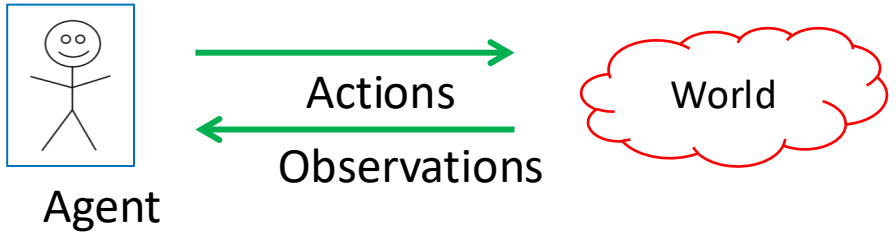
Note: we don't have to always do this from scratch

- **Pretrained** reward models available (like in our FMs in general)
- Benchmarks for this:

	Model	Model Type	Score	Chat	Chat Hard
1	nvidia/Llama-3.1-Nemotron-70B-Reward	Custom Classifier	94.1	97.5	85.7
2	Skywork/Skywork-Reward-Gemma-2-27B	Seq. Classifier	93.8	95.8	91.4
3	SF-Foundation/TextEval-Llama3.1-70B	Generative	93.5	94.1	90.1
4	meta-metrics/MetaMetrics-RM-v1.0	Custom Classifier	93.4	98.3	86.4
5	Skywork/Skywork-Critic-Llama-3.1-70B	Generative	93.3	96.6	87.9
6	LxzGordon/URM-LLaMa-3.1-8B	Seq. Classifier	92.9	95.5	88.2
7	Salesforce/SFR-LLaMa-3.1-70B-Judge-r *	Generative	92.7	96.9	84.8
8	Skywork/Skywork-Reward-Llama-3.1-8B	Seq. Classifier	92.5	95.8	87.3

Lambert et al '24

RLHF: **Fine-Tuning with RL**



Third stage: RL

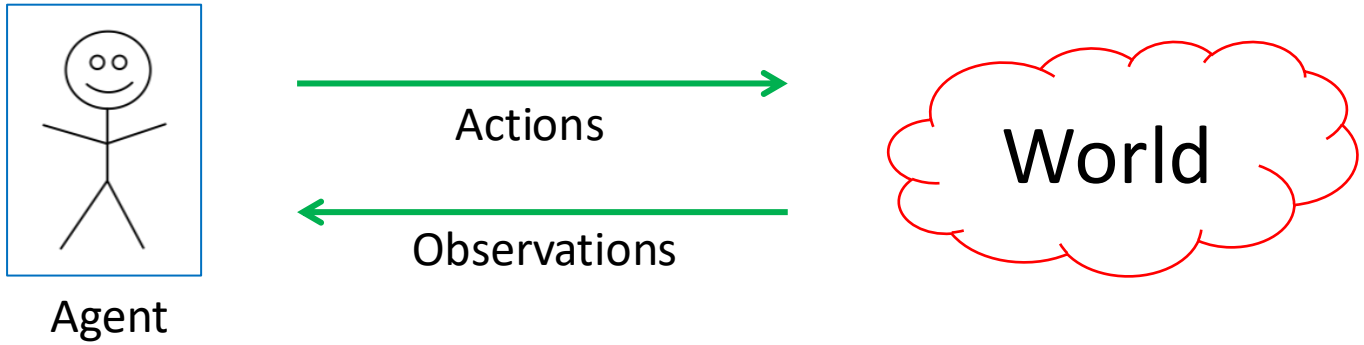
- Use an RL algorithm
- **Goal:** produce outputs that have high reward

RL formulation (see overview coming up for RL reminder!)

- **Action space**: all the tokens possible to output
- **State space**: all the sequences of tokens
- **Reward function**: the trained reward model
- **Policy**: the new version of the LM, taking in state and returning tokens

Reinforcement Learning Review

We have an **agent** **interacting** with the **world**

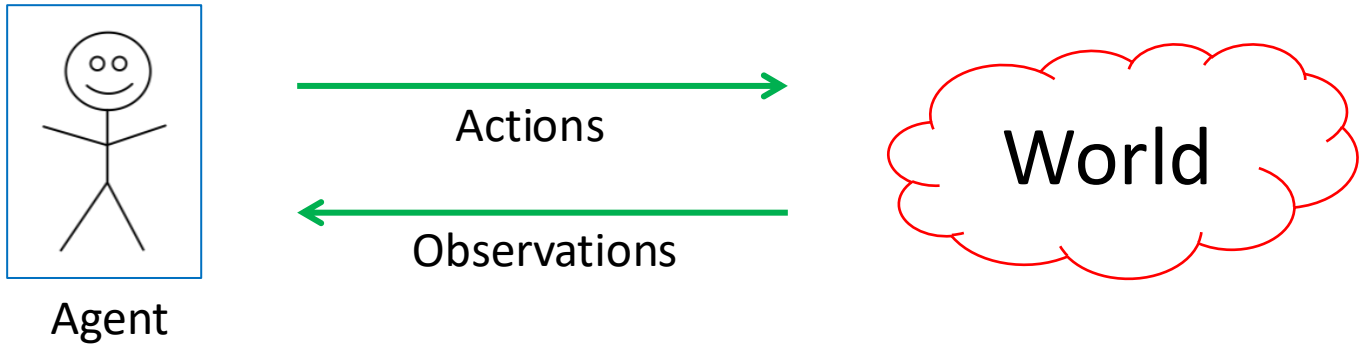


- Agent receives a reward based on state of the world
  - **Goal**: maximize reward / utility **($$$)**

RL Review: **Theoretical Model**

Basic setup:

- Set of states, S
- Set of actions A



- Information: at time t, observe state $s_t \in S$. Get reward r_t
- Agent makes choice $a_t \in A$. State changes to $s_{t+1,}$ continue

Goal: find a map from **states to actions** maximize rewards.

A "policy"

RL Review: **Markov Decision Process (MDP)**

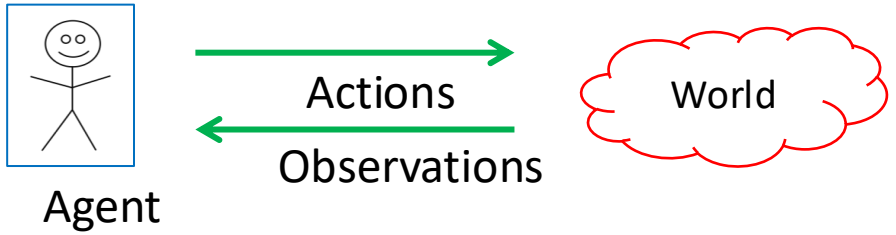
The formal mathematical model:

- **State set** S. Initial state s_0. **Action set** A
- **State transition model**: $P(s_{t+1}|s_t, a_t)$
  - Markov assumption: transition probability only depends on s_t and a_t, and not previous actions or states.
- **Reward function: *r*(s_t)**
- **Policy**: $\pi(s) : S \rightarrow A$ action to take at a particular state.

$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \dots$$

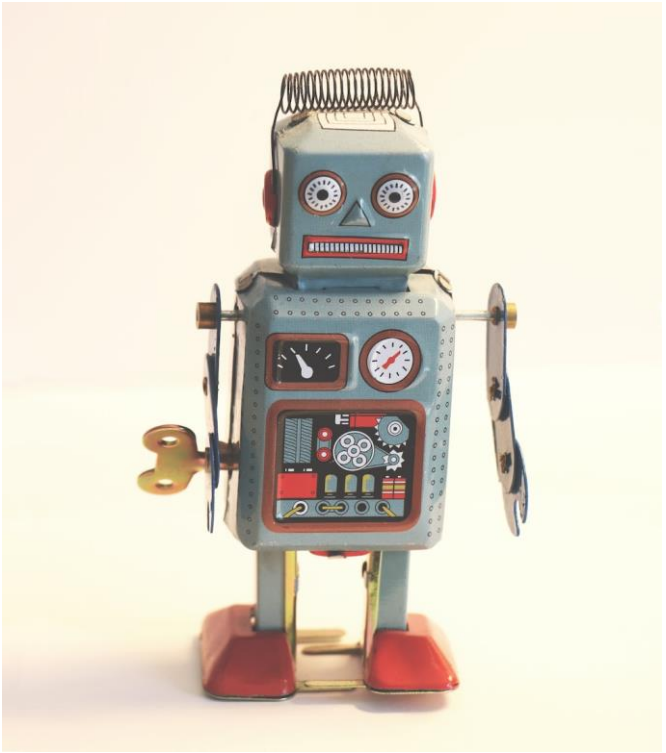
RLHF: **RL Approach**

What approach for RL stage?



- Many deep RL methods available
- Policy gradient methods
- Popular: PPO (Proximal Policy Optimization)
  - Main difference from vanilla policy gradient, you constrain change to policy at each step (Schulman et al)

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(y|x)}\left[r_\phi(x,y)\right] - \beta\mathbb{D}_{\mathrm{KL}}\left[\pi_\theta(y \mid x) \,||\, \pi_{\mathrm{ref}}(y \mid x)\right]$$



RLHF **Alternatives**

- **Direct preference optimization** (DPO)
  - Bypass separate trained reward model: just use preference information **directly** (Rafailov et al,'23)
  - **How?** Model a preference distribution from samples, integrate into a single loss (one-stage approach)

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta\log\frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right].$$

  - **Gradient step:**

$$\nabla_\theta\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$

$$-\beta\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\Big[\underbrace{\sigma(\hat{r}_\theta(x,y_l) - \hat{r}_\theta(x,y_w))}_{\text{higher weight when reward estimate is wrong}}\Big[\underbrace{\nabla_\theta\log\pi(y_w \mid x)}_{\text{increase likelihood of } y_w} - \underbrace{\nabla_\theta\log\pi(y_l \mid x)}_{\text{decrease likelihood of } y_l}\Big]\Big]$$

Break & Questions

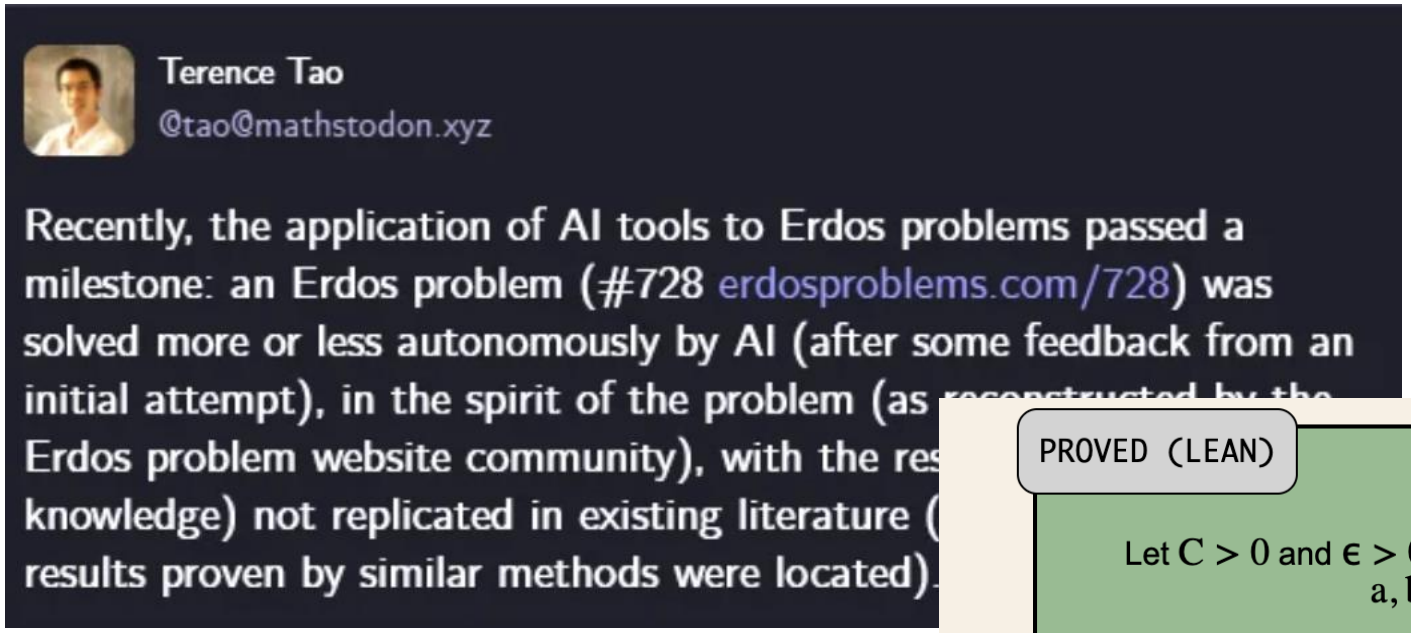
Outline

- **Review: Alignment and RLHF**
  - Basic alignment idea, goals, mechanisms, RL review, RLHF steps
- **Reasoning: Back to Chain-of-Thought**
  - Reasoning intro, chain-of-thought for reasoning, generalizations, CoT challenges and functionality
- **RL for Reasoning**
  - Notion of “verifiers”, using RL for reasoning rather than alignment

Reasoning

Perhaps the most exciting application of LLMs today:

- How do we get there? In some sense, mix CoT and RL
  - CoT: more tokens to think through, RL: think better



Terence Tao
@tao@mathstodon.xyz

Recently, the application of AI tools to Erdos problems passed a milestone: an Erdos problem (#728 erdosproblems.com/728) was solved more or less autonomously by AI (after some feedback from an initial attempt), in the spirit of the problem (as reconstructed by the Erdos problem website community), with the res knowledge) not replicated in existing literature (results proven by similar methods were located).

PROVED (LEAN)

Let $C > 0$ and $\epsilon > 0$ be sufficiently small. Are there infinitely many integers a, b, n with $a \geq \epsilon n$ and $b \geq \epsilon n$ such that

$$a!b! \mid n!(a+b-n)!$$

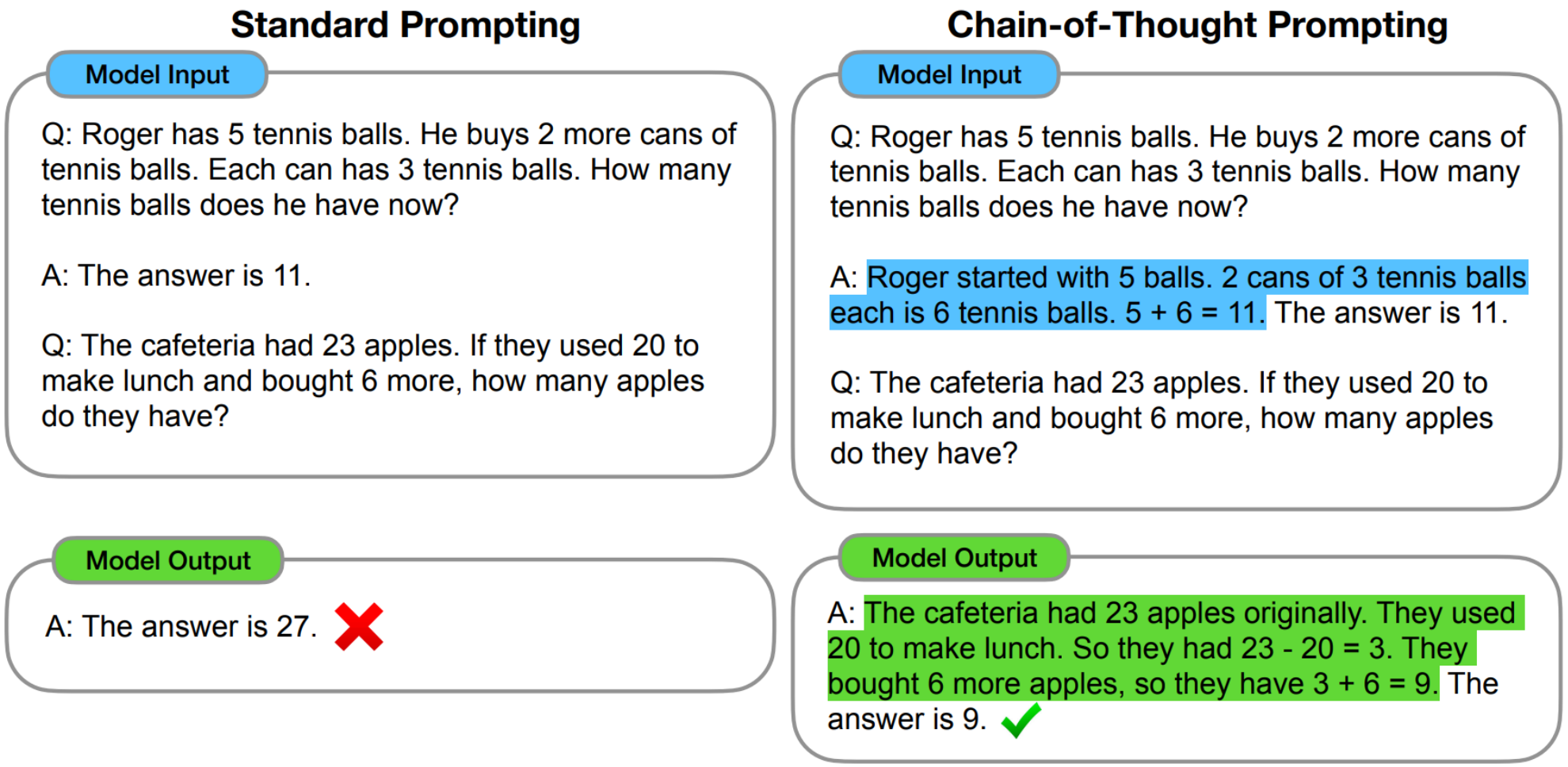
and $a + b > n + C \log n$?

#728: [EGRS75,p.91]

number theory | factorials

Starting Point: Chain-of-Thought

Performing complex reasoning is hard. Help the model:

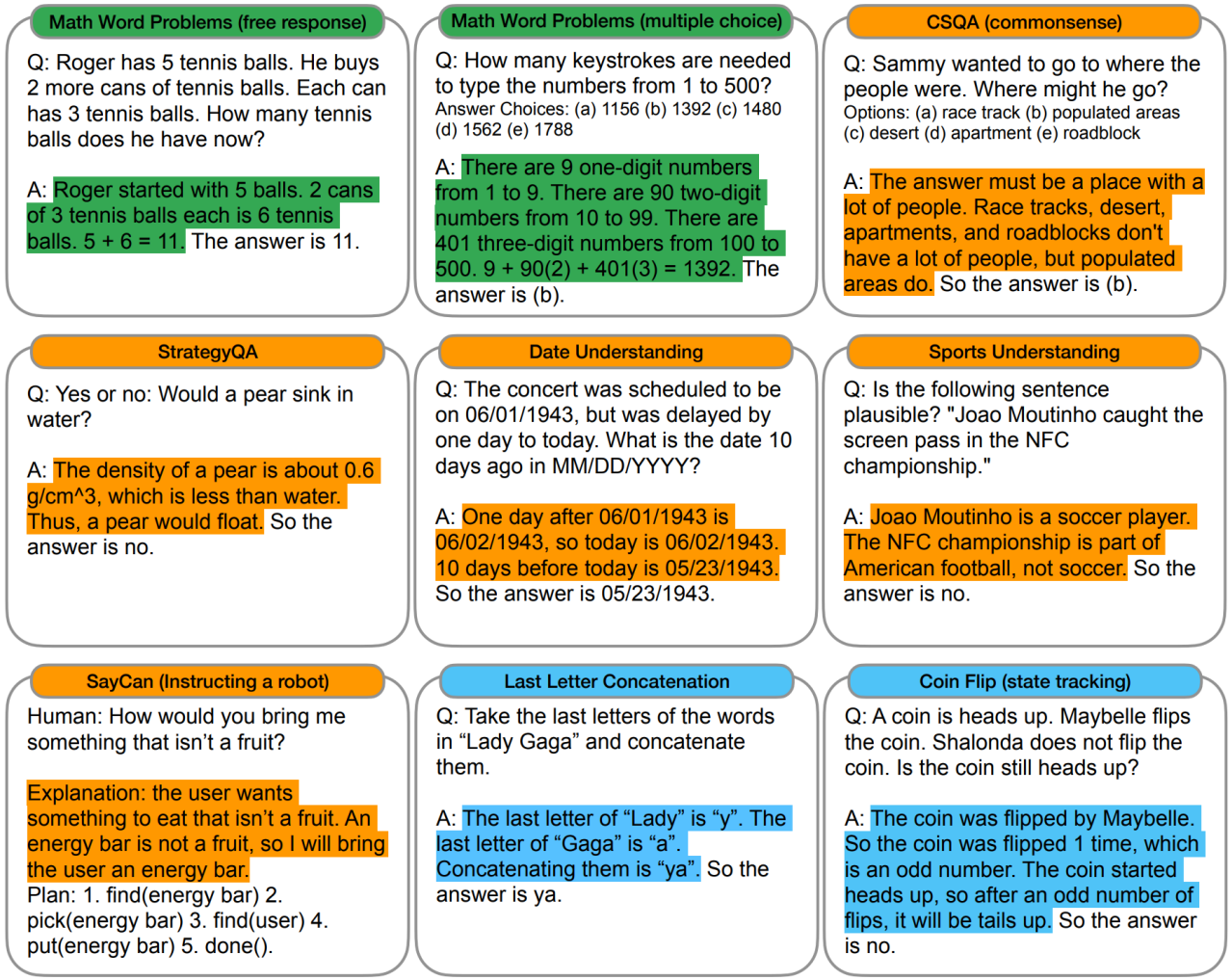


Wei et al ‘22

Chain-of-Thought: **Applications**

Some things it can be used for:

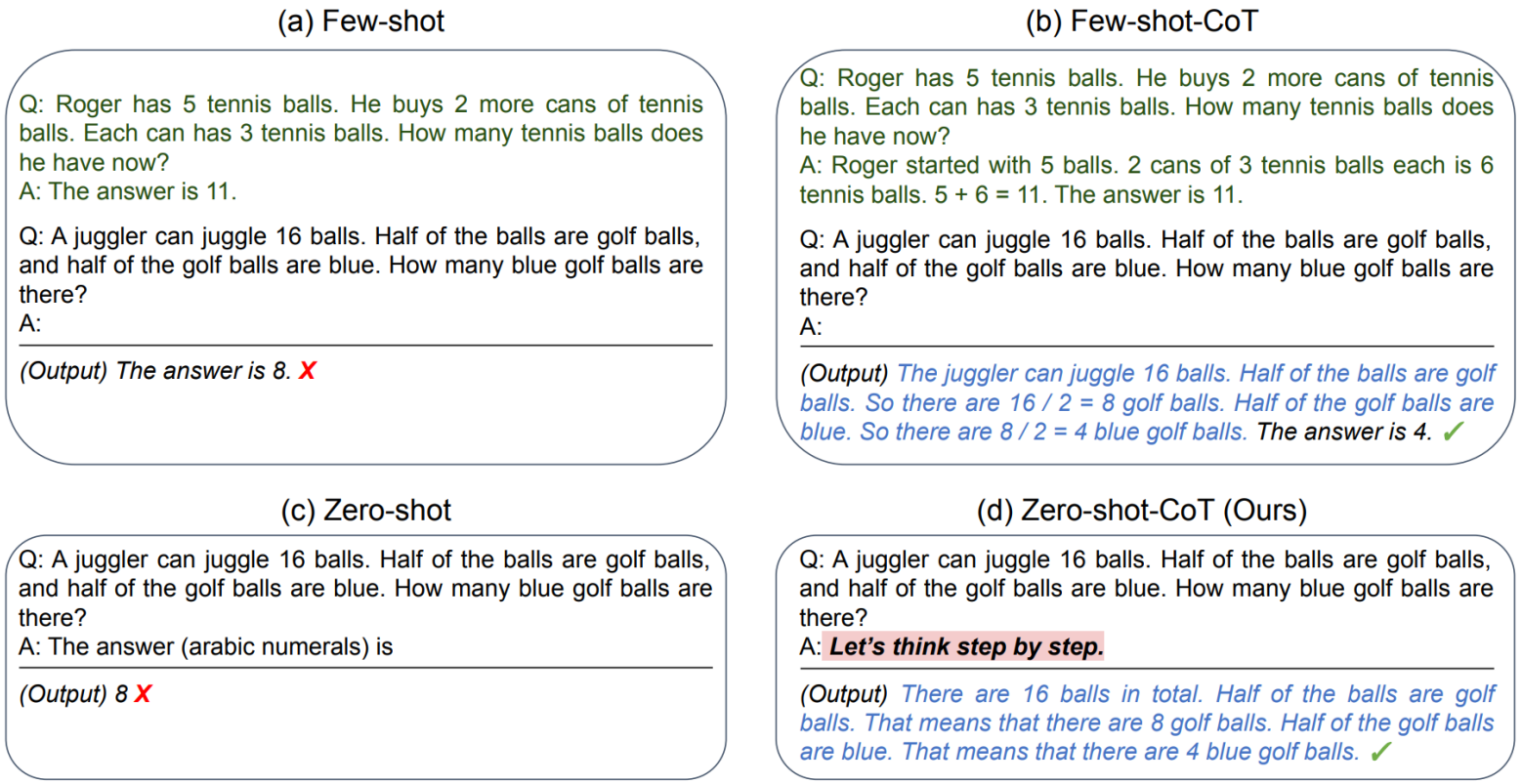
- **Math problems**
- **Commonsense reasoning**
- **Symbolic reasoning**



Chain-of-Thought: **Zero-Shot**

No examples shown; encourage model to decompose

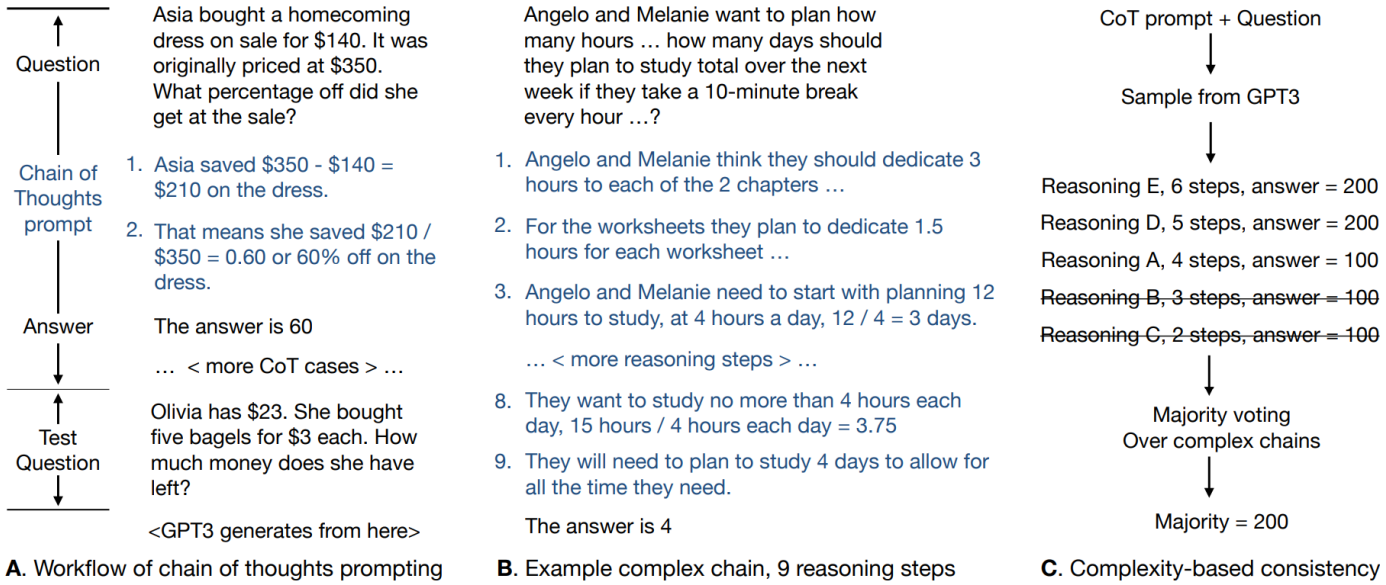
- Add to prompt: “Let’s think step by step” before each answer
- For answer extraction, add prompts like “Therefore, the answer (arabic numerals) is” (Kojima et al ‘23)



Chain-of-Thought: **Few-Shot Examples**

As before, we must choose few-shot examples.

- More structured than simple semantic similarity
- *Complexity-based* prompting.
  - "[S]imply choose complex prompts over simple ones."
- Prompting: include most steps. Ensembling: MV over set of most complex chains.

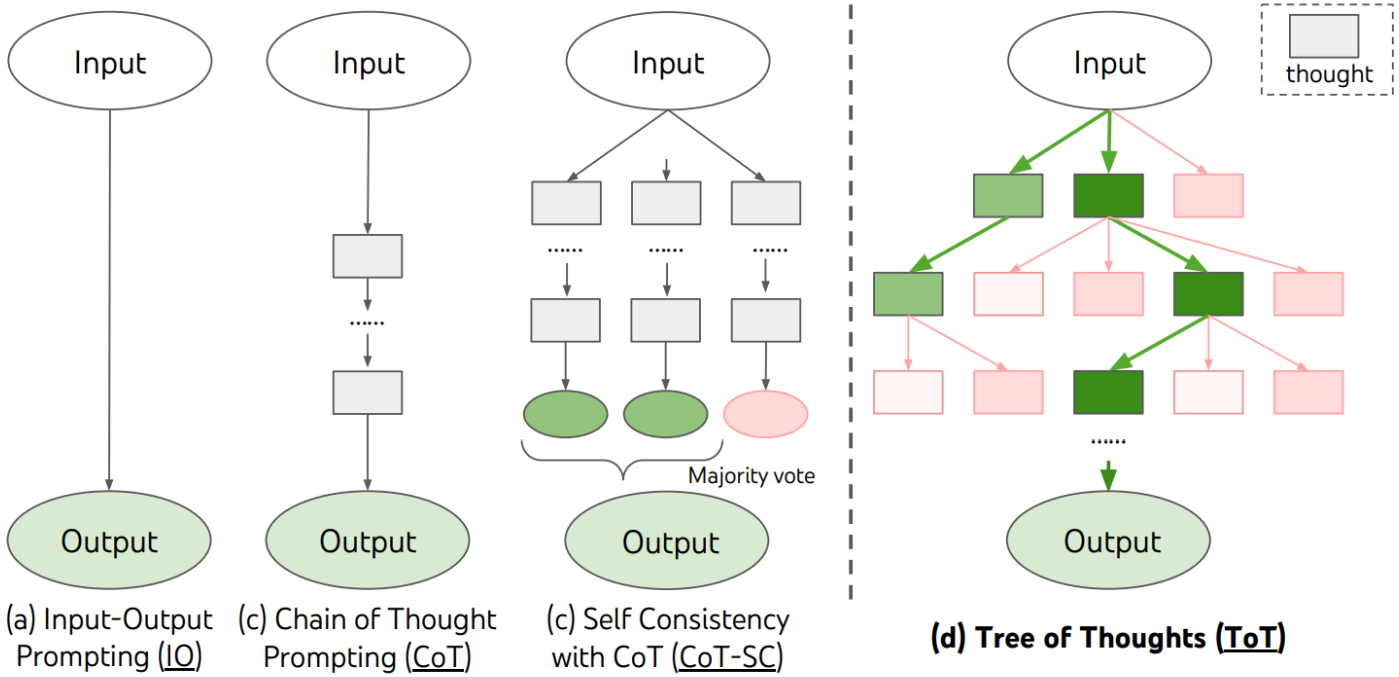


Fu et al '23

Chain-of-Thought: **Generalizations**

How do we really “reason”?

- Not really by sampling a bunch of chains…



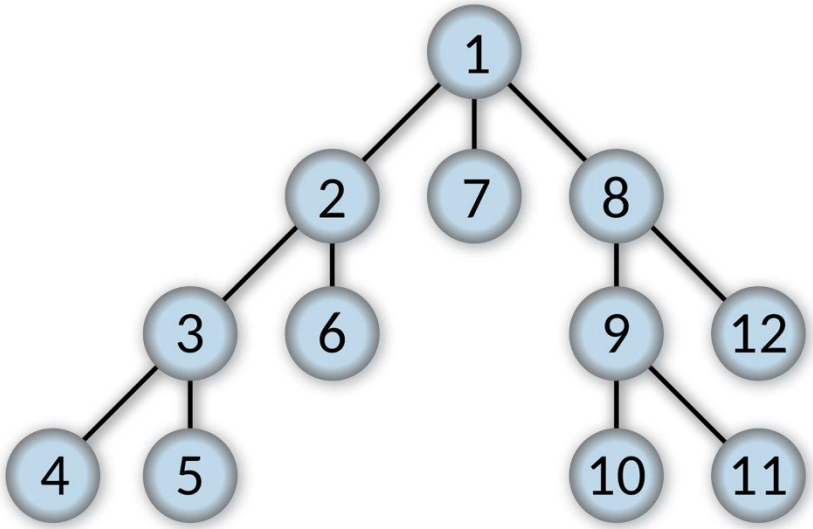
Yao et al ‘23

Chain-of-Thought: **Generalizations**

Tree-of-thoughts **basic idea**:

- **Notation**: thoughts z_1, z_2, …, z_n bridge x and y
- Comparison to other methods:
  - Vanilla CoT: sample $z_i \sim p_\theta(z_i \mid x, z_1,\ldots,z_{i-1})$, $y \sim p_\theta(y \mid x, z_1,\ldots,z_n)$
  - CoT Self-Consistency: sample multiple times, take majority vote

- Idea: create a state $s=[x,z_1,\ldots,z_n]$
- Generate multiple candidates for next state
  - Then run standard search (i.e., BFS, DFS, A*)

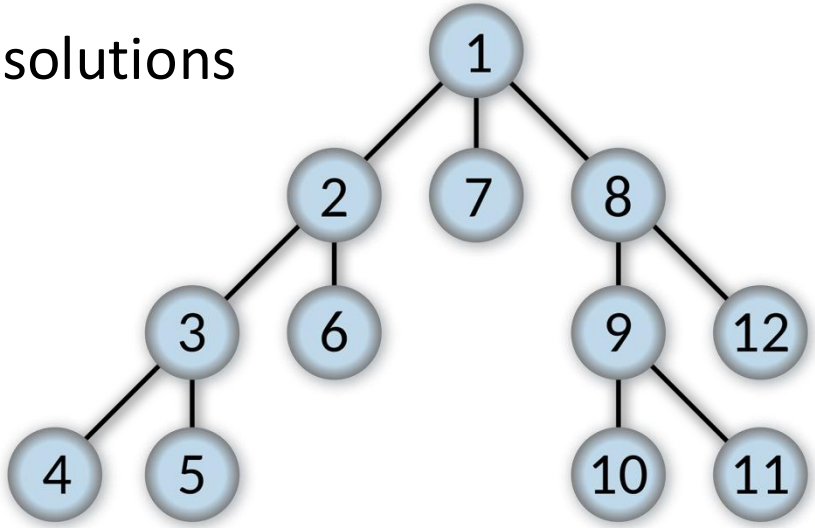


Drichel (Wiki)

Chain-of-Thought: **Generalizations**

Tree-of-thoughts **key aspects**:

- **Thought decomposition:** how big *z*s should be
- **Thought generation:** obtaining the next sample
  - Try to avoid duplication
- **State evaluation**: How close are we to solution?
  - Recall heuristics for search from CS 540
  - Either use LM itself, or vote/weighted vote across solutions
- **Search**: BFS or DFS
  - Or more advanced search methods



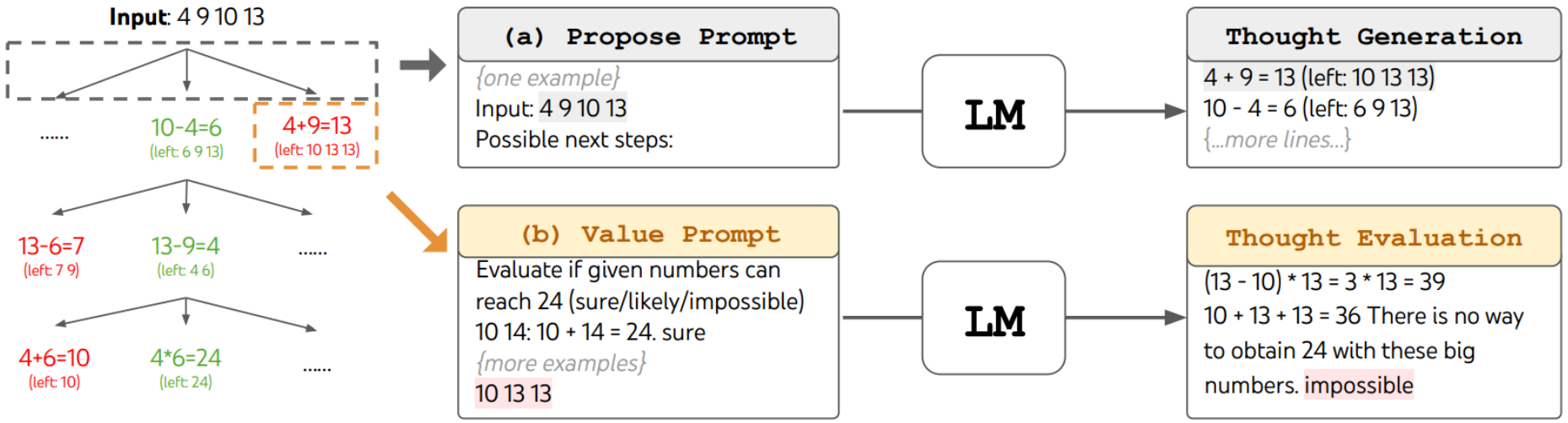
Drichel (Wiki)

Chain-of-Thought: **Generalizations**

Tree-of-thoughts **example:**

4.1 Game of 24

Game of 24 is a mathematical reasoning challenge, where the goal is to use 4 numbers and basi arithmetic operations (+-*/) to obtain 24. For example, given input "4 9 10 13", a solution outpu could be "(10 - 4) * (13 - 9) = 24".



What Matters for CoT? Scale?

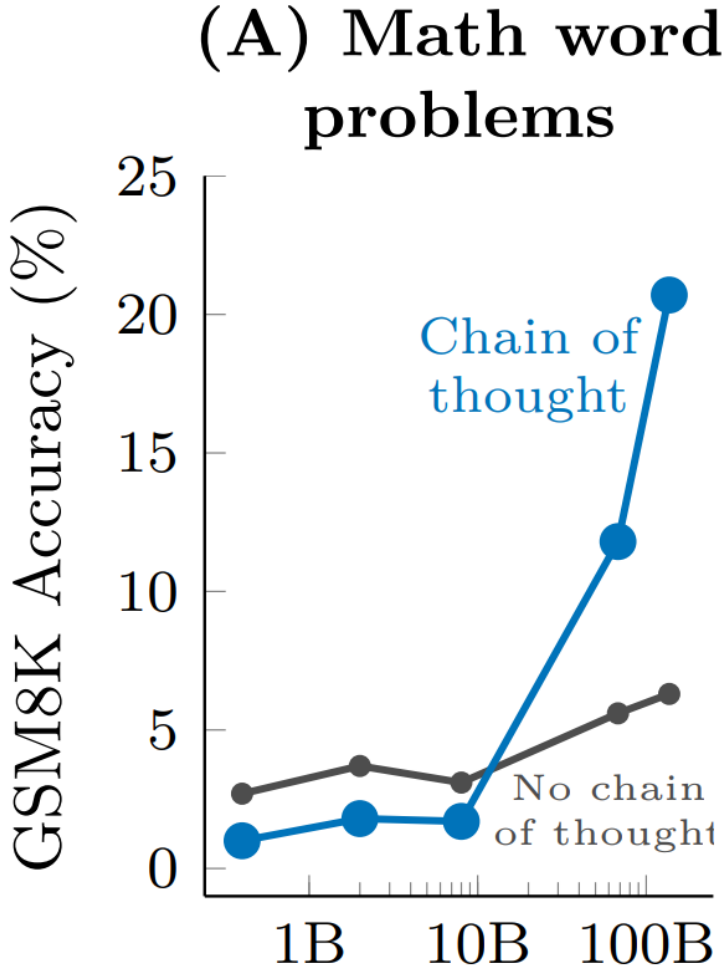
Do all language models exhibit CoT behavior?

A: No. Shows up only at certain sizes

- "Emergent behavior"

- Example: CoT does not help until ~10B

(Model: LaMDA, Dataset: Math)



Wei et al '23

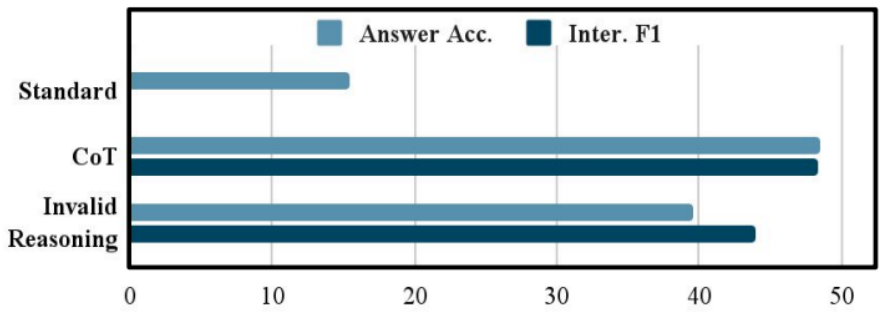
What Matters for CoT? **Correctness?**

Does reasoning have to work in provided examples?

A: **No**. Invalid reasoning in CoT still helps

- Versus no CoT
- What does matter
  - Relevant steps (still bridge gap)
  - Steps follow each other

	In-context Demonstration	Inference by LLM
Query	Leah had 32 chocolates and her sister had 42. If they ate 35, how many pieces do they have left in total?	Julie is reading a 120-page book. Yesterday, she was able to read 12 pages and today, she read twice as many pages as yesterday. If she wants to read half of the remaining pages tomorrow, how many pages should she read?
Standard	The answer is 39.	The answer is 18. ✗
CoT	Originally, Leah had 32 chocolates and her sister had 42. So in total they had 32 + 42 = 74. After eating 35, they had 74 - 35 = 39 pieces left in total. The answer is 39.	Julie is reading a 120-page book. Yesterday, she read 12 pages and today, she read 24 pages. So she read a total of 12 + 24 = 36 pages. Now she has 120 - 36 = 84 pages left. Since she wants to read half of the remaining pages, she should read 84 / 2 = 42 pages. The answer is 42. ✓
Invalid Reasoning	Originally, Leah had 32 chocolates and her sister had 42. So her sister had 42 - 32 = 10 chocolates more than Leah has. After eating 35, since 10 + 35 = 45, they had 45 - 6 = 39 pieces left in total. The answer is 39.	Yesterday, Julie read 12 pages. Today, she read 12 * 2 = 24 pages. So she read a total of 12 + 24 = 36 pages. Now she needs to read 120 - 36 = 84 more pages. She wants to read half of the remaining pages tomorrow, so she needs to read 84 / 2 = 42 pages tomorrow. The answer is 42. ✓



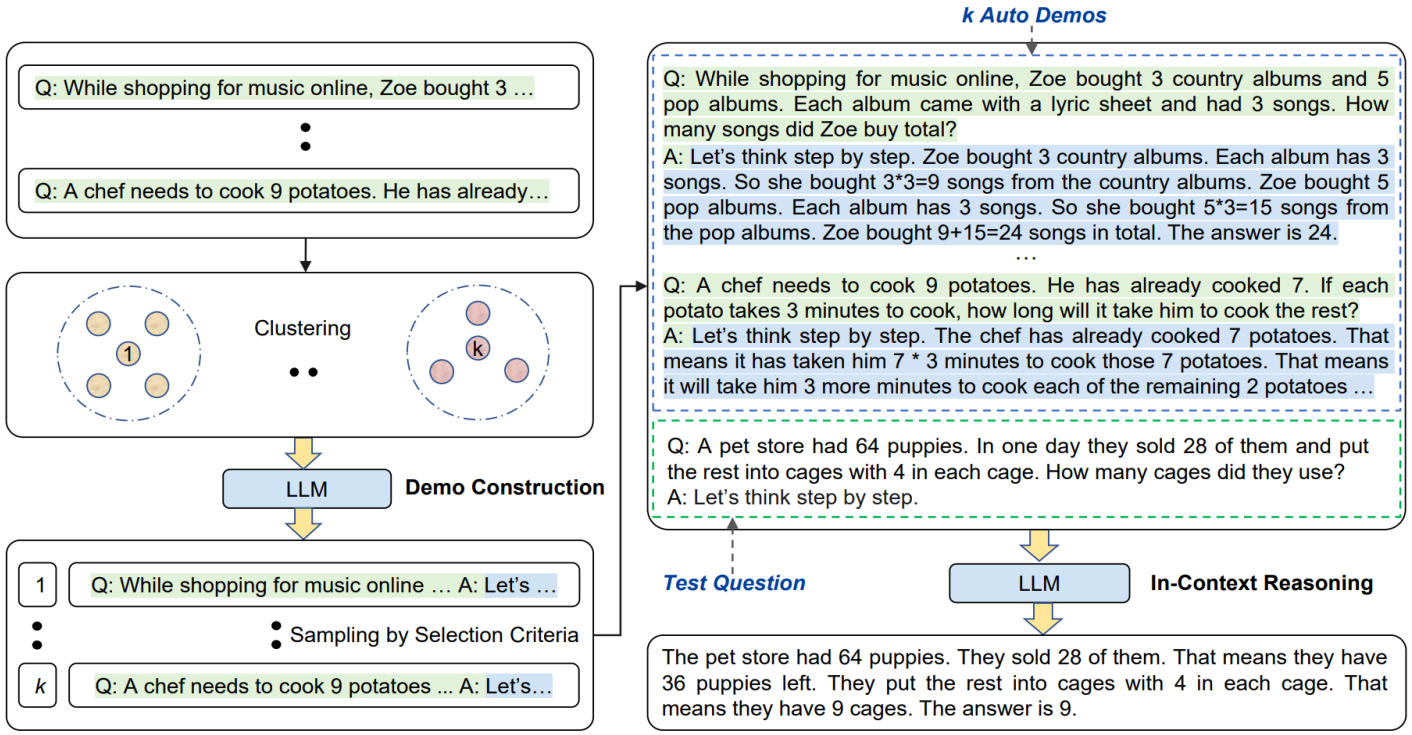
Wang et al ‘23

What Matters for CoT? **Human signal?**

Do examples have to be manually crafted?

A: **No**. Auto-CoT: generate examples to be used

- Need diversity: first cluster, then sample from each cluster



Zhang et al '23

Break & Questions

Outline

- **Review: Alignment and RLHF**
  - Basic alignment idea, goals, mechanisms, RL review, RLHF steps
- **Reasoning: Back to Chain-of-Thought**
  - Reasoning intro, chain-of-thought for reasoning, generalizations, CoT challenges and functionality
- **RL for Reasoning**
  - Notion of “verifiers”, using RL for reasoning rather than alignment

RL Outside of Alignment

- Let's get back to building **a good model**-doesn't need to be within the context of alignment
  - This means we don't have human preference data, but potentially something else

Where does RL fit in here?

- And what are the new reward models going to look like?
- One simple approach: "rewards" for just the correct answers
  - But, unlike in the supervised case, not just one solution

Verifiers

- Suppose we have a way of determining an answer is correct
  - Note: doesn't mean a full generation has to be identical to a ground-truth answer.

This is especially useful for e.g., math problems

Problem 12

The set of points in 3-dimensional coordinate space that lie in the plane $x+y+z=75$ whose coordinates satisfy the inequalities

$$x-yz<y-zx<z-xy$$

forms three disjoint convex regions. Exactly one of those regions has finite area. The area of this finite region can be expressed in the form $a\sqrt{b}$, where a and b are positive integers and b is not divisible by the square of any prime. Find $a+b$.

Reasoned about geometry and inequalities for 2m 47s ›

We will show that the (unique) finite region is a triangle. In fact, one may prove that among the four half-planes into which the two equalities

$$x-yz=y-zx \quad \text{and} \quad y-zx=z-xy,$$

split the plane $x+y+z=75$, exactly one of the four resulting sets is bounded. (In fact, a short analysis shows that writing

$$x-yz=y-zx \implies (y-x)(z-1)=0,$$

and

$$y-zx=z-xy \implies (z-y)(1+x)=0,$$

one deduces that the boundary lines are

$$L_1:\ z=1, \quad L_2:\ x=-1, \quad L_3:\ y=x, \quad L_4:\ y=z.$$

A careful discussion of the four cases – according to whether $z-1$ and $1+x$ are positive or negative – shows that the unique bounded (finite) region is given by

$$(y-x)(z-1)>0 \quad \text{and} \quad (z-y)(1+x)>0 \quad \text{with} \quad x+y+z=75,$$

which (when one accounts for the signs) is equivalent to the inequalities

$$1+x>0, \quad z-1>0, \quad y-x>0, \quad z-y>0,$$

i.e.

$$x>-1, \quad z>1, \quad x<y<z.$$

When one then "peels off" the boundary by considering the two equalities (which occur when either $y=x$ or $z=y$), one finds that the boundary lines of the finite region are exactly

$$x=-1, \quad y=x, \quad y=z.$$

A short computation shows that on the plane $x+y+z=75$ the three lines $x=-1$, $y=x$, and $y=z$ meet at the points

$$A=(-1,-1,77), \quad B=(25,25,25), \quad C=(-1,38,38).$$

These three points form a triangle that is the finite region in question.

To compute its area we may use the cross product. Define

$$\vec{AB}=B-A=(25-(-1),\ 25-(-1),\ 25-77)=(26,26,-52),$$

$$\vec{AC}=C-A=(-1-(-1),\ 38-(-1),\ 38-77)=(0,39,-39).$$

Their cross product is

$$\vec{AB}\times\vec{AC}=(1014,\ 1014,\ 1014),$$

so that

$$\|\vec{AB}\times\vec{AC}\|=1014\sqrt{3}.$$

Thus the area of the triangle is

$$\text{Area}=\frac{1}{2}\|\vec{AB}\times\vec{AC}\|=\frac{1014\sqrt{3}}{2}=507\sqrt{3}.$$

Since the area is given in the form $a\sqrt{b}$ with $a=507$ and $b=3$, the final answer is

$$a+b=507+3=510.$$

Thus, the answer is $\boxed{510}$.

Verifiers

- Note that verifiers don't need to just be answer checks
  - For example, we can write unit tests for code and use them for verification
  - Plus, lots of these out there!
  - As a result, much of **RLVR** is aimed at math and code

```
[TestMethod]
public void TestZoom()
{
  bool gestureDetected = false;
  var threadHolder = new AutoResetEvent(false);

  GestureTestFramework.Validate("Zoom", "TouchInteraction02",
    // On successful gesture detection
    (sender, e) =>
    {
      gestureDetected = true;
      if (e.Error == null)
      {
        var distanceChanged = e.Values.Get<DistanceChanged>();
        // User defined validation code
      }
      else
      {
        Assert.Fail(e.Error.Message);
      }
    });
```

Thank You!