Cognition Engineering: The Inevitable Path to AGI

Shijie Xia
Shanghai Jiao Tong University, SII
2024/11/30

Outlines

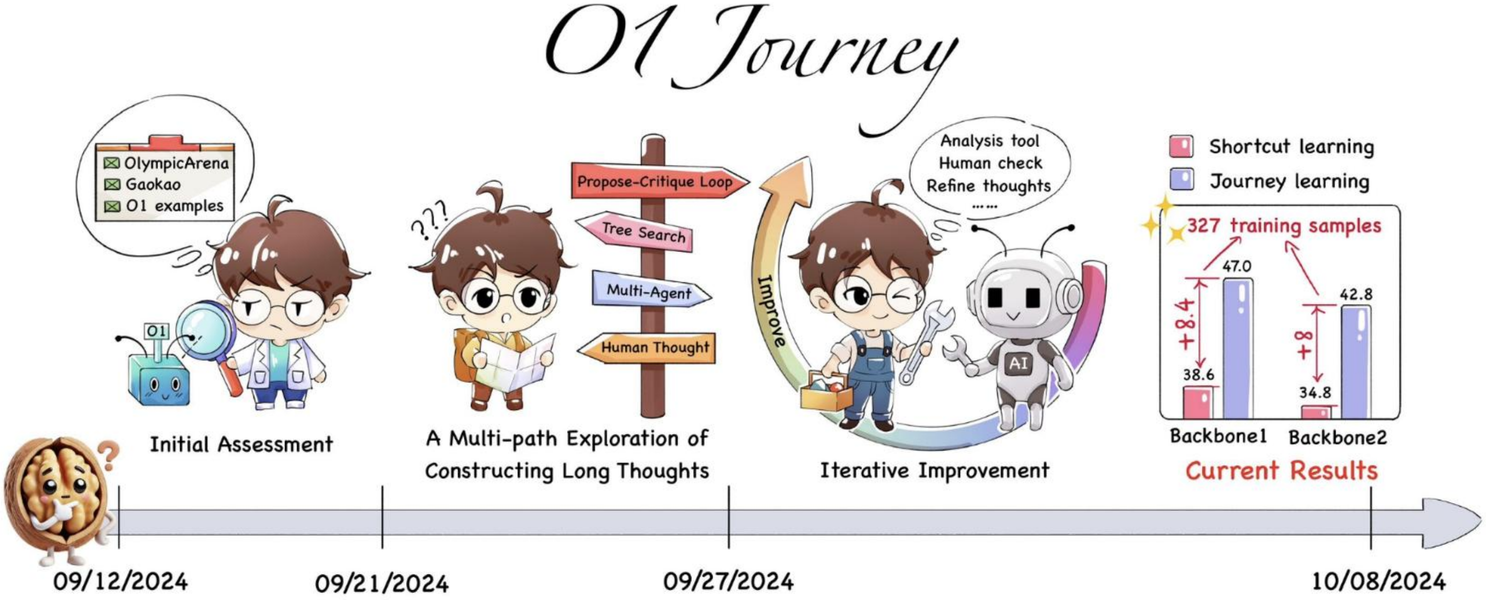
- What’s the cognition engineering?
- Why now cognition engineering?
- How to implement the cognition engineering?

Outlines

- What's the cognition engineering?
- Why now cognition engineering?
- How to implement the cognition engineering?

O1 Journey

We record the entire human thought process in solving math problems and we find ...



CoT vs. Human Thinking Process

Models lack robust iterative problem-solving capabilities

Let $a_0, a_1, a_2, \ldots$ be an infinite sequence of real numbers such that $a_0 = \frac{5}{13}$ and $a_n = 2a_{n-1}^2 - 1$ for every positive integer n. Let c be the smallest number such that for every positive integer n, the product of the first n terms satisfies the inequality $|a_0 a_1 \ldots a_{n-1}| \le \frac{c}{2^n}$. What is the value of $100c$, rounded to the nearest integer?

gpt4-o

$$a_0 = \frac{5}{13}, \quad a_1 = \frac{-119}{169}, \quad a_2 = \frac{-23439}{28561}.$$

Step 2: Analyze the product of the terms

We now examine the product of the terms up to a_{n-1}. The product of the first few terms is:

$$|a_0 a_1| = \left|\frac{5}{13} \times \frac{-119}{169}\right| = \frac{5 \times 119}{13 \times 169} = \frac{595}{2197},$$

and

$$|a_0 a_1 a_2| = \left|\frac{5}{13} \times \frac{-119}{169} \times \frac{-23439}{28561}\right| = \frac{5 \times 119 \times 23439}{13 \times 169 \times 28561}.$$

For large n, the product $|a_0 a_1 \cdots a_{n-1}|$ tends to become smaller. Our goal is to determine the smallest c such that the inequality

$$|a_0 a_1 \cdots a_{n-1}| \le \frac{c}{2^n}$$

holds for all n. This requires understanding the behavior of the sequence and the growth of the product.

Step 3: Estimate the constant c

Given that the recurrence involves squaring terms, we expect the terms a_n to rapidly decrease in magnitude. This suggests that $|a_0 a_1 \cdots a_{n-1}|$ decays exponentially, and we are looking for the smallest c such that the inequality holds.

Using numerical methods or deeper analysis of the recurrence relation, it can be determined that the smallest c is approximately $\frac{5}{8}$.

Step 4: Compute $100c$

Given that $c \approx \frac{5}{8}$, we compute

$$100c \approx 100 \times \frac{5}{8} = 62.5.$$

We have an infinite real number sequence $a_0, a_1, a_2, \ldots$, where the initial term $a_0 = \frac{5}{13}$. The recurrence relation is: $a_n = 2a_{n-1}^2 - 1$ for all $n \ge 1$. Our goal is to find the smallest constant c such that for all n, the inequality $|a_0 a_1 \cdots a_{n-1}| \le \frac{c}{2^n}$ holds true. Ultimately, we need to calculate $100c$ and round it to the nearest integer.

First, we have the recurrence formula: $a_n = 2a_{n-1}^2 - 1$. The initial condition is $a_0 = \frac{5}{13}$. To better understand this recurrence relation, we can try calculating the first few terms: $a_1 = 2a_0^2 - 1 = 2\left(\frac{5}{13}\right)^2 - 1 = 2 \times \frac{25}{169} - 1 = \frac{50}{169} - 1 = \frac{50-169}{169} = \frac{-119}{169}$. $a_2 = 2a_1^2 - 1 = 2\left(\frac{-119}{169}\right)^2 - 1 = 2 \times \frac{14161}{28561} - 1 = \frac{28322}{28561} - 1 = \frac{28322-28561}{28561} = \frac{-239}{28561}$. By calculating the first few terms, we can observe the trend of the sequence a_n. However, this does not directly help us solve the inequality problem. Therefore, we need to find a more effective method.

The original inequality is: $|a_0 a_1 \cdots a_{n-1}| \le \frac{c}{2^n}$. To simplify this inequality, we can multiply both sides by 2^n: $|2a_0 \cdot 2a_1 \cdots 2a_{n-1}| \le c$. This way, we have transformed the inequality into a product form involving $2a_i$. However, while this step simplifies the form of the inequality, it does not immediately reveal how to further handle the product of $2a_i$. Thus, we need to explore other methods to address this recurrence relation.

Observing the recurrence formula: $a_n = 2a_{n-1}^2 - 1$. This resembles the double-angle formula for the cosine function: $\cos(2\theta) = 2\cos^2\theta - 1$. This similarity leads us to the idea that perhaps we can relate the sequence a_n to the cosine function. Suppose each term a_n can be represented as the cosine of a certain angle. Specifically, let: $a_n = \cos\theta_n$. Thus, the recurrence formula becomes: $\cos\theta_n = 2\cos^2\theta_{n-1} - 1$. According to the double-angle formula, we know: $\cos(2\theta) = 2\cos^2\theta - 1$. Therefore, we can deduce: $\cos\theta_n = \cos(2\theta_{n-1})$. This implies: $\theta_n = 2\theta_{n-1}$ or $\theta_n = -2\theta_{n-1} + 2k\pi$, but here we take the positive angle. Thus, the relationship between the angles is: $\theta_n = 2\theta_{n-1}$. By iterating further, we obtain: $\theta_n = 2^n\theta_0$ where θ_0 is the initial angle corresponding to a_0. Based on our definition: $a_0 = \cos\theta_0 = \frac{5}{13}$. To find θ_0, we need to calculate: $\theta_0 = \arccos\left(\frac{5}{13}\right)$. Additionally, we can compute:

$\sin\theta_0 = \sqrt{1-\cos^2\theta_0} = \sqrt{1-\left(\frac{5}{13}\right)^2} = \sqrt{1-\frac{25}{169}} = \sqrt{\frac{144}{169}} = \frac{12}{13}$. Thus, we have $\sin\theta_0 = \frac{12}{13}$. Returning to our inequality: $|a_0 a_1 \cdots a_{n-1}| \le \frac{c}{2^n}$. Based on our definition $a_i = \cos\theta_i$, we have: $a_0 a_1 \cdots a_{n-1} = \prod_{i=0}^{n-1}\cos\theta_i$. Since $\theta_i = 2^i\theta_0$, we can express the product as: $\prod_{i=0}^{n-1}\cos(2^i\theta_0)$. Using the product formula for cosines, we know: $\prod_{i=0}^{n-1}\cos(2^i\theta_0) = \frac{\sin(2^n\theta_0)}{2^n\sin\theta_0}$. Therefore, the inequality can be transformed to: $\left|\frac{\sin(2^n\theta_0)}{2^n\sin\theta_0}\right| \le \frac{c}{2^n}$. Multiplying both sides by $2^n\sin\theta_0$ (noting that $\sin\theta_0 > 0$), we obtain: $|\sin(2^n\theta_0)| \le c \cdot \sin\theta_0$. Substituting the known $\sin\theta_0 = \frac{12}{13}$, we get: $|\sin(2^n\theta_0)| \le c \cdot \frac{12}{13}$. To ensure this inequality holds for all n, we require: $c \ge \frac{|\sin(2^n\theta_0)|}{\frac{12}{13}} = \frac{13}{12}|\sin(2^n\theta_0)|$. Since $|\sin(2^n\theta_0)| \le 1$ for all n, the smallest c must satisfy: $c \ge \frac{13}{12}$. Therefore, the smallest c is $\frac{13}{12}$. The problem requires us to calculate $100c$ and round it to the nearest integer. Based on our value for c: $100c = 100 \times \frac{13}{12} = \frac{1300}{12} = 108.\overline{3}$. Rounding $108.\overline{3}$ to the nearest integer, we obtain: $100c \approx 108$. Therefore, the final answer is 108.

Annotations by Haoyang Zou

CoT vs. Human Thinking Process

Models lack robust iterative problem-solving capabilities

? 求 $\frac{wx+xy+yz}{w^2+x^2+y^2+z^2}$ 最大值，其中 $w, x, y, z \in R^+$.

gpt4-o

w,x,y,z属于r+,求(wx+xy+yz) /(w^2 + x^2 + y^2 + z^2)最大值

我们要求解的是表达式 $\frac{wx+xy+yz}{w^2+x^2+y^2+z^2}$ 的最大值，其中 $w, x, y, z \in \mathbb{R}^+$（即所有的变量都是正实数）。

步骤 1：尝试对称化问题

我们可以通过对称化假设某些变量相等来简化问题。设 $w = z$ 和 $x = y$，这样我们就得到了一个较为对称的表达式。

步骤 2：代入对称化假设

在假设 $w = z$ 和 $x = y$ 后，表达式变为：

$$\frac{wx+xy+yz}{w^2+x^2+y^2+z^2} = \frac{wx+x^2+x^2}{w^2+2x^2+w^2} = \frac{wx+2x^2}{2w^2+2x^2}$$

简化为：

$$\frac{wx+2x^2}{2(w^2+x^2)}$$

步骤 3：进一步简化

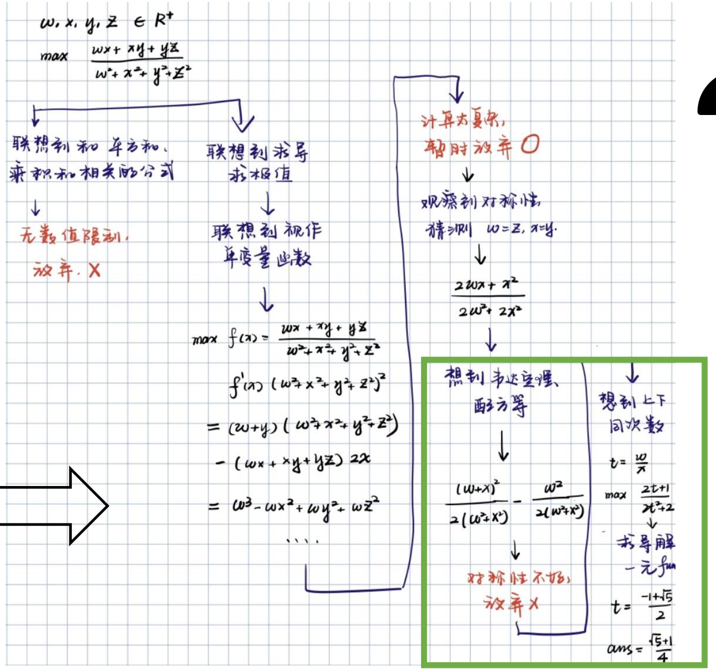
为了进一步简化这个表达式，我们考虑通过某种方法来最大化它。假设 $w = x$，则表达式变为：

$$\frac{wx+2x^2}{2w^2+2x^2} = \frac{2x^2+2x^2}{4x^2} = \frac{4x^2}{4x^2} = 1$$

因此，表达式的最大值为 1。

结论：

通过对称化和简化，可以得出原表达式的最大值为 1。



Annotations by Yixin Ye

CoT vs. Human Thinking Process

Models lack detailed thinking processes

? 设 $p(x)$ 是一个 6 次多项式，对于 $n=0,1,2,\dots 6$ 有 $p(2^n)=\frac{1}{2^n}$。求 $p(0)$

gpt4-o

我们有一个 6 次多项式 $p(x)$，且知道对于 $n=0,1,2,\dots,6$ 有以下条件：$p(2^n)=\frac{1}{2^n}$. 我们需要求 $p(0)$。首先，我们构造一个新的多项式：$q(x)=xp(x)-1$. 这是一个 6 次多项式，因为 $p(x)$ 是 6 次多项式，因此 $xp(x)$ 也是一个 6 次多项式。显然，$q(x)$ 也是一个 6 次多项式。

根据题目条件，$p(2^n)=\frac{1}{2^n}$ 对于 $n=0,1,2,\dots,6$，我们可以代入 $x=2^n$ 得到：$q(2^n)=2^np(2^n)-1=2^n\cdot\frac{1}{2^n}-1=0$. 因此，$q(2^n)=0$ 对于 $n=0,1,2,\dots,6$。这表明 $x=2^n$（对于 $n=0,1,2,\dots,6$）是 $q(x)$ 的根。

由于 $q(x)$ 是一个 6 次多项式，并且有 7 个不同的根（$x=2^0,2^1,2^2,\dots,2^6$），根据代数基本定理，$q(x)$ 必定可以写成以下形式：$q(x)=c(x-2^0)(x-2^1)(x-2^2)(x-2^3)(x-2^4)(x-2^5)(x-2^6)$，其中 c 是常数。

我们知道 $q(x)=xp(x)-1$。为了求 c，我们可以代入 $x=0$ 来得到 $q(0)$ 的值。首先计算：$q(0)=0\cdot p(0)-1=-1$. 另一方面，代入 $x=0$ 到因式分解形式中：$q(0)=c(0-2^0)(0-2^1)(0-2^2)(0-2^3)(0-2^4)(0-2^5)(0-2^6)$. 计算这个乘积：$q(0)=c(-1)(-2)(-4)(-8)(-16)(-32)(-64)$. 这个乘积等于：$(-1)^7\cdot(1\cdot2\cdot4\cdot8\cdot16\cdot32\cdot64)=-1\cdot2^21=-2^{21}$. 因此，$q(0)=-c\cdot2^{21}$. 我们知道 $q(0)=-1$，因此：$-c\cdot2^{21}=-1$, 解得：$c=\frac{1}{2^{21}}$.

步骤 5：求 $p(0)$ 我们知道 $q(x)=xp(x)-1$，因此：$xp(x)=q(x)+1$. 代入 $x=0$ 得到：$0\cdot p(0)=q(0)+1=-1+1=0$. 所以，$p(0)=0$。因此，$p(0)$ 的值为：$\boxed{0}$.

题目给出的公式是 $P(2^n)=\frac{1}{2^n}$。这个式子看起来有点复杂，但仔细观察后我们可以发现，它有一个简单的结构：对于每一个 n，$P(2^n)$ 的结果总是 2^n 的倒数。

我们首先可以尝试将它转换为一个更为一般的表达式。假设我们想要找到一个通用的多项式 $P(x)$，使得对于每一个 $x=2^n$，$P(2^n)$ 都成立。于是我们可以将 $P(2^n)$ 的式子写成：$P(x)=\frac{1}{x}$，这一点非常重要，因为我们将 x 看作是一个变量，而不仅仅是 2^n 的幂次。现在我们可以继续推理。

现在我们已经有了一个初步的发现，接下来我们尝试进行一些操作，看看如何简化这个问题。首先，我们将 $P(x)$ 改写成更具一般性的形式：$xP(x)-1=0$。为什么我们要这么做呢？因为将等式两边乘以 x 可以消掉分母，变成一个整式方程。这一步对后续的推理非常关键，后面我们会一步步进行验证。

题目接下来告诉我们，给定的多项式 $P(2^n)=\frac{1}{2^n}$ 在 $n=0,1,2,\dots,6$ 时成立。也就是说，当 x 取 $x=1,2,4,\dots,64$ 这些值时，方程 $xP(x)-1=0$ 有解。这些 x 值可以看作是方程的根，因此我们可以得出：$f(x)=xP(x)-1=0$。在 $x=1,2,4,8,16,32,64$ 时，方程 $f(x)$ 的值为 0，意味着这些点是 $f(x)$ 的 7 个根。此时我们已经知道了 7 个根的位置，但多项式的次数是多少呢？这就需要进一步思考了。

题目中提到 $P(x)$ 是一个 6 次多项式，因此 $f(x)=xP(x)-1$ 是一个 7 次多项式。而由于我们已经有了 7 个根，一个 n 次多项式最多有 n 个根，所以我们获得了 $f(x)$ 的所有根。多项式的形式一般可以表示为 $k(x-x_1)(x-x_2)\dots(x-x_7)$，其中 $x_1,x_2,\dots,x_7$ 是已知的根，而 k 是一个待定常数。

根据上面的分析，我们可以将多项式 $f(x)$ 写成：$f(x)=k(x-1)(x-2)(x-4)(x-8)(x-16)(x-32)(x-64)$。这个表达式看起来非常复杂，但别担心，我们已经知道了所有的根，因此接下来要做的就是确定常数 k 的值。

为了确定 k，我们可以利用常数项的计算。观察 $f(x)=xP(x)-1$，当 $x=1$ 时，常数项为 -1。那么我们可以将已知根带入上面的多项式，来解出 k 的值。带入 $x=2$ 时，k 满足以下方程：$-1=k\cdot2\cdot2^2\cdot2^3\cdot2^4\cdot2^5\cdot2^6$。化简这个式子，我们可以得到：$-1=k\cdot2^{21}$。因此，$k=-\frac{1}{2^{21}}$。

我们现在已经找到了 k 的值，可以确定整个多项式的具体形式了。此时，我们知道整个多项式可以写成：$f(x)=-\frac{1}{2^{21}}(x-1)(x-2)(x-4)(x-8)(x-16)(x-32)(x-64)$。接下来我们要思考的问题是，如何利用这个表达式求解 $P(0)$。

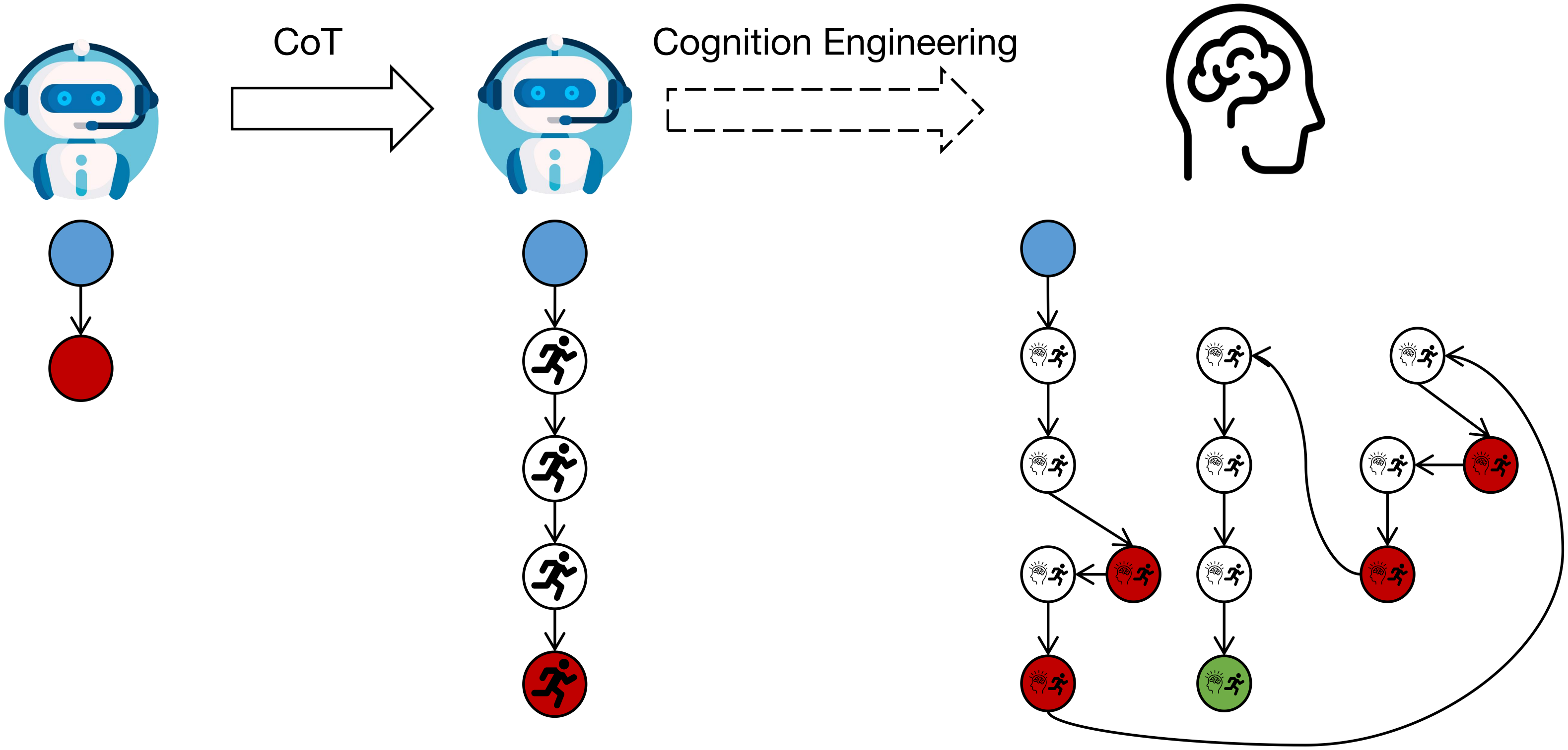
直接展开表达式太过复杂。我们需要求 $P(0)$，而观察 $f(x)=xP(x)-1$ 可以发现，$P(0)$ 就是右侧常数项的值。因此，我们可以解出 $P(0)$ 为：$P(0)=-k*\sum_{i=0,1,\dots6}\frac{2^21}{2^i}=\sum_{i=0,1,\dots6}=\frac{1}{2^i}=\frac{127}{64}$

通过仔细的观察和推理，我们一步步找到了 $P(0)$ 的值为 $\frac{127}{64}$。

Annotations by Haoyang Zou, Yixin Ye

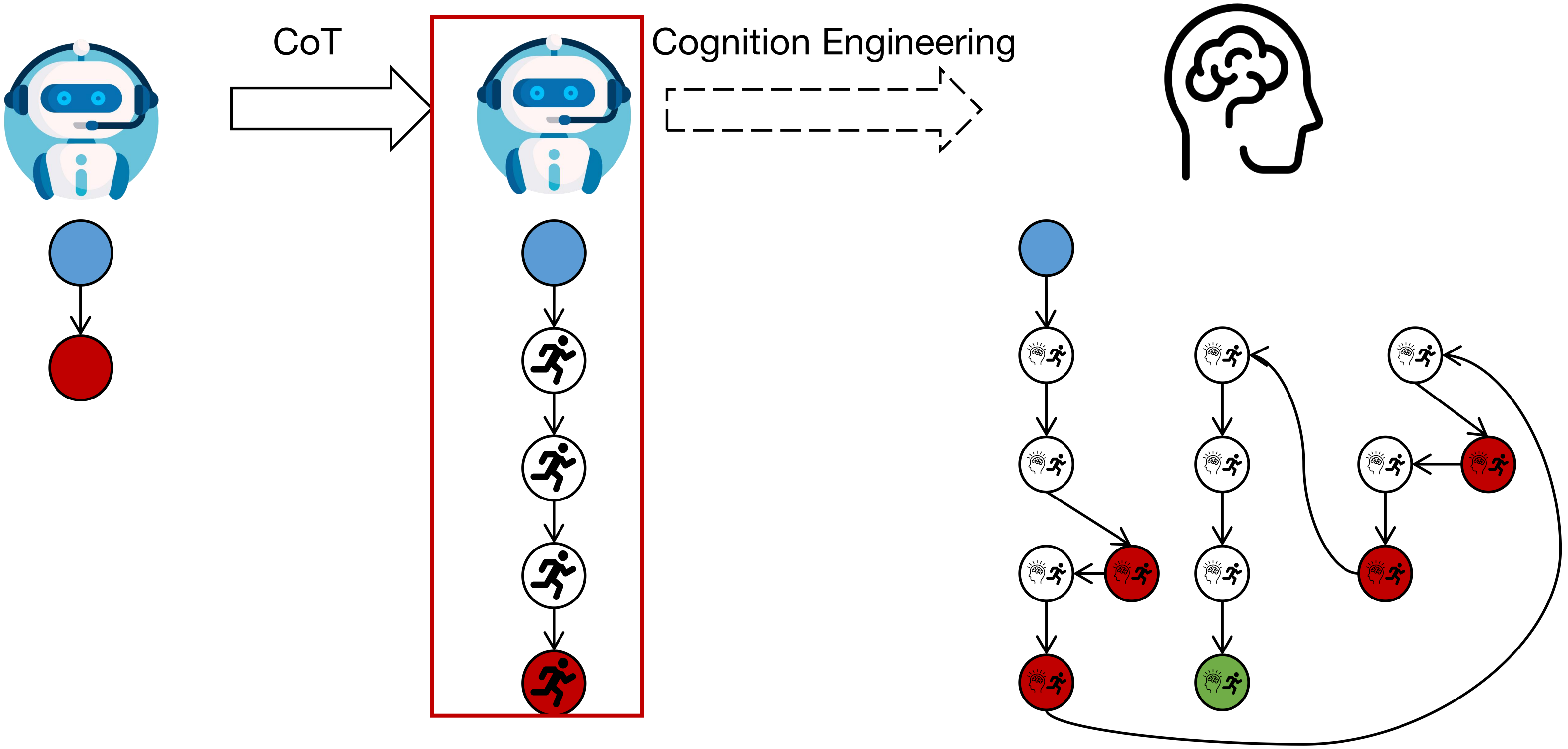
Cognition Engineering

Master the entire human cognitive process



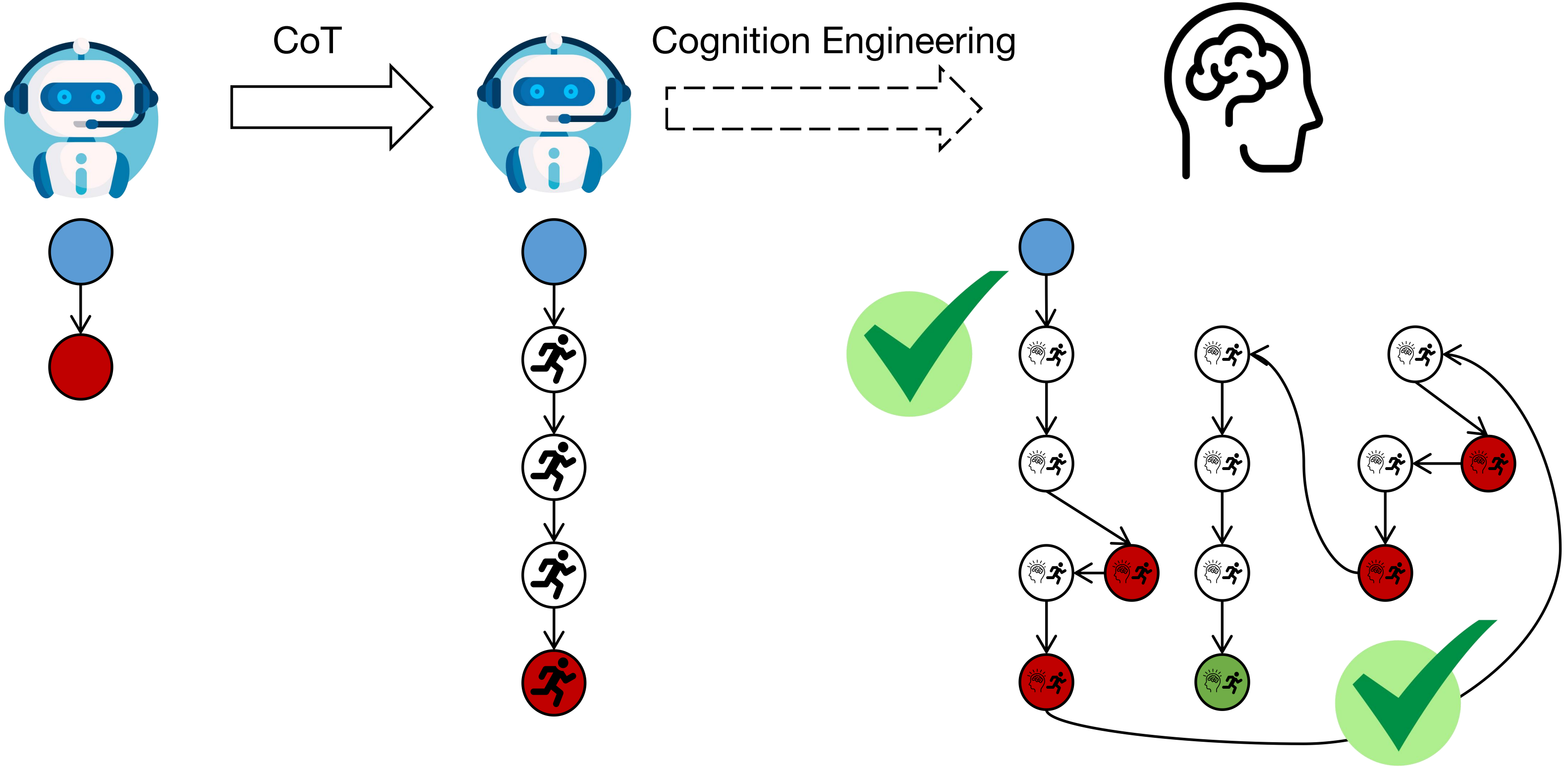
Cognition Engineering

CoT: the importance of the intermediate steps



Cognition Engineering

Cognition Engineering: the importance of the entire human thought process



Outlines

- What's the cognition engineering?
- Why now cognition engineering?
- How to implement the cognition engineering?

OpenAI's Five Steps to AGI

书生·浦语

Meta 文心一言

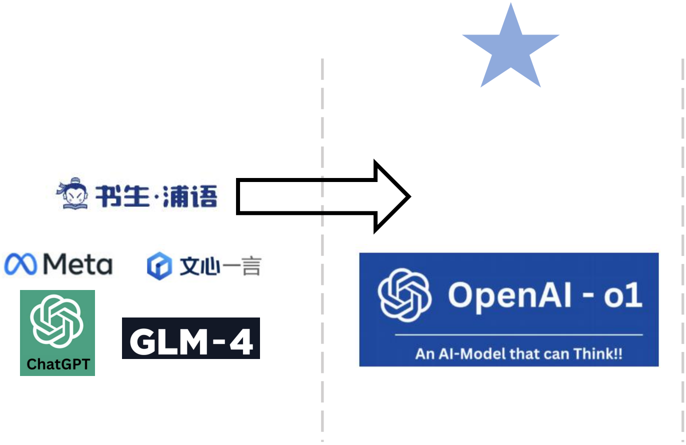
ChatGPT GLM-4

OpenAI - o1

An AI-Model that can Think!!

Level 1: Chatbots	Level 2: Reasoners	Level 3: Agents	Level 4: Innovators	Level 5: Organizations
AI with conversational language	Human-level problem solving	Systems that can take actions	AI that can aid in invention	AI that can do the work of an organization

Cognition Engineering Advances the Models to Reasoners

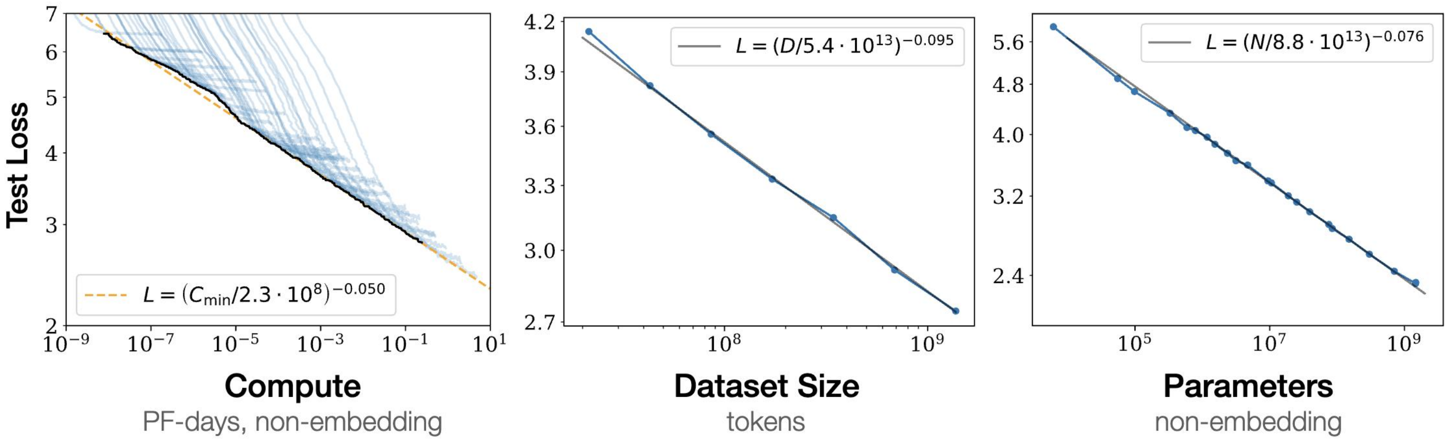


Level 1: Chatbots → **Level 2: Reasoners** → **Level 3: Agents** → **Level 4: Innovators** → **Level 5: Organizations**

- Level 1: Chatbots — AI with conversational language
- Level 2: Reasoners — Human-level problem solving
- Level 3: Agents — Systems that can take actions
- Level 4: Innovators — AI that can aid in invention
- Level 5: Organizations — AI that can do the work of an organization

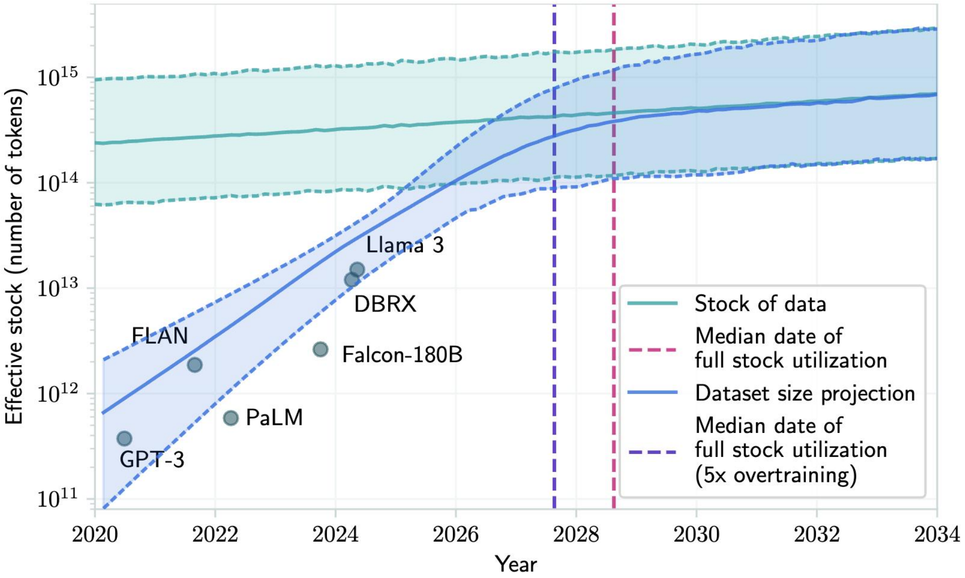
The Success of Scaling Law

The development in level 1 AGI is mainly attributed to the success of the scaling law.



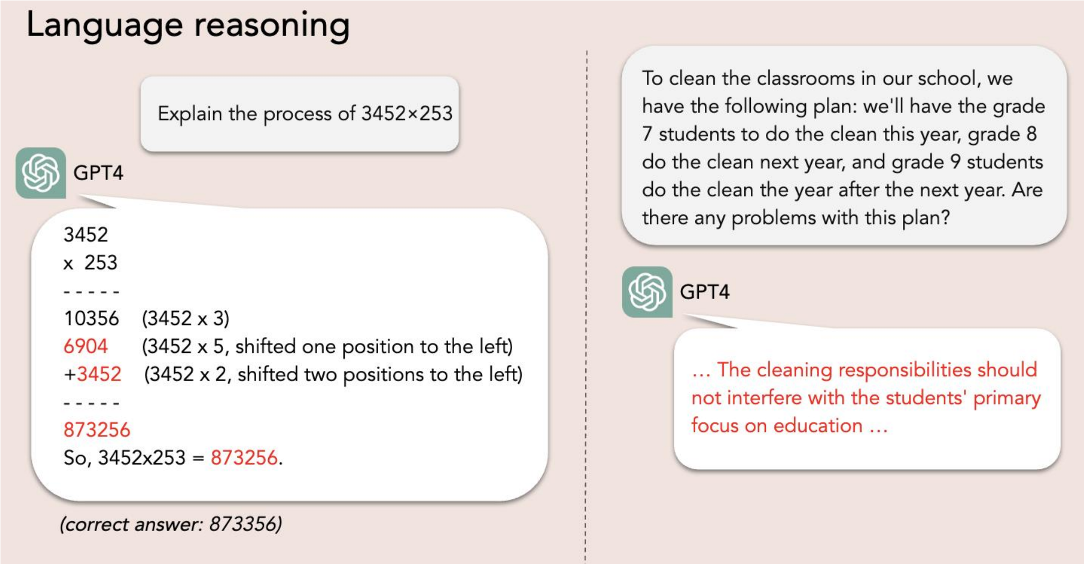
The Bottleneck of Public Human Text Data

Models will be trained on datasets roughly equal in size to the available stock of public human text data between 2026 and 2032.



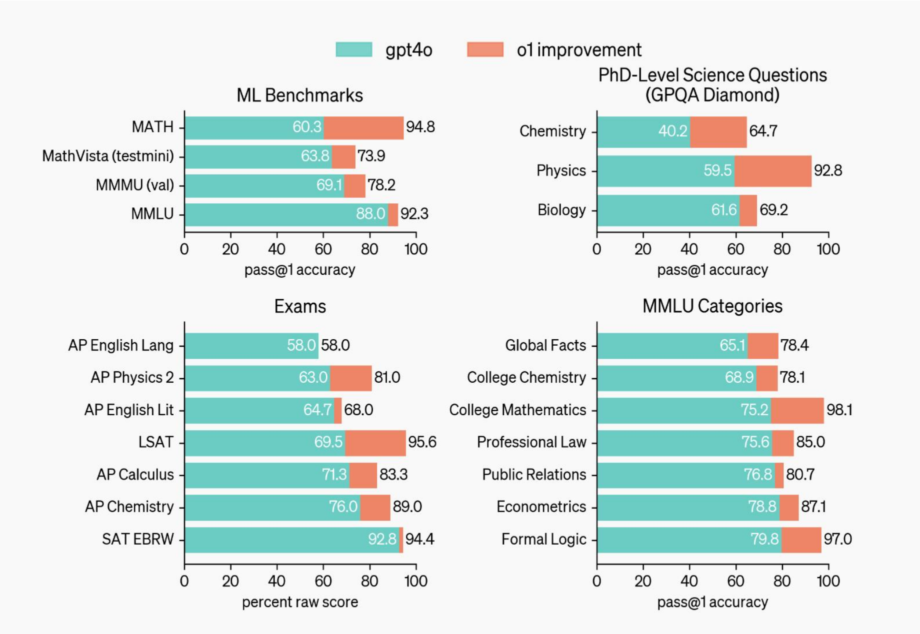
Will we run out of data? Limits of LLM scaling based on human-generated data, in ICML 2024

The Advanced Models are Still a Long Way from Level 2 Reasoners

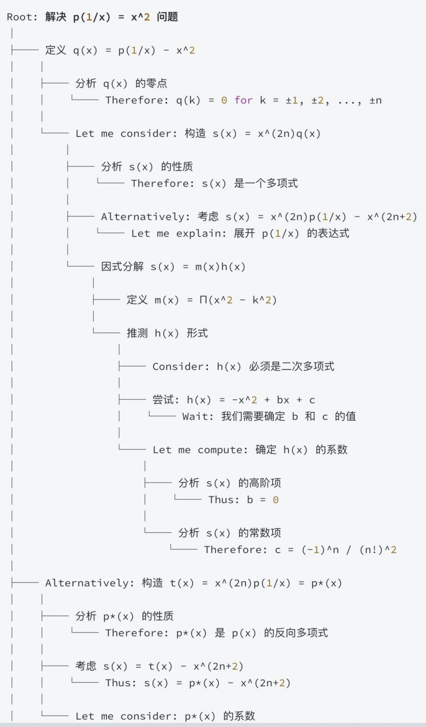


Language Models, Agent Models, and World Models: The LAW for Machine Reasoning and Planning, in arxiv 2023

The O1 Preliminary Demonstrates the Key Role of Cognition Engineering in Reasoning



The performance of o1



```
Root: 解决 p(1/x) = x^2 问题
|
├──── 定义 q(x) = p(1/x) - x^2
|     |
|     ├──── 分析 q(x) 的零点
|     |     └──── Therefore: q(k) = 0 for k = ±1, ±2, ..., ±n
|     |
|     └──── Let me consider: 构造 s(x) = x^(2n)q(x)
|           |
|           ├──── 分析 s(x) 的性质
|           |     └──── Therefore: s(x) 是一个多项式
|           |
|           ├──── Alternatively: 考虑 s(x) = x^(2n)p(1/x) - x^(2n+2)
|           |     └──── Let me explain: 展开 p(1/x) 的表达式
|           |
|           └──── 因式分解 s(x) = m(x)h(x)
|                 |
|                 ├──── 定义 m(x) = Π(x^2 - k^2)
|                 |
|                 └──── 推测 h(x) 形式
|                       |
|                       ├──── Consider: h(x) 必须是二次多项式
|                       |
|                       ├──── 尝试: h(x) = -x^2 + bx + c
|                       |     └──── Wait: 我们需要确定 b 和 c 的值
|                       |
|                       └──── Let me compute: 确定 h(x) 的系数
|                             |
|                             ├──── 分析 s(x) 的高阶项
|                             |     └──── Thus: b = 0
|                             |
|                             └──── 分析 s(x) 的常数项
|                                   └──── Therefore: c = (-1)^n / (n!)^2
|
├──── Alternatively: 构造 t(x) = x^(2n)p(1/x) = p*(x)
|     |
|     ├──── 分析 p*(x) 的性质
|     |     └──── Therefore: p*(x) 是 p(x) 的反向多项式
|     |
|     ├──── 考虑 s(x) = t(x) - x^(2n+2)
|     |     └──── Thus: s(x) = p*(x) - x^(2n+2)
|     |
|     └──── Let me consider: p*(x) 的系数
```

The hidden thinking process of o1 contains the intrinsic human-like cognitive process.

Cognition Engineering Provides the Foundation for the Next Level of AGI

书生·浦语

Meta 文心一言

ChatGPT GLM-4

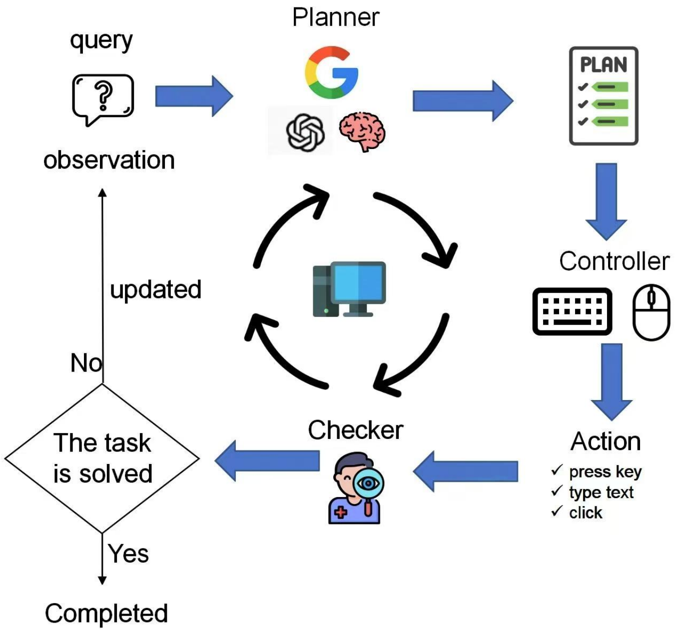
OpenAI - o1

An AI-Model that can Think!!

Level 1: Chatbots	Level 2: Reasoners	Level 3: Agents	Level 4: Innovators	Level 5: Organizations
AI with conversational language	Human-level problem solving	Systems that can take actions	AI that can aid in invention	AI that can do the work of an organization

An Example: the Level 3 Agents

The model needs to iteratively refine its actions according to the state of the tasks, which requires strong cognitive abilities, such as exploration, reflection



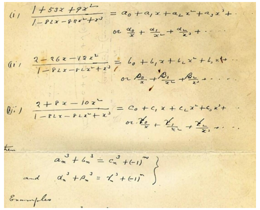
utilizing LLMs/LMMs for operate computers automatically

An Example: the Level 4 Innovators

generate new knowledge from existing knowledge

mathematics

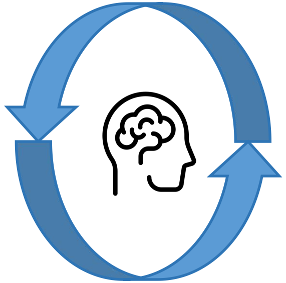
unsolved problems for humans, e.g., the Riemann Hypothesis.



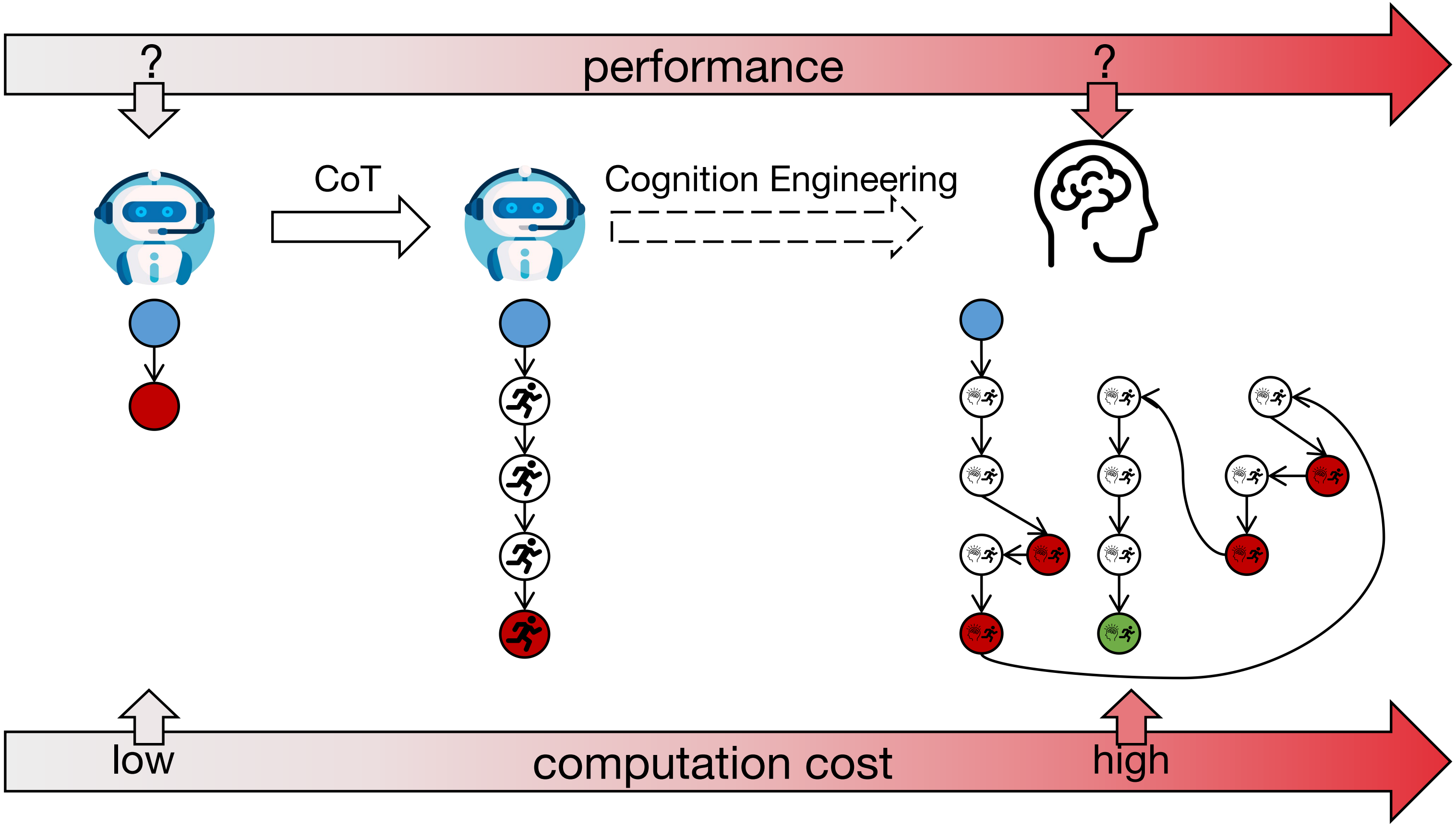
❌ No shortcuts

cognition engineering

experimental disciplines

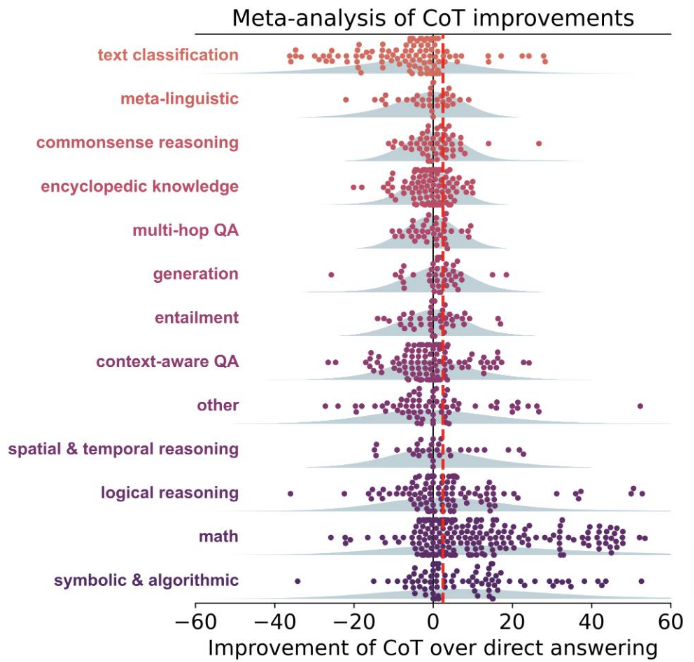


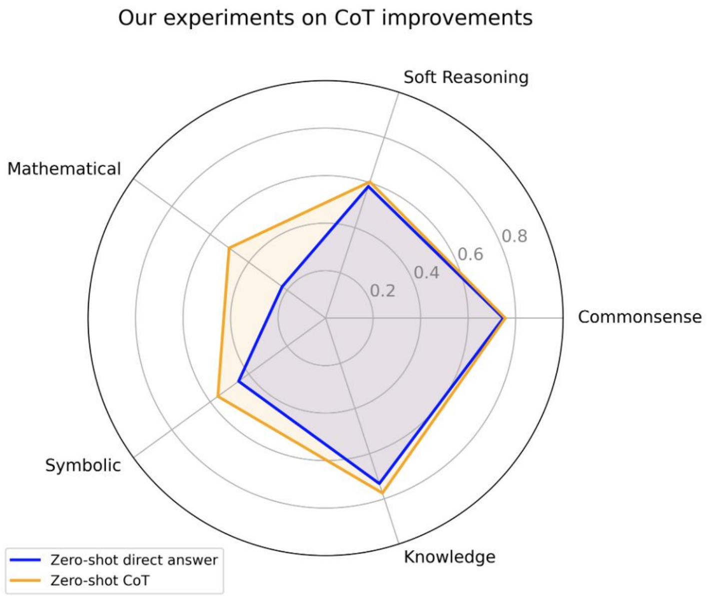
The Computation Cost of Cognition Engineering



The Goal of Cognition Engineering

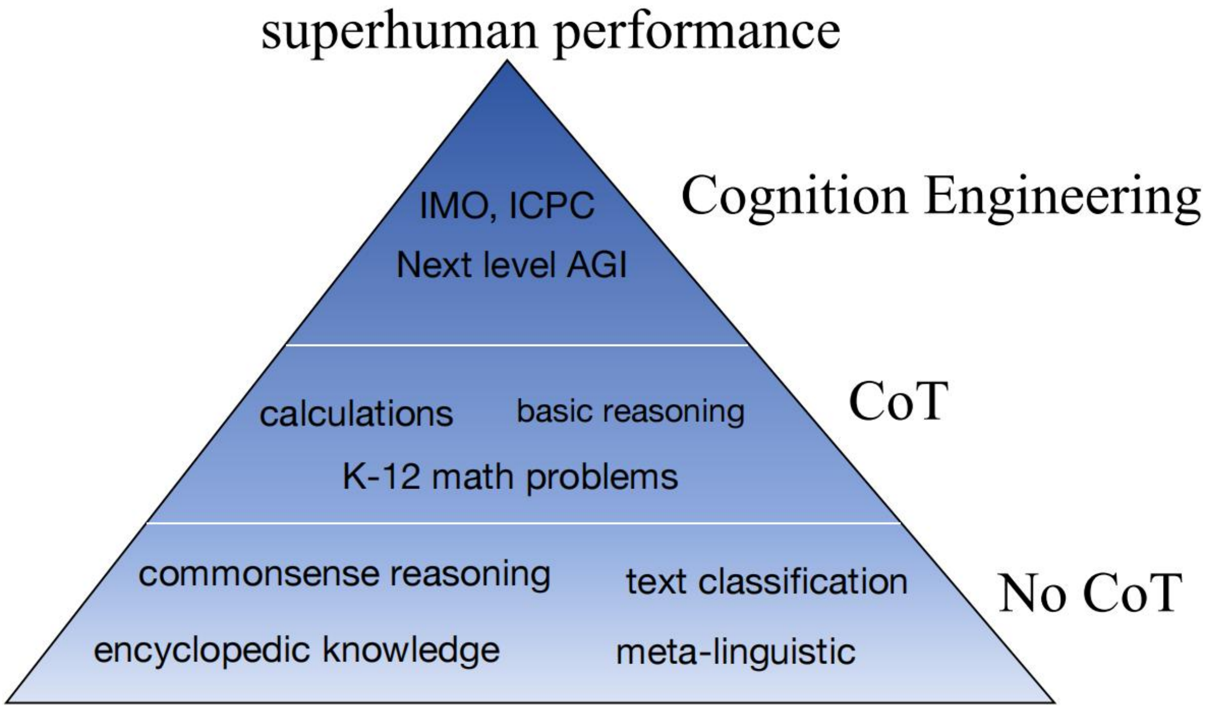
CoT gives strong performance benefits primarily on tasks involving math or logic, with much smaller gains on other types of tasks





To Cot or not to Cot? Chain-of-thought helps mainly on math and symbolic reasoning, in arxiv 2024

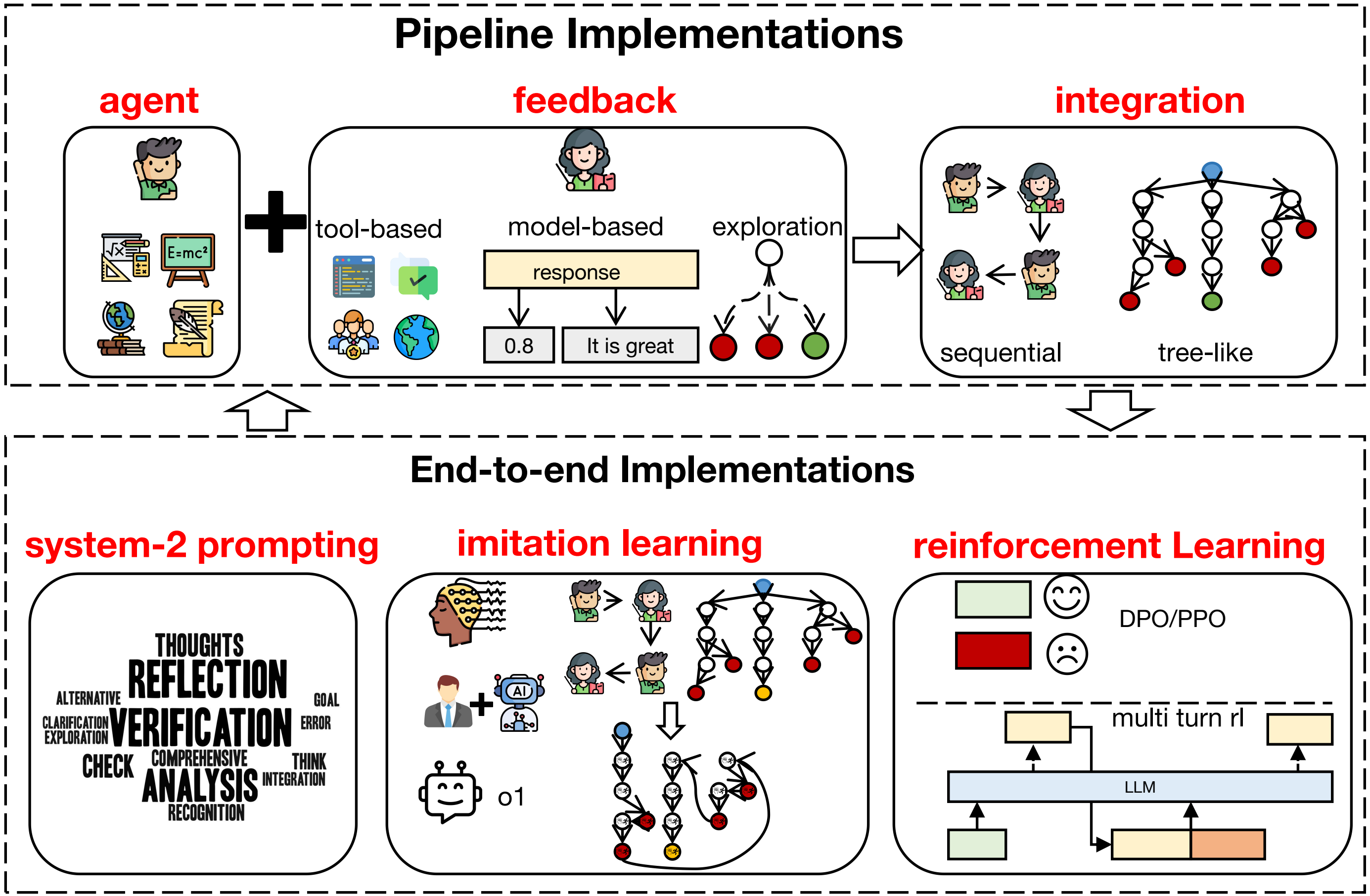
The Goal of Cognition Engineering



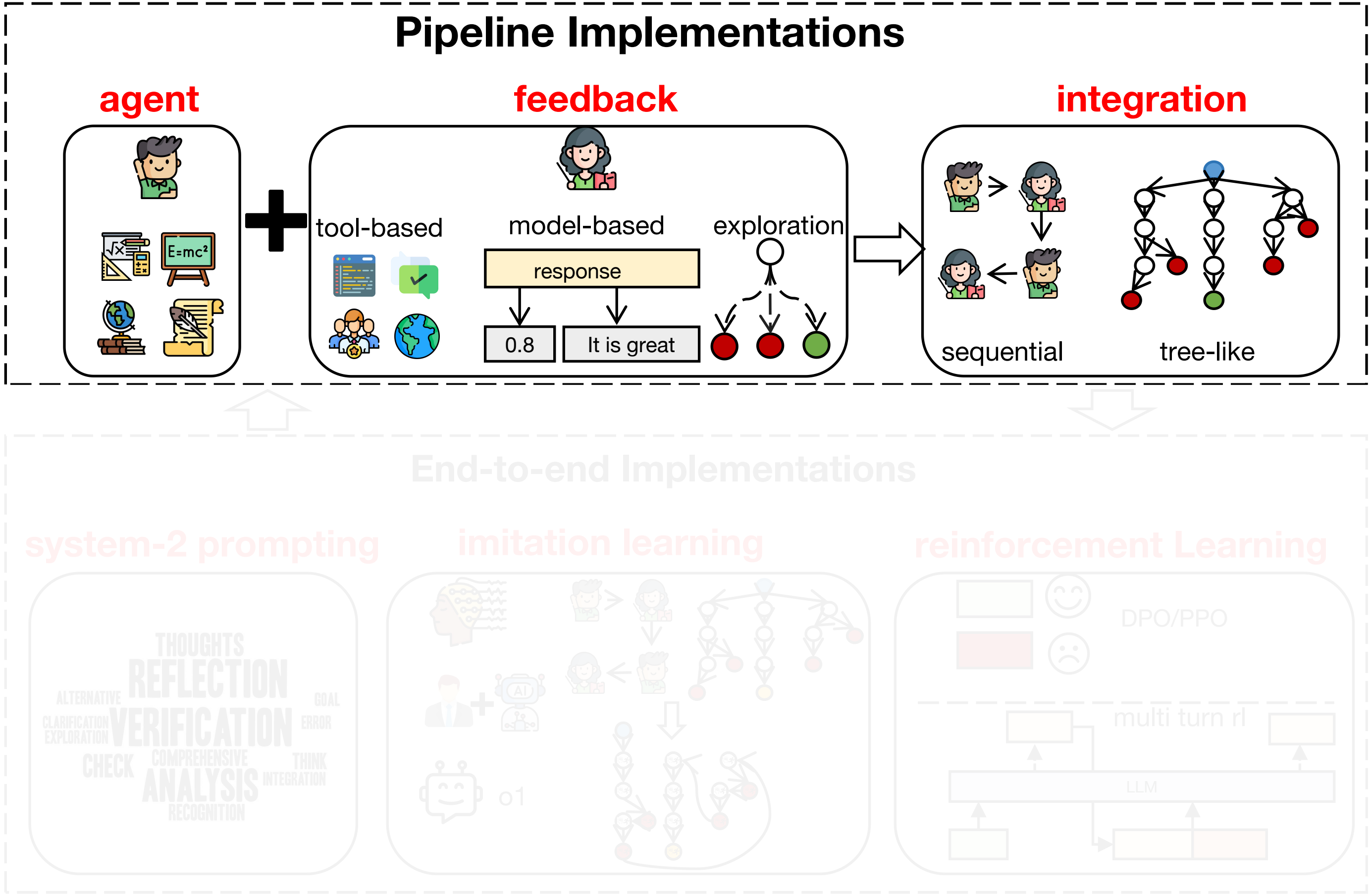
Outlines

- What's the cognition engineering?
- Why now cognition engineering?
- How to implement the cognition engineering?

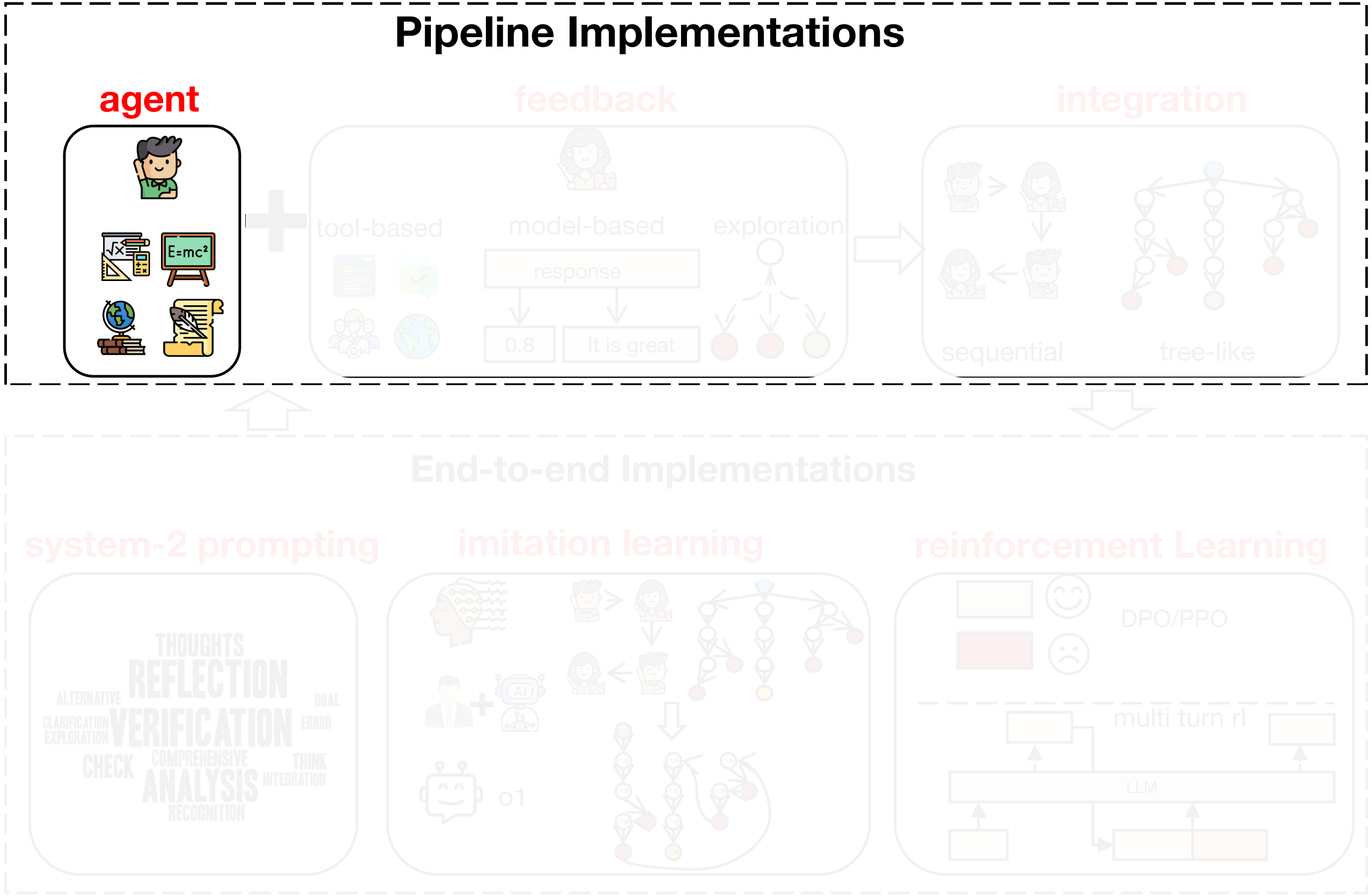
Implementations of Cognition Engineering



Pipeline Implementations

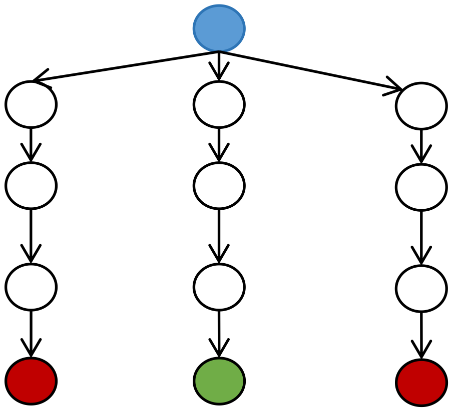


Selection Criterion for Agent



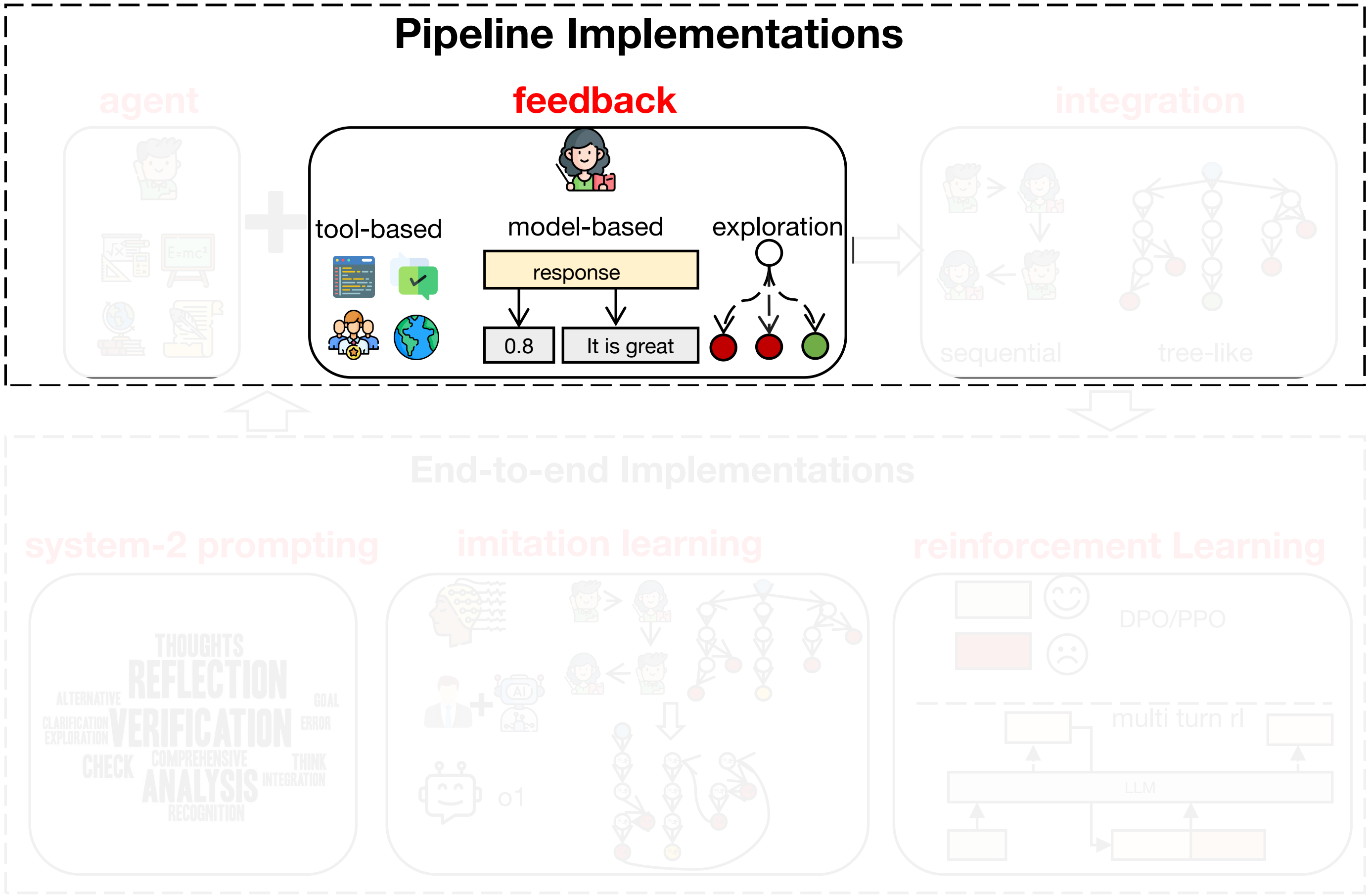
Selection Criterion for Agent

Utilizing Pass@N to estimate the upper bound of performance gain from the pipeline.

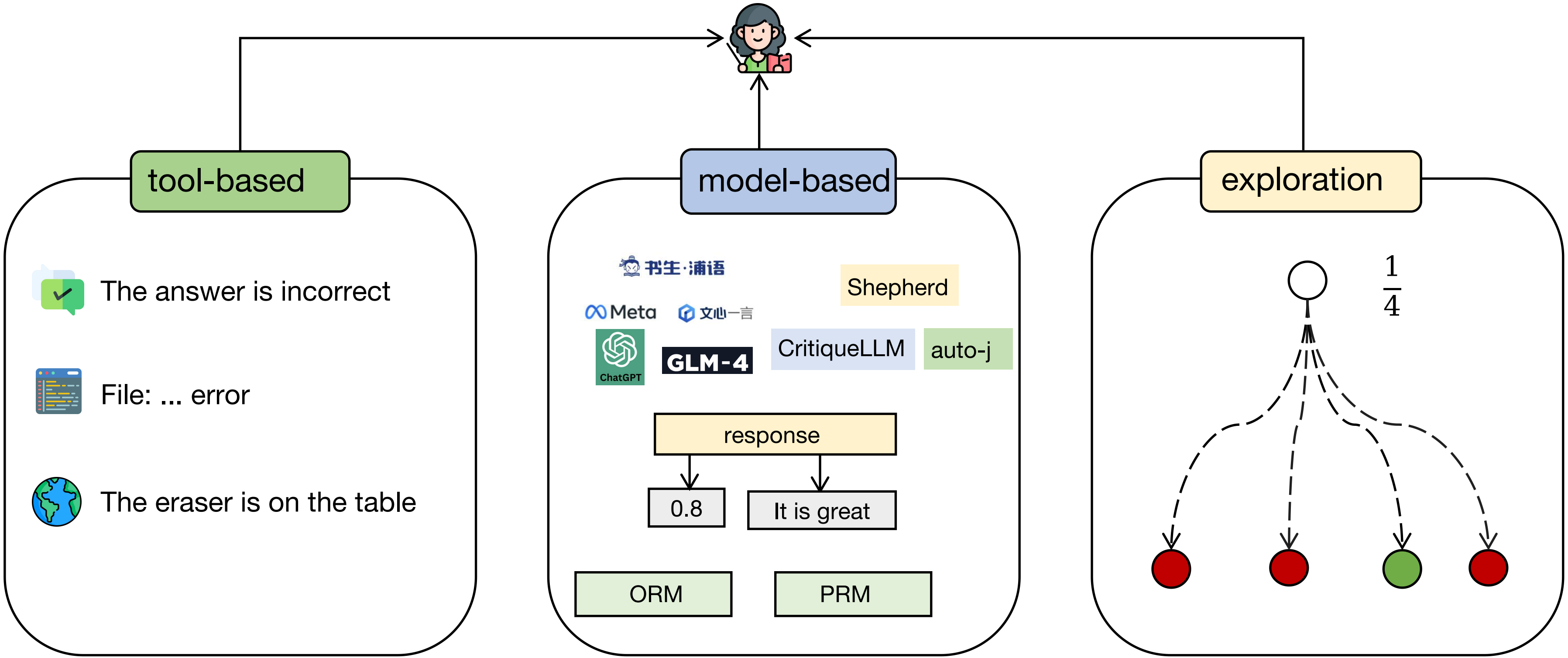


Pass@N

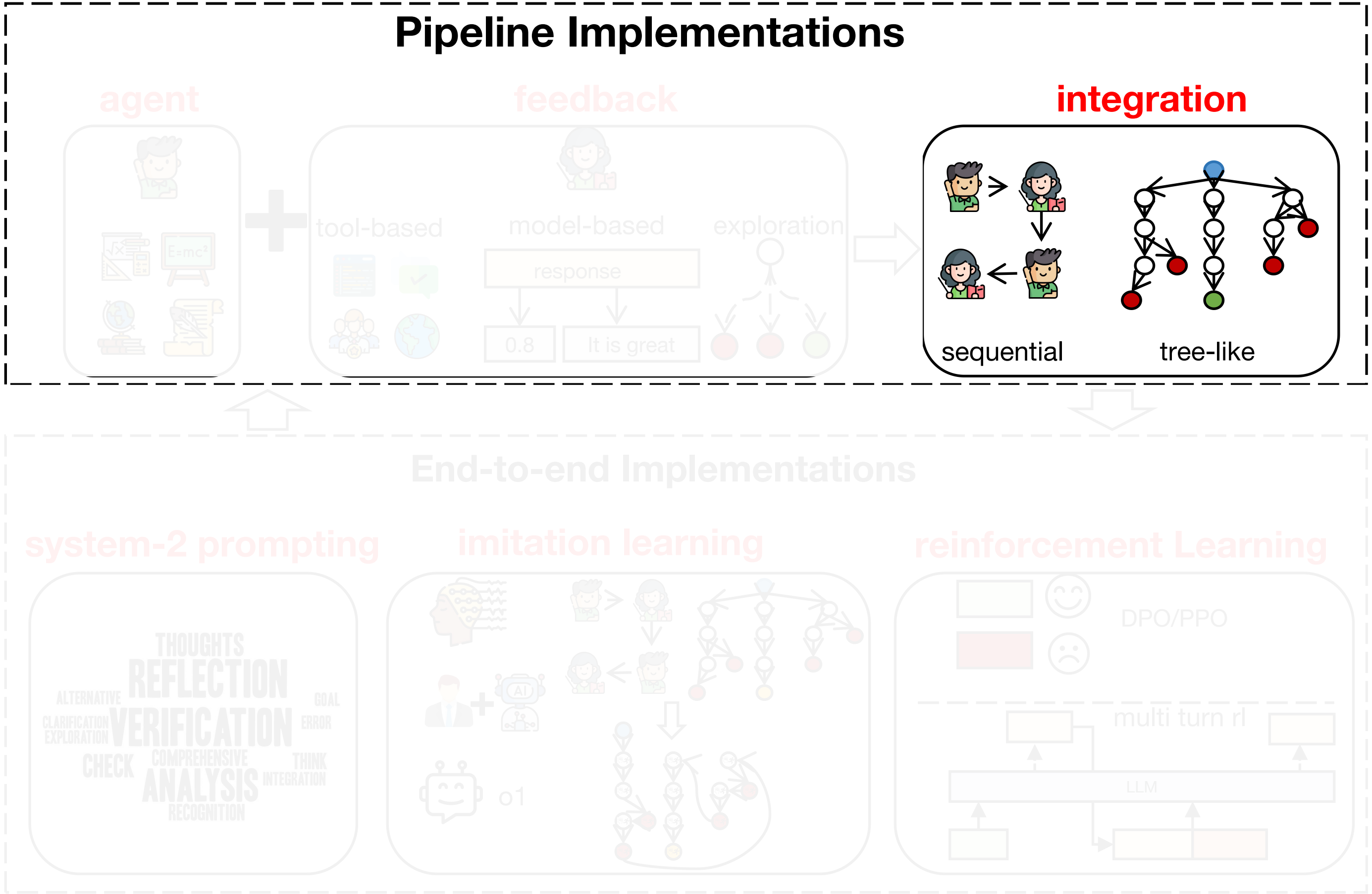
Design Consideration for Feedback Types



Design Consideration for Feedback Types

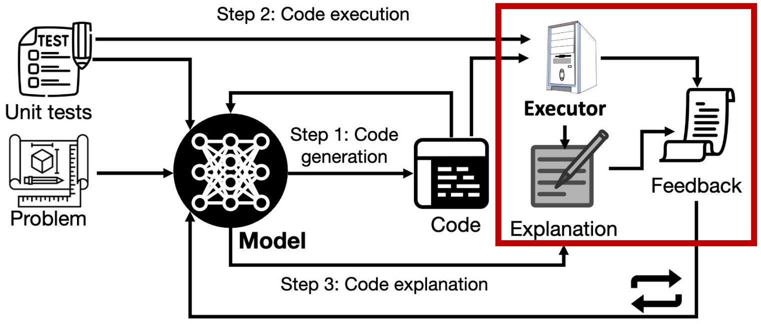


Design Consideration for Integration Methods

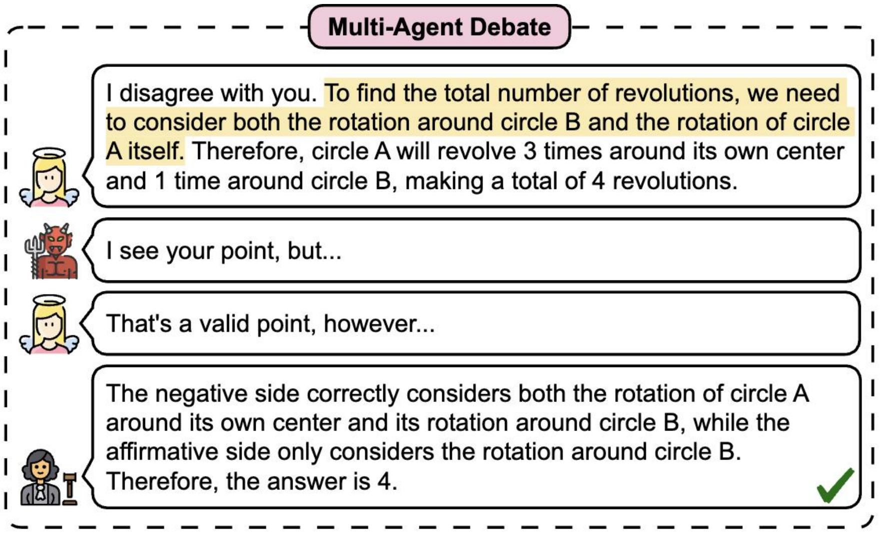


Design Consideration for Integration Methods

sequential integration: $(q, r_1, f_1, r_2, f_2, ...)$



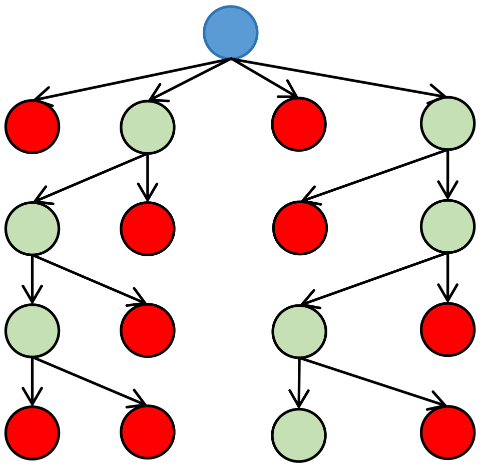
Teaching Large Language Models to Self-Debug, in ICLR 2024



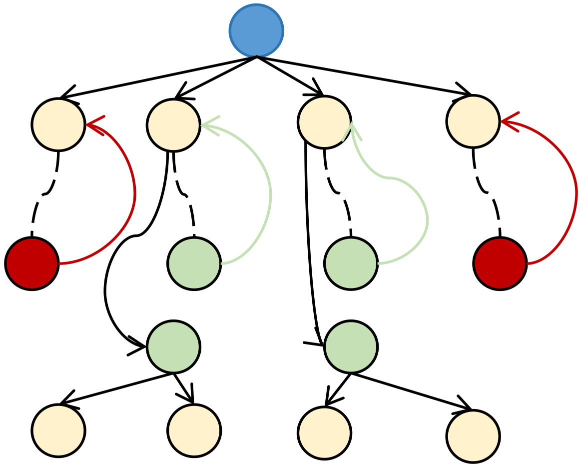
Encouraging Divergent Thinking in Large Language Models through Multi-Agent Debate, in EMNLP 2024

Design Consideration for Integration Methods

tree-like integration: search along the tree guided by reward

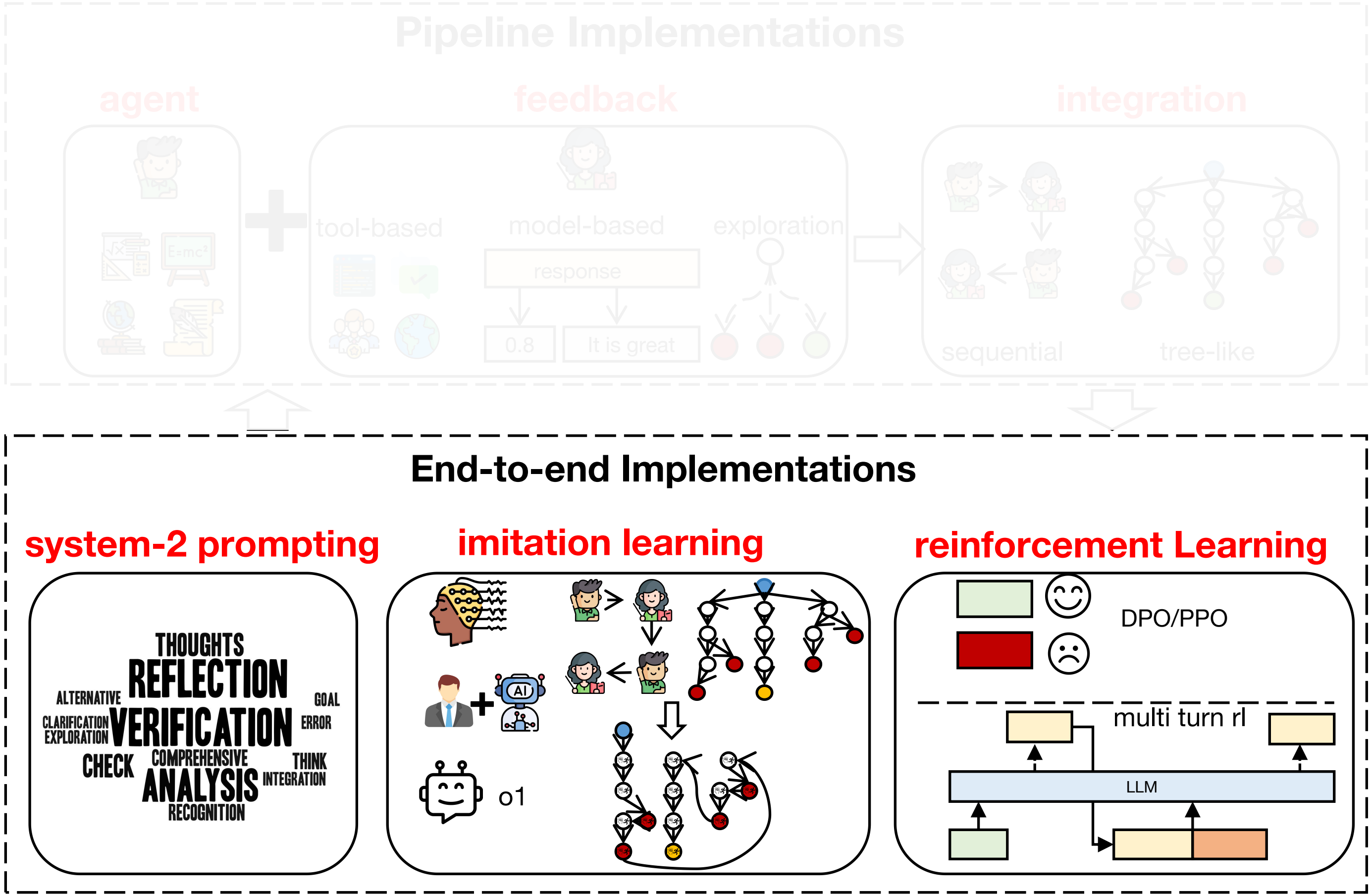


Beam Search

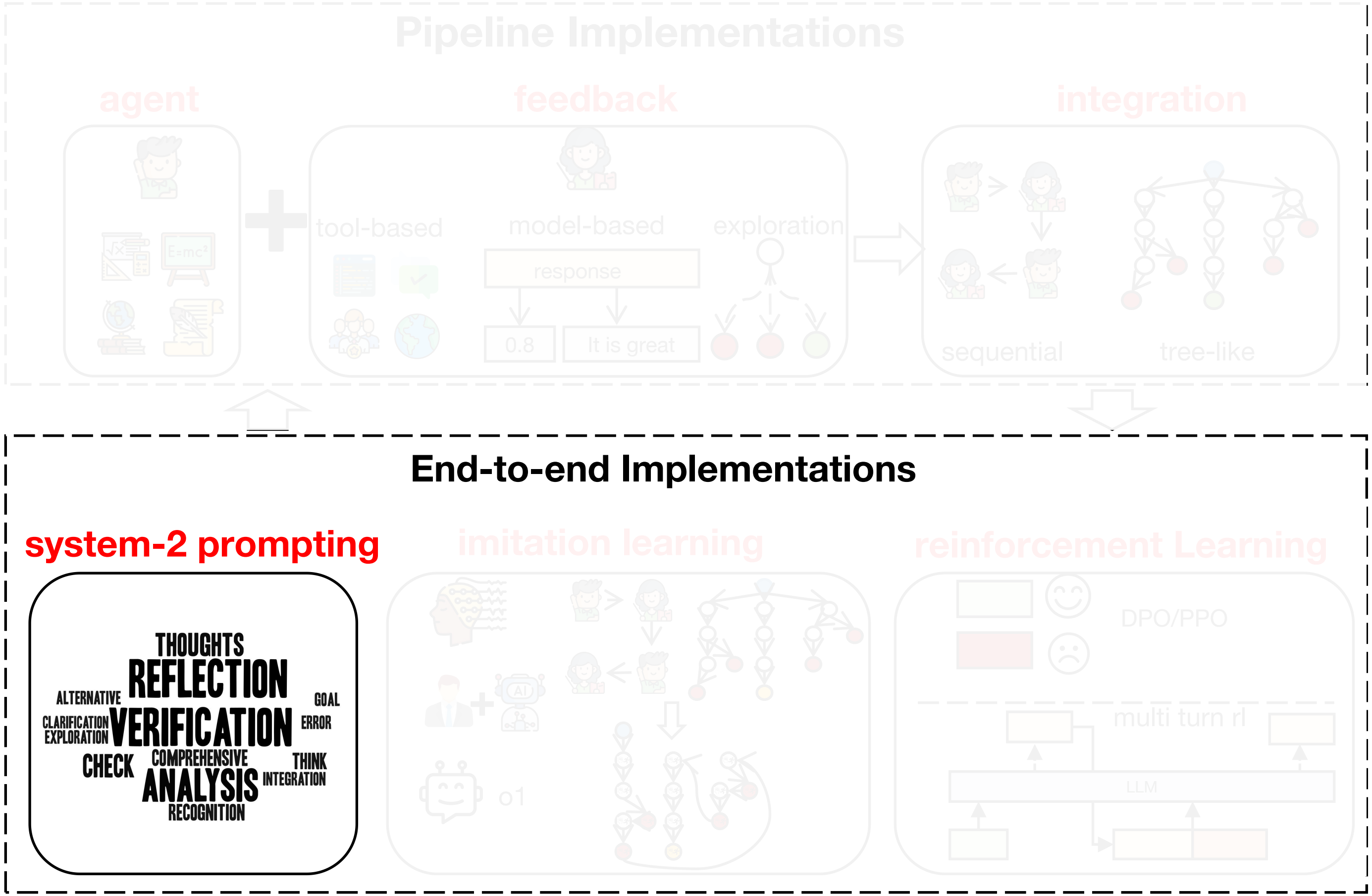


Lookahead Search

End-to-end Implementations



System-2 prompting



Prompting models for System 2 thinking

Pros: low cost
Cons: no empirical results for performance gain

Thinking Claude

Let Claude think comprehensively before responding!

> **A super quick reminder:** Thinking claude **is not aimed for benchmarks or huge leaps in math or something**, since those are pre-determined by the base model (new Claude-3.5 Sonnet). I only want to explore how further we could reach with Claude's "deep mindset". That said, when using it in your daily tasks, you will find Claude's inner monolog (thinking process) very very fun and interesting.

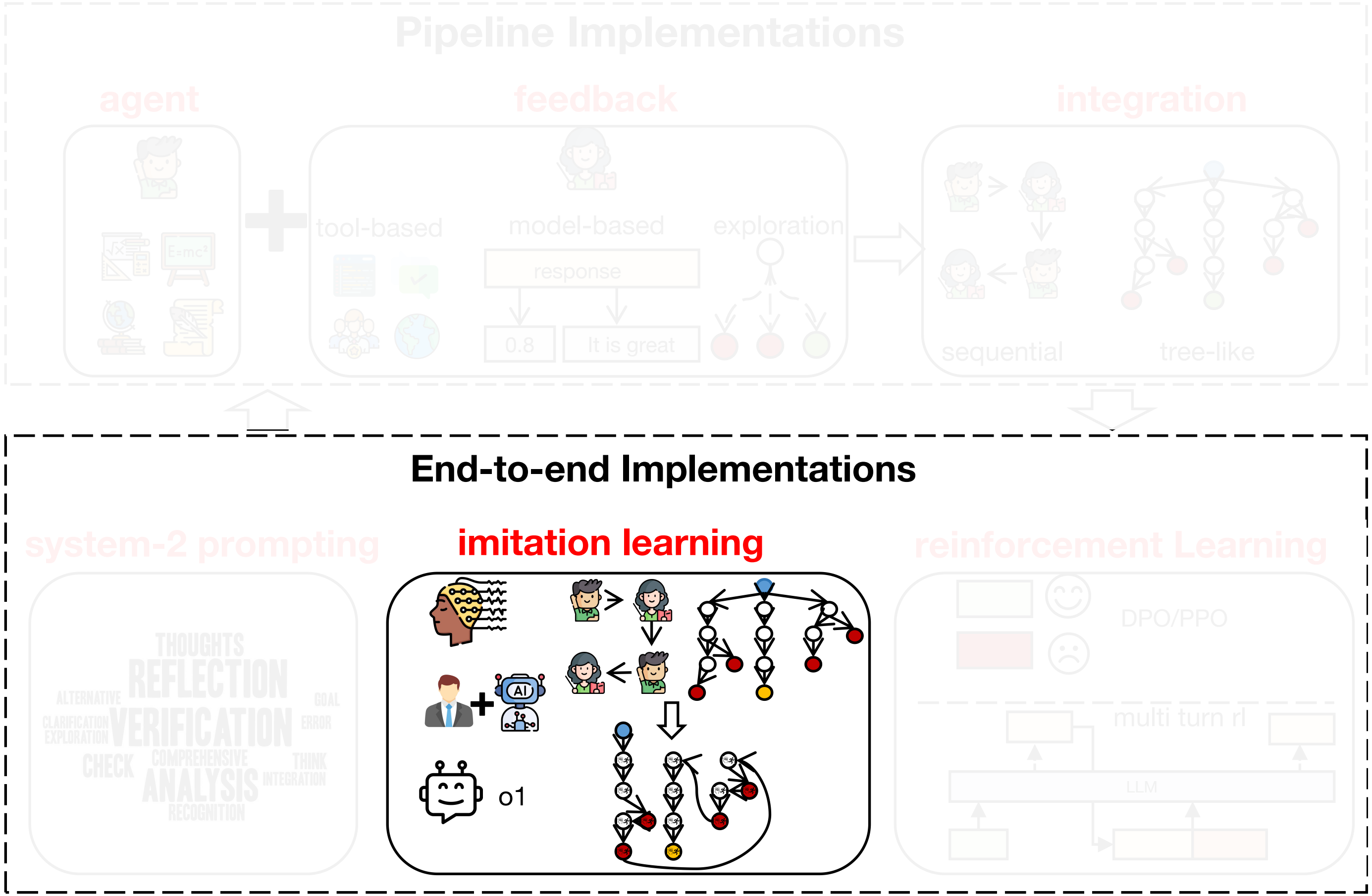
https://github.com/richards199999/Thinking-Claude/tree/main

g1: Using Llama-3.1 70b on Groq to create o1-like reasoning chains

g1_demo.1.mp4

https://github.com/bklieger-groq/g1

Imitation Learning

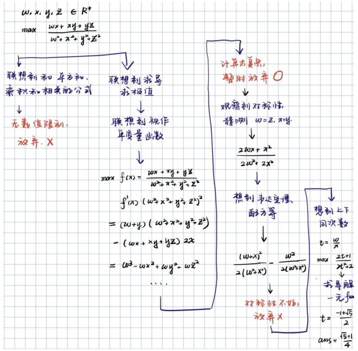


Constructing Long Thoughts

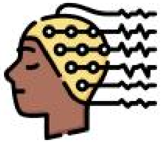
Human-involved methods

Pros: high quality

Cons: high cost

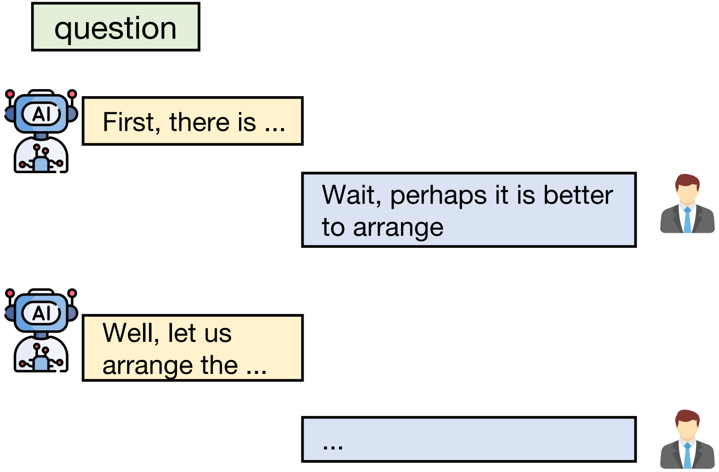


think aloud



convert human EEG signals into explicit language

Human annotations



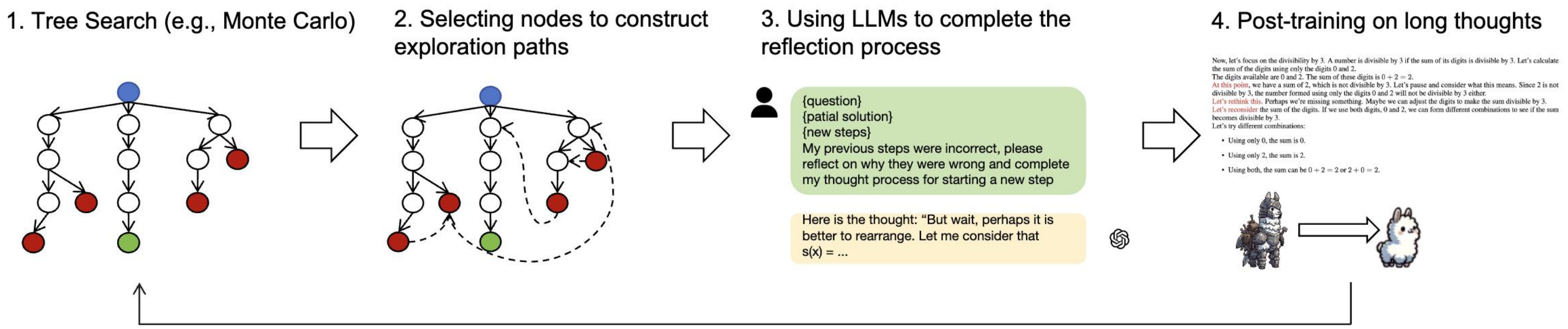
Human-AI involved

Constructing Long Thoughts

Gathering the trajectory of the system implementations

Pros: scaling

Cons: The alignment between the action from the agent and feedback signal is challenged.



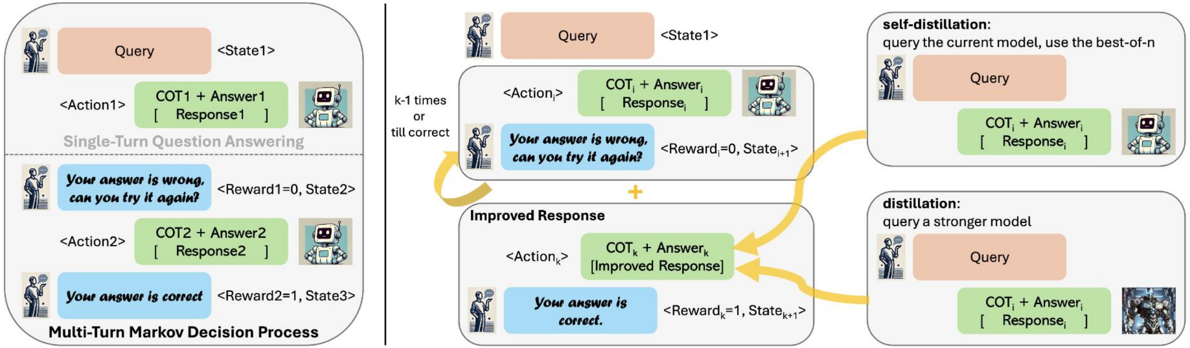
o1 Replication journey: A strategic Progress Report--Part 1, in arxiv 2024

Constructing Long Thoughts

Gathering the trajectory of the system implementations

Pros: scaling

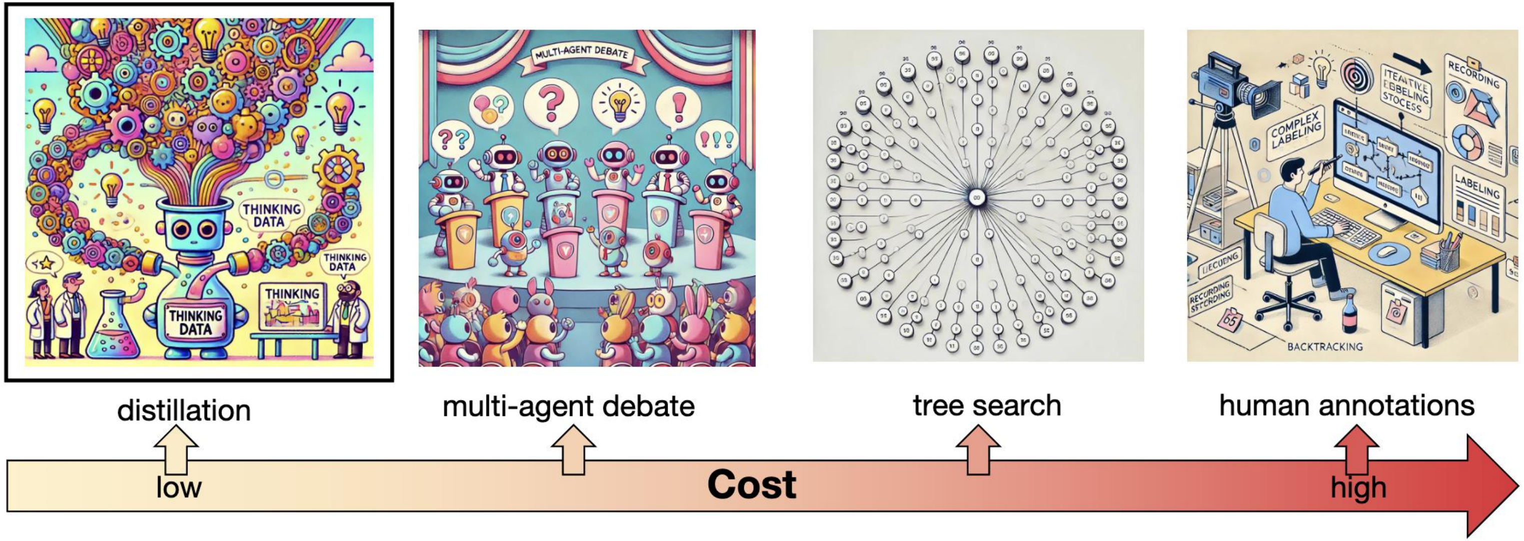
Cons: The alignment between the action from the agent and feedback signal is challenged.



Recursive Introspection: Teaching Language Model Agents How to Self-Improve, in arxiv 2024

Constructing Long Thoughts

Distillation from models like o1
Pros: low cost and guarantees performance
Cons: The upper bound of the models is o1



O1 Replication Journey -- Part 2: Surpassing O1-preview through Simple Distillation, Big Progress or Bitter Lesson?, in arxiv 2024

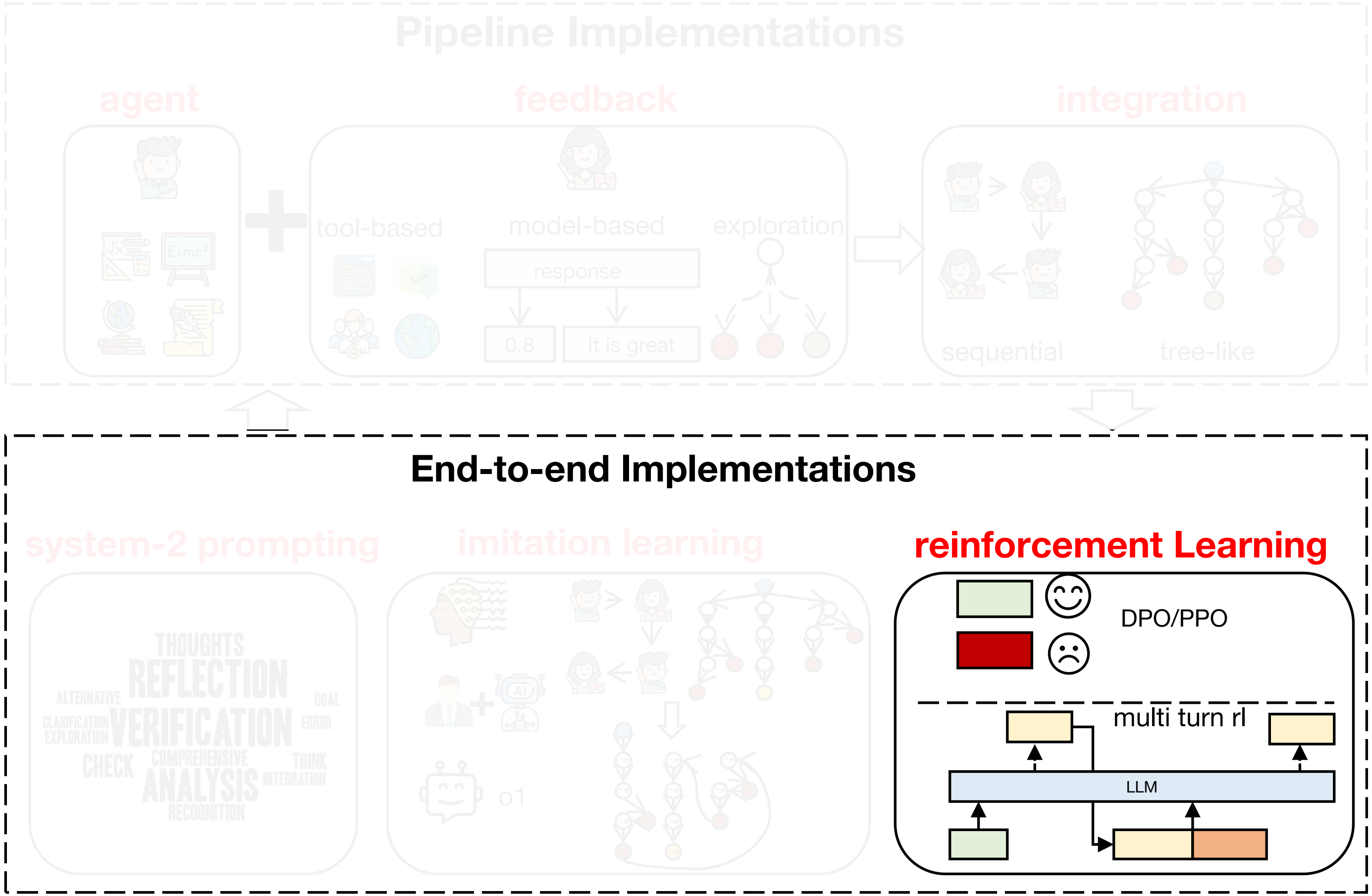
Constructing Long Thoughts

Distillation from models like o1
Pros: low cost and guarantees performance
Cons: The upper bound of the models is o1

Model	AIME(2024)		MATH500	
	Accuracy	# Average Token	Accuracy	# Average Token
Proprietary				
o1-preview	12/30	9083	85.5	1501
o1-mini	21/30	9903	90.0	944
Parameter Size: 72B				
Ours-72B	13/30	8016	87.2	2235

O1 Replication Journey -- Part 2: Surpassing O1-preview through Simple Distillation, Big Progress or Bitter Lesson?, in arxiv 2024

Reinforcement Learning



Reinforcement Learning

CoT

Define the sequence (θ_n) by $\theta_0 = \arccos\frac{5}{13}$ and $\theta_n = 2\theta_{n-1}$. Then $\cos\theta_0 = \frac{5}{13}$ and $\cos\theta_n = \cos(2\theta_{n-1}) = 2\cos^2\theta_{n-1} - 1$. Since the sequences (a_n) and $(\cos\theta_n)$ have the same initial term, and the same recursion, they coincide. We have that $\sin^2\theta_0 = 1 - \cos^2\theta_0 = \frac{144}{169}$. Since θ_0 is acute, $\sin\theta_0 = \frac{12}{13}$. Now, $a_0 a_1 \ldots a_{n-1} = \cos\theta_0 \cos\theta_1 \ldots \cos\theta_{n-1} = \cos\theta_0 \cos 2\theta_0 \ldots \cos 2^{n-1}\theta_0$. Multiplying both sides by $\sin\theta_0 = \frac{12}{13}$, we get $\frac{12}{13} a_0 a_1 \ldots a_{n-1} = \sin\theta_0 \cos\theta_0 \cos 2\theta_0 \cos 4\theta_0 \ldots \cos 2^{n-1}\theta_0 = \frac{1}{2}\sin 2\theta_0 \cos 2\theta_0 \cos 4\theta_0 \ldots \cos 2^{n-1}\theta_0 = \frac{1}{4}\sin 4\theta_0 \ldots \cos 2^{n-1}\theta_0 = \ldots = \frac{1}{2^n}\sin 2^n\theta_0$. Hence, $|a_0 a_1 \ldots a_{n-1}| = \frac{1}{2^n}\cdot\frac{13}{12}|\sin 2^n\theta_0| \le \frac{1}{2^n}\cdot\frac{13}{12}$. This tells us $c \le \frac{13}{12}$. We can compute that $a_1 = 2a_0^2 - 1 = 2\left(\frac{5}{13}\right)^2 - 1 = -\frac{119}{169}$, so $\frac{5}{13}\cdot\frac{119}{169} \le \frac{c}{4}$. Then $c \ge \frac{2380}{2197}$. The bound $\frac{2380}{2197} \le c \le \frac{13}{12}$ tells us that the integer closest to $100c$ is $\boxed{108}$.

Cognition Engineering

We have an infinite real number sequence $a_0, a_1, a_2, \ldots$, where the initial term $a_0 = \frac{5}{13}$. The recurrence relation is: $a_n = 2a_{n-1}^2 - 1$ for all $n \ge 1$. Our goal is to find the smallest constant c such that for all n, the inequality $|a_0 a_1 \cdots a_{n-1}| \le \frac{c}{2^n}$ holds true. Ultimately, we need to calculate $100c$ and round it to the nearest integer.

First, we have the recurrence formula: $a_n = 2a_{n-1}^2 - 1$. The initial condition is $a_0 = \frac{5}{13}$. To better understand this recurrence relation, we can try calculating the first few terms: $a_1 = 2a_0^2 - 1 = 2\left(\frac{5}{13}\right)^2 - 1 = 2\times\frac{25}{169} - 1 = \frac{50}{169} - 1 = \frac{50-169}{169} = \frac{-119}{169}$.

$a_2 = 2a_1^2 - 1 = 2\left(\frac{-119}{169}\right)^2 - 1 = 2\times\frac{14161}{28561} - 1 = \frac{28322}{28561} - 1 = \frac{28322-28561}{28561} = \frac{-239}{28561}$. By calculating the first few terms, we can observe the trend of the sequence a_n. However, this does not directly help us solve the inequality problem. Therefore, we need to find a more effective method.

The original inequality is: $|a_0 a_1 \cdots a_{n-1}| \le \frac{c}{2^n}$. To simplify this inequality, we can multiply both sides by 2^n: $|2a_0 \cdot 2a_1 \cdots 2a_{n-1}| \le c$. This way, we have transformed the inequality into a product form involving $2a_i$. However, while this step simplifies the form of the inequality, it does not immediately reveal how to further handle the product of $2a_i$. Thus, we need to explore other methods to address this recurrence relation.

Observing the recurrence formula: $a_n = 2a_{n-1}^2 - 1$. This resembles the double-angle formula for the cosine function: $\cos(2\theta) = 2\cos^2\theta - 1$. This similarity leads us to the idea that perhaps we can relate the sequence a_n to the cosine function. Suppose each term a_n can be represented as the cosine of a certain angle. Specifically, let: $a_n = \cos\theta_n$. Thus, the recurrence formula becomes: $\cos\theta_n = 2\cos^2\theta_{n-1} - 1$. According to the double-angle formula, we know: $\cos(2\theta) = 2\cos^2\theta - 1$. Therefore, we can deduce: $\cos\theta_n = \cos(2\theta_{n-1})$. This implies: $\theta_n = 2\theta_{n-1}$ or $\theta_n = -2\theta_{n-1} + 2k\pi$, but here we take the positive angle. Thus, the relationship between the angles is: $\theta_n = 2\theta_{n-1}$. By iterating further, we obtain: $\theta_n = 2^n\theta_0$ where θ_0 is the initial angle corresponding to a_0. Based on our definition: $a_0 = \cos\theta_0 = \frac{5}{13}$.

To find θ_0, we need to calculate: $\theta_0 = \arccos\left(\frac{5}{13}\right)$. Additionally, we can compute:

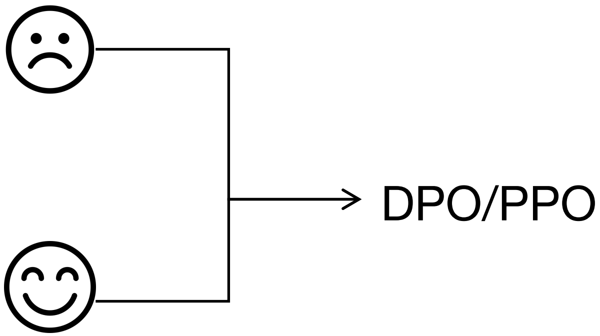
$\sin\theta_0 = \sqrt{1-\cos^2\theta_0} = \sqrt{1-\left(\frac{5}{13}\right)^2} = \sqrt{1-\frac{25}{169}} = \sqrt{\frac{144}{169}} = \frac{12}{13}$. Thus, we have $\sin\theta_0 = \frac{12}{13}$. Returning to our inequality:

$|a_0 a_1 \cdots a_{n-1}| \le \frac{c}{2^n}$. Based on our definition $a_i = \cos\theta_i$, we have: $a_0 a_1 \cdots a_{n-1} = \prod_{i=0}^{n-1}\cos\theta_i$. Since $\theta_i = 2^i\theta_0$, we can express the product as: $\prod_{i=0}^{n-1}\cos(2^i\theta_0)$. Using the product formula for cosines, we know: $\prod_{i=0}^{n-1}\cos(2^i\theta_0) = \frac{\sin(2^n\theta_0)}{2^n\sin\theta_0}$. Therefore, the inequality can be transformed to: $\left|\frac{\sin(2^n\theta_0)}{2^n\sin\theta_0}\right| \le \frac{c}{2^n}$. Multiplying both sides by $2^n\sin\theta_0$ (noting that $\sin\theta_0 > 0$), we obtain: $\left|\sin(2^n\theta_0)\right| \le c\cdot\sin\theta_0$.

Substituting the known $\sin\theta_0 = \frac{12}{13}$, we get: $\left|\sin(2^n\theta_0)\right| \le c\cdot\frac{12}{13}$. To ensure this inequality holds for all n, we require:

$c \ge \frac{|\sin(2^n\theta_0)|}{\frac{12}{13}} = \frac{13}{12}|\sin(2^n\theta_0)|$. Since $|\sin(2^n\theta_0)| \le 1$ for all n, the smallest c must satisfy: $c \ge \frac{13}{12}$. Therefore, the smallest c is $\frac{13}{12}$. The problem requires us to calculate $100c$ and round it to the nearest integer. Based on our value for c: $100c = 100\times\frac{13}{12} = \frac{1300}{12} = 108.\overline{3}$.

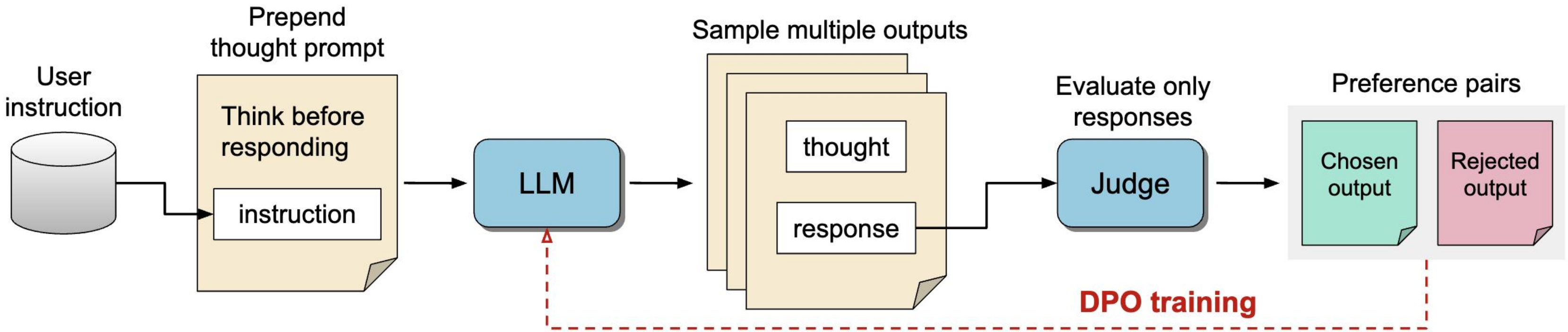
Rounding $108.\overline{3}$ to the nearest integer, we obtain: $100c \approx 108$. Therefore, the final answer is 108.



core: training a reward model for high-quality human thinking process

Reinforcement Learning

- outputs: thought + response
- dpo only in the response



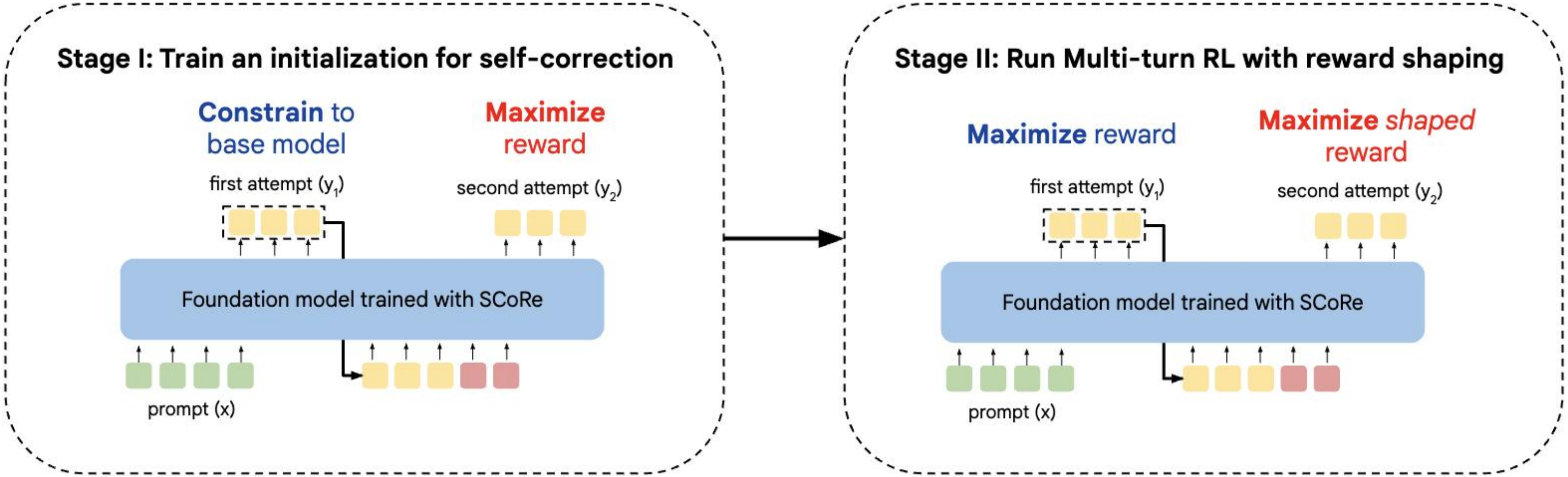
Thinking LLMs: General Instruction Following with Thought Generation, in arxiv 2024

Reinforcement Learning

multi-turn rl

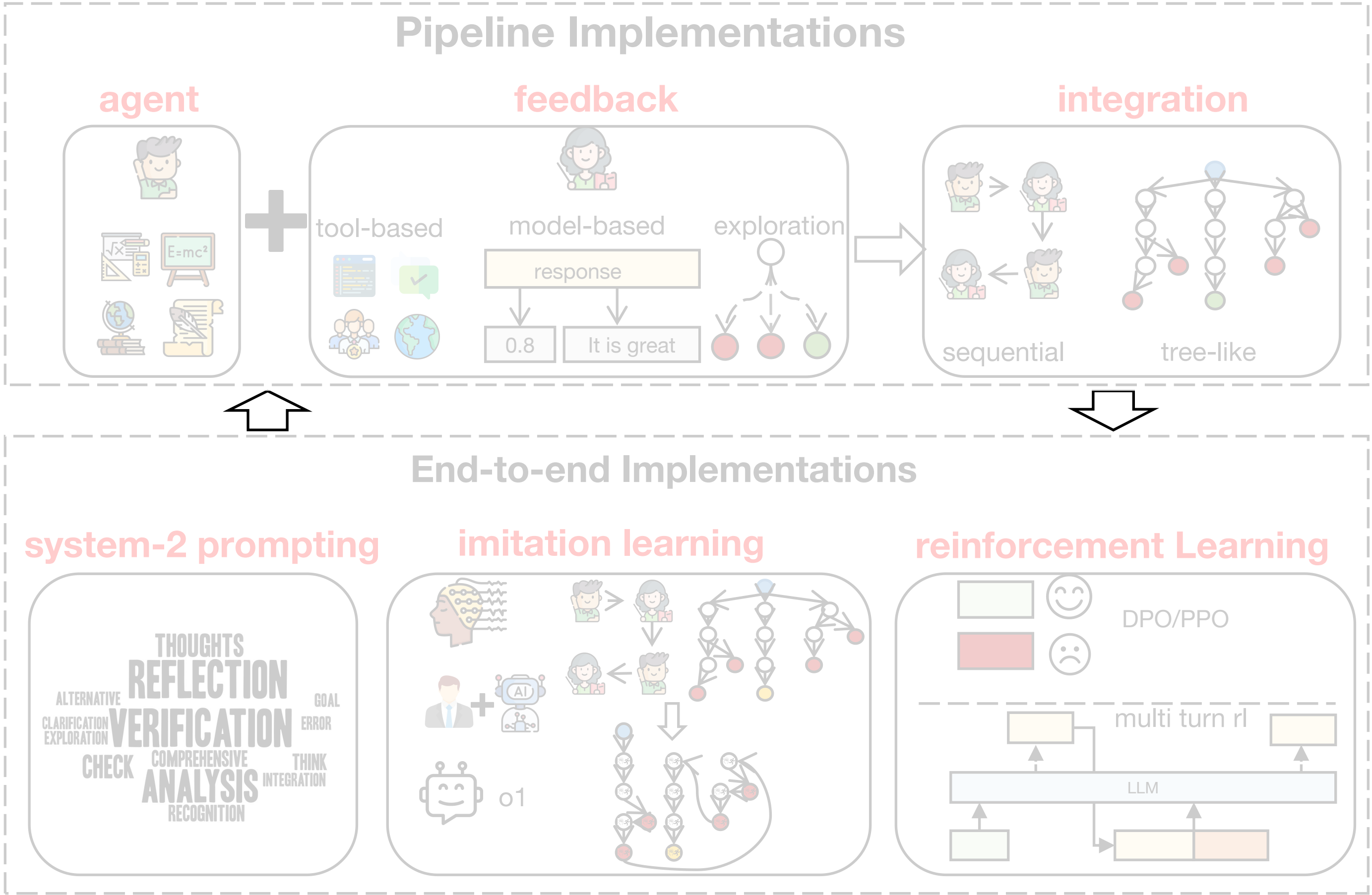
✓ the first attempt is correct

✓ the first attempt is incorrect but the second attempt is correct



Training Language Models to Self-Correct via Reinforcement Learning, in arxiv 2024

Connections Between the Two Implementations



Q&A

- What's the cognition engineering?
  - Teach models to learn from the entire human cognitive process.
- Why now cognition engineering?
  - Level 2 AGI: cognition engineering advances the models to reasoners
  - Level 3-5 AGI: cognition engineering provides the foundation for the next level of AGI
- How to implement the cognition engineering?
  - Pipeline implementation
    - Agent
    - Feedback
    - Integration
  - End-to-end implementation
    - System-2 prompting
    - Imitation learning
    - Reinforcement learning

Our position paper 'Cognition Engineering' will be preprinted soon!