Natural Language Processing

Lecture : Large Reasoning Models

Master Degree in Computer Engineering
University of Padua
Lecturer : Giorgio Satta

Lecture partially based on material originally developed by :
Maarten Grootendorst: A Visual Guide to Reasoning LLMs

Large reasoning models

©Placement Preparation

Maarten Grootendorst, A Visual Guide to Reasoning LLMs

https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-reasoning-llms

Large reasoning models

OpenAI's 5 steps towards artificial general intelligence (AGI)

- conversational AI
- reasoning AI
- autonomous agent AI
- innovating AI
- organizational AI

Large reasoning models

Large reasoning models (LRM) are LLMs empowered with reasoning capabilities.

Examples as for 2025: OpenAI o-3, DeepSeek R1, Google Gemini 2.0 Flash Thinking.

LRMs break down a problem into smaller steps (often called reasoning steps or thought processes) before answering a given question.

LRMs learn to mimic human reasoning through

- long chain-of-thought (CoT)
- reinforcement learning

Large reasoning models

Example :

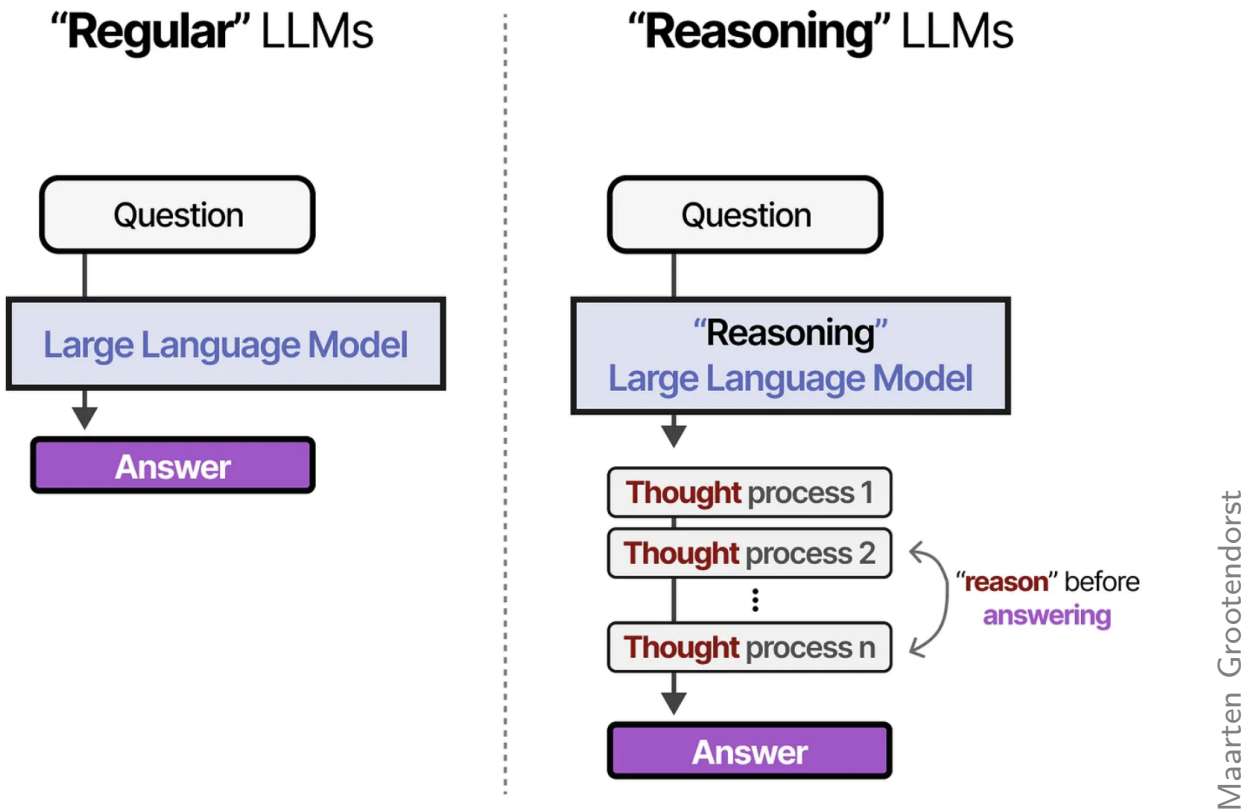
Julia has two sisters and one brother. How many sisters does her brother Martin have?

- Julia has two sisters; that means there are three girls in total (Julia + two more)
- Julia also has one brother, named Martin
- altogether, there are four siblings: three girls and one boy (Martin)
- from Martin's perspective, his sisters are all three of the girls (Julia and her two sisters)
- therefore Martin has three sisters.

M. Mitchell, https://www.science.org/doi/10.1126/science.adw5211

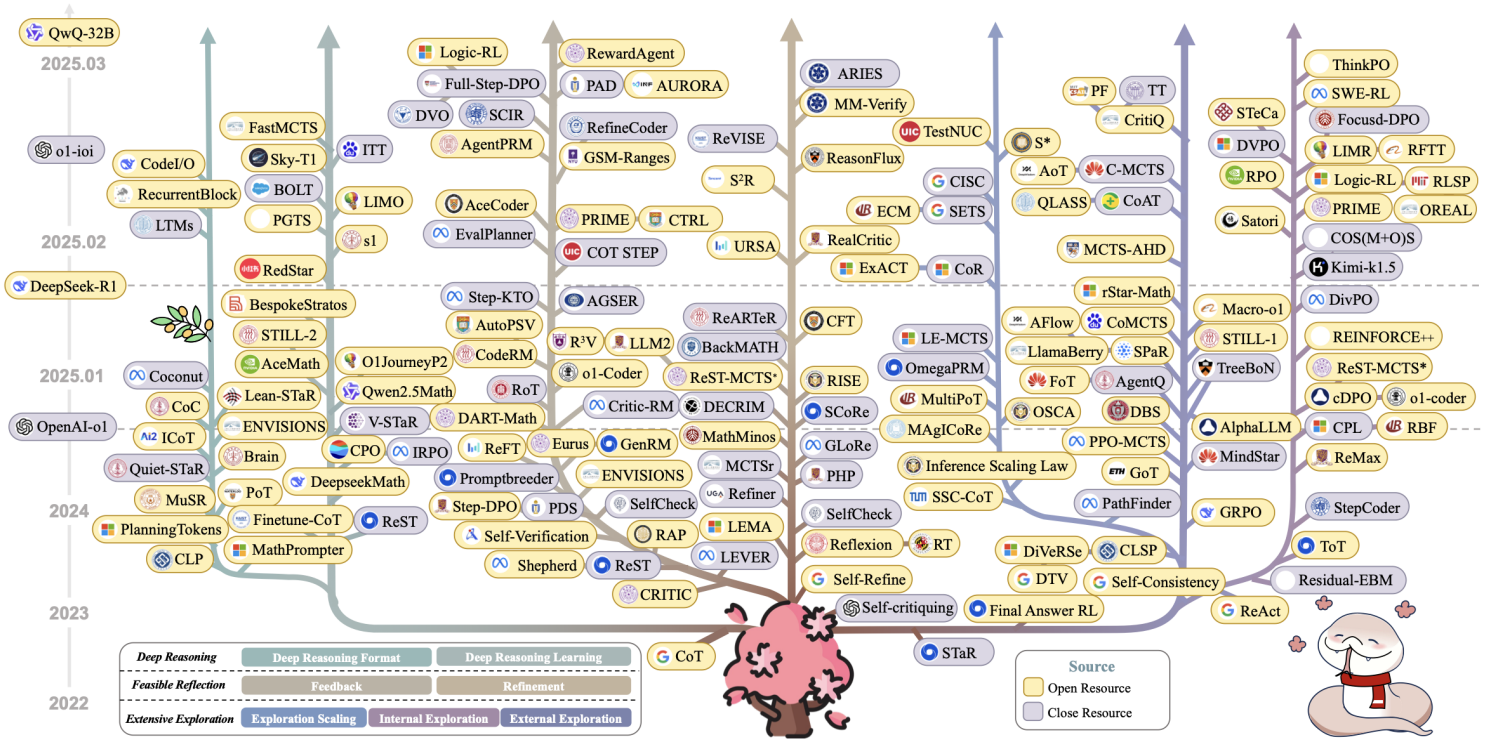
Large reasoning models

LRMs generates entire CoT, which can be really long, and most of which are not revealed to the user.



Large reasoning models

Evolution of reasoning systems based on long CoT.



Chen et al., Towards Reasoning Era

Test-time compute

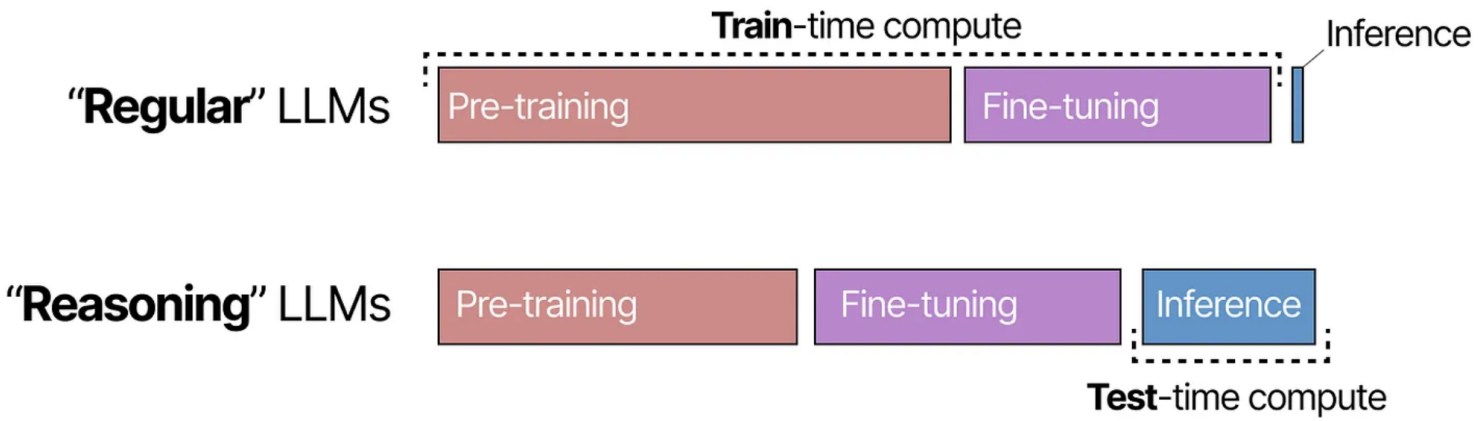
LRMs mark the paradigm shift from scaling **train-time compute** to scaling **test-time compute**.

Old paradigm: the larger your pretraining budget (parameters, tokens, FLOPs) the better the resulting model will be.

New paradigm: allow model to use more time resources during inference and explore large search space, i.e. think longer.

Test-time compute

LLM output the answer and skip any reasoning step. LRM use more tokens to derive their answer, through a systematic 'thinking process'

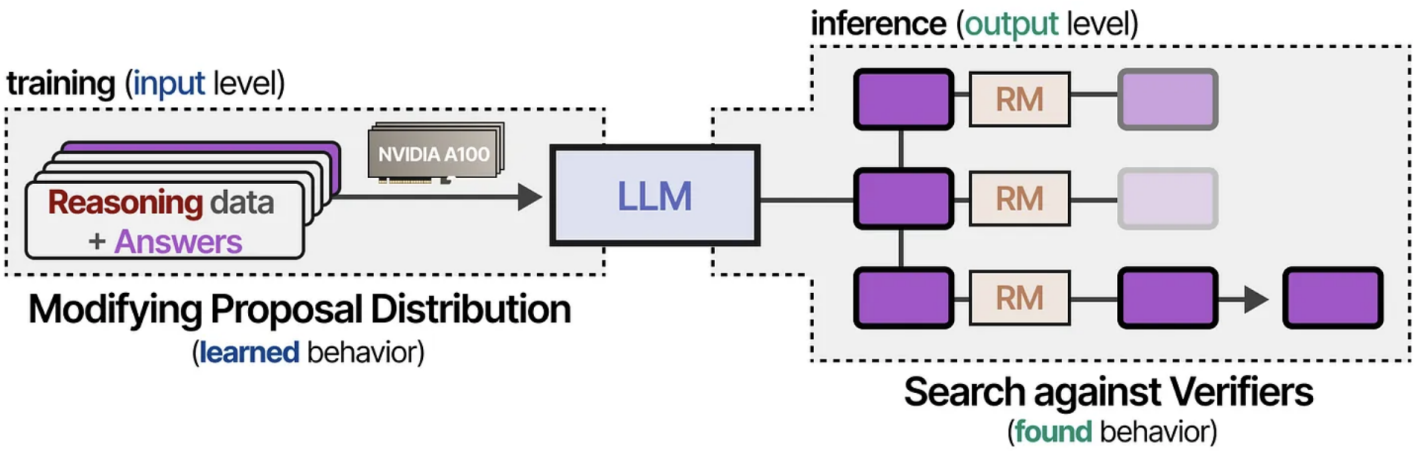


Maarten Grootendorst

Test-time compute

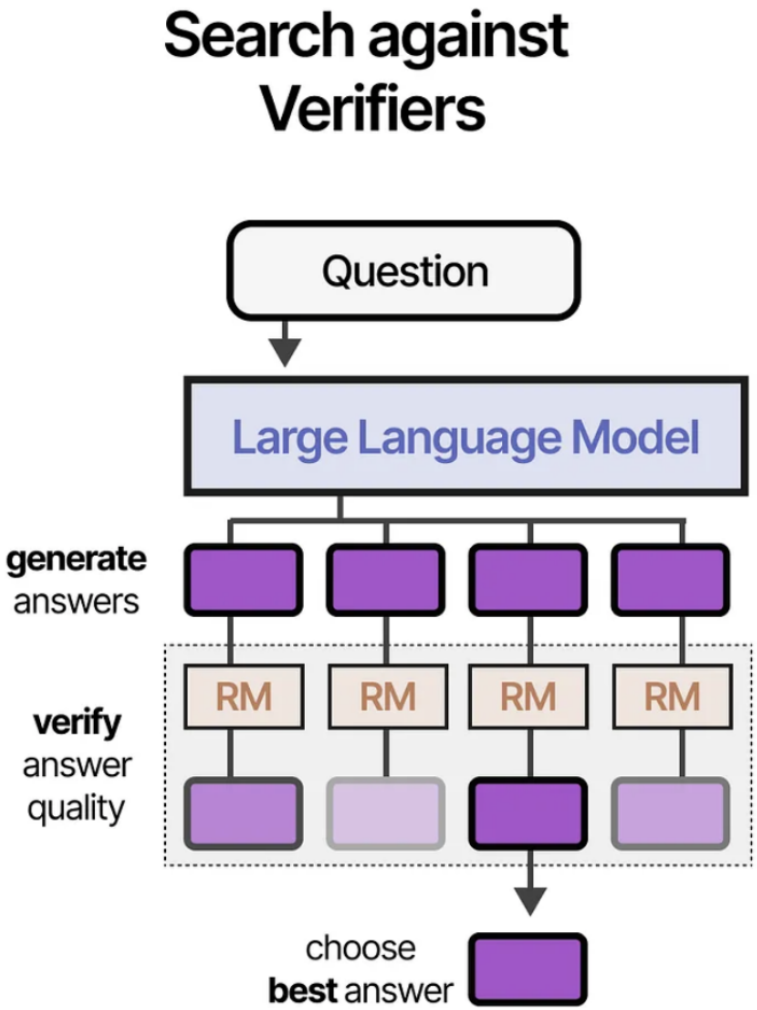
Test-time compute can be roughly put into two categories

- search against verifiers: sampling CoT generations and selecting the best answer (output-focused)
- modifying proposal distribution: the model is trained to create improved reasoning steps (input-focused)



Maarten Grootendorst

Search against verifiers



Search against verifiers

Search against verifiers involves two steps

- multiple samples of reasoning processes and answers are created
- a verifier scores the generated output

The **verifier** is based on a LLM, fine-tuned for judging the quality of the process and acting as a reword model (RM).

Two methodologies for the verifier

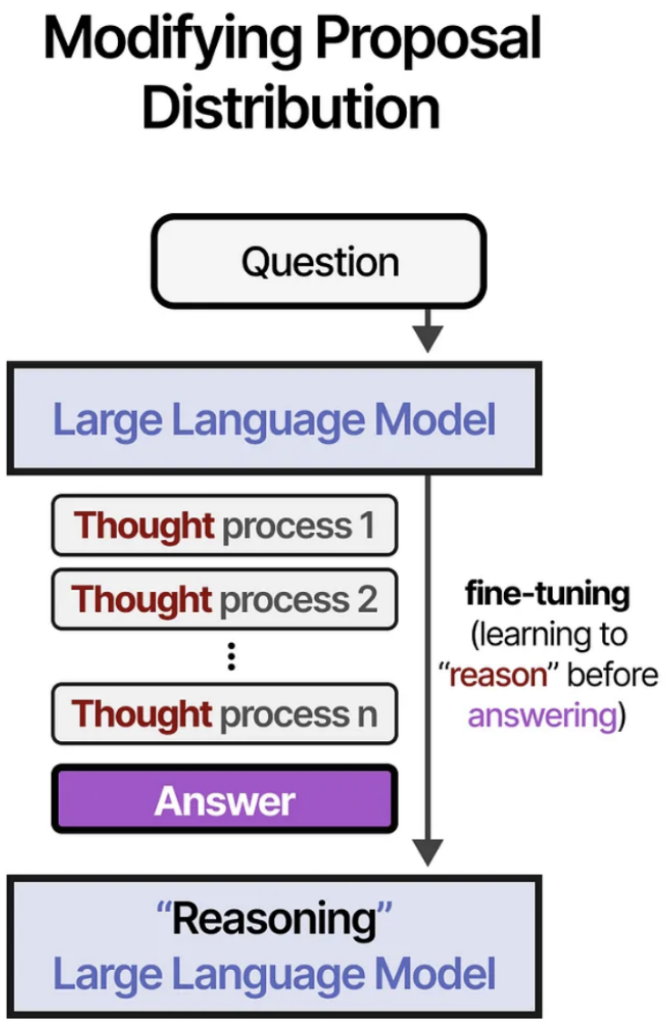
- outcome verifier: only judges the answer
- process verifier: judges the reasoning that leads to the answer

Search against verifiers

Any of the following **search strategies** may be implemented

- majority voting: baseline, no verifier involved
- best-of-N samples: generate N samples and use an outcome or process verifier to judge each answer
- beam search: multiple reasoning steps are sampled and top D best-scoring paths are tracked by a process verifier
- Monte Carlo tree search is a balance between exploration and exploitation
  - expansion of selected node
  - randomly create new nodes until you reach the end (rollout)
  - update parent node scores based on output (backprop)

Modifying proposal distribution

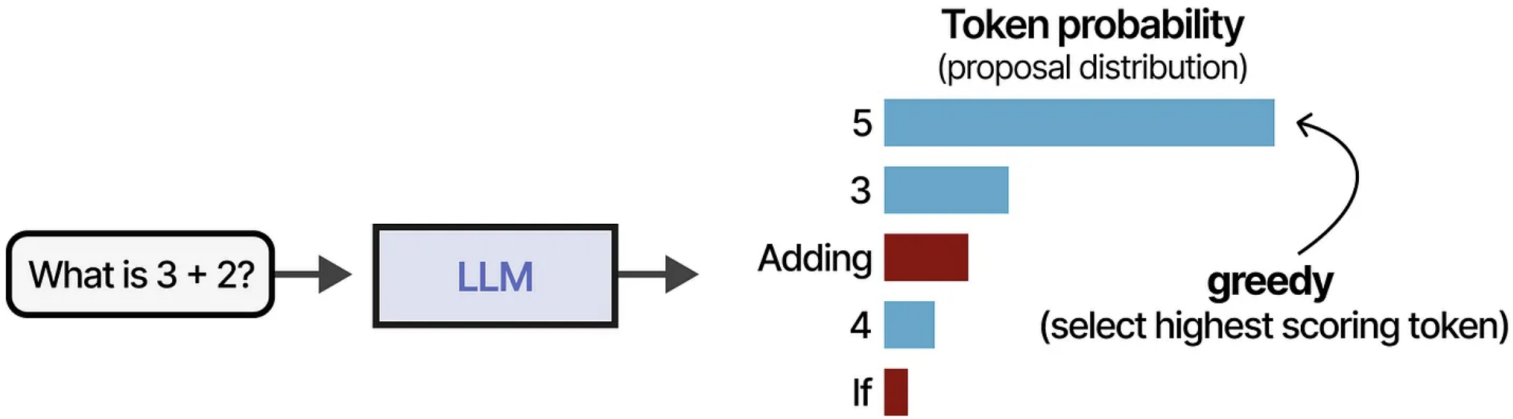


Maarten Grootendorst

Modifying proposal distribution

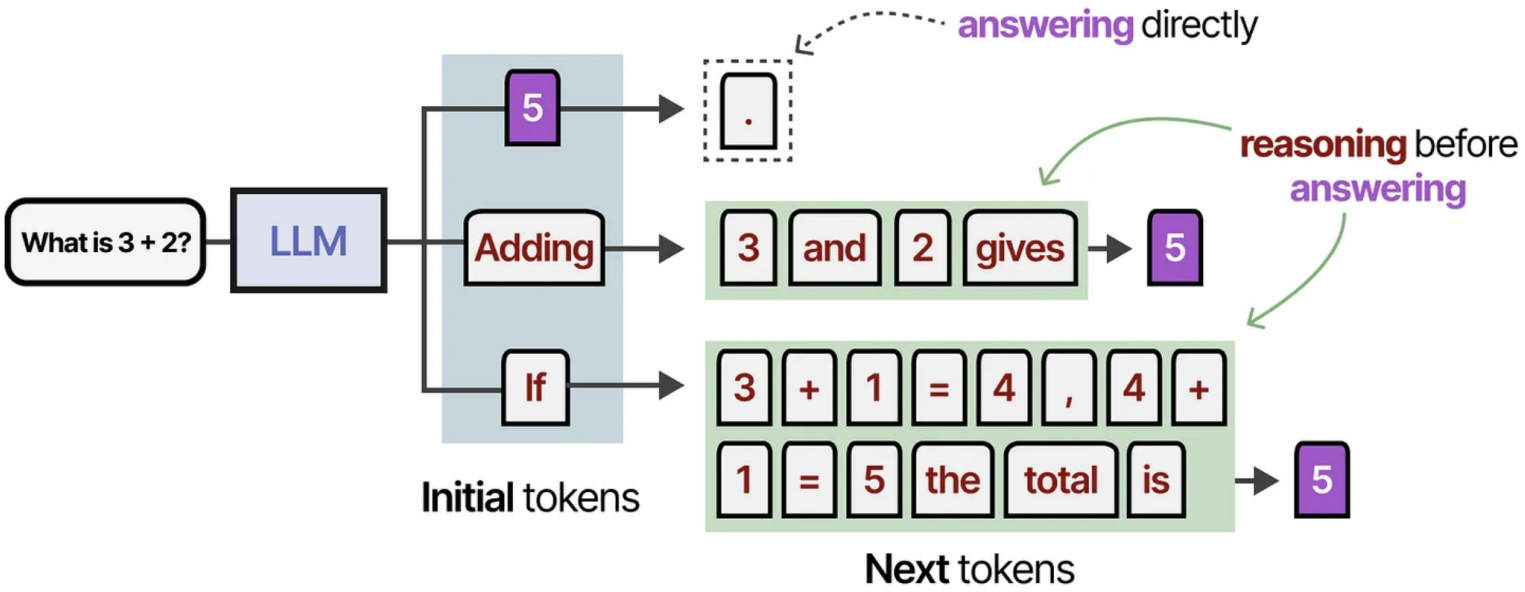
Modifying proposal distribution trains the model to create improved reasoning steps.

We are interested in tokens that are more likely to lead to a reasoning process.



Maarten Grootendorst

Modifying proposal distribution



Maarten Grootendorst

Modifying proposal distribution

Two general methods for modifying the next token distribution.

Use of prompt engineering: provide examples to the model, or use prompt that induce reasoning-like behavior.

Reward the model for generating reasoning steps; this requires

- reasoning dataset
- reinforcement learning algorithms