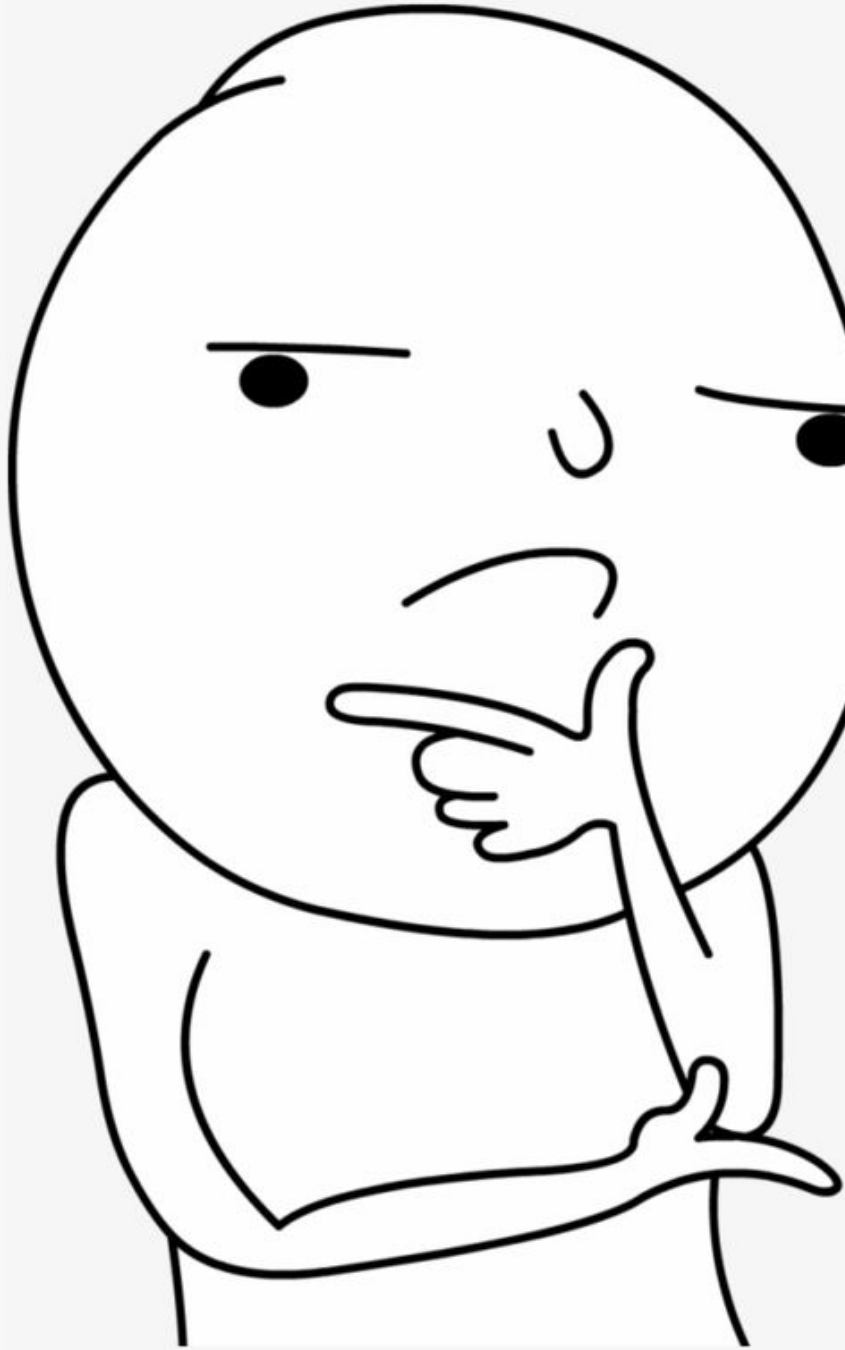
education

LLM Agents

Егор Спирин

1. Заставить модель рассуждать
2. Исправлять ошибки за собой
3. Использовать внешние инструменты
4. Взаимодействие с другими БЯМ



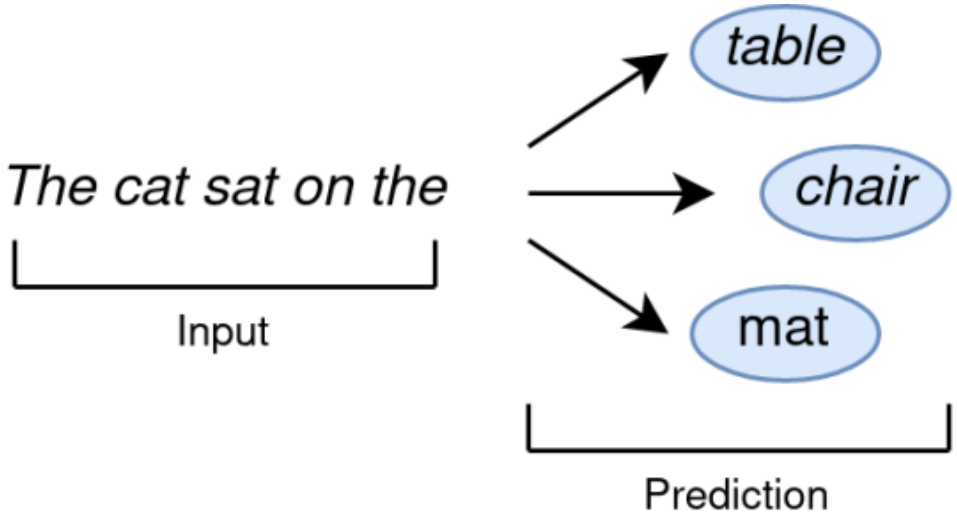
Quick Recap

Языковые модели

Умеем предсказывать следующий токен 🤳

Модель обучается в несколько стадий:

1. Pre-Train — много текста, “впитывание” знаний
2. Instruction Tuning — добавляем инструкции, “пользуемся” знаниями
3. Preference Learning — выравниваем ответы



GPT-2, 3, ...

"Our largest model, GPT-2, is a 1.5B parameter Transformer that achieves state of the art results on 7 out of 8 tested language modeling datasets in a zero-shot setting but still underfits WebText"

Модель обученная на предсказывание следующего слова оказалась SOTA:

👉 POS-tagging
👉 Summarization
👉 Translation
👉 ...

И все это без обучения под задачи против специально обученных моделей 🔥

[1] — Language Models are Unsupervised Multitask Learners, Alec Radford, Jeffrey Wu, Dario Amodei, Ilya Sutskever et al., OpenAI, 2019

Prompt Crafting 

Prompt a.k.a. Затравка

Текстовый запрос или инструкция, подаваемая языковой модели для генерации ответа или выполнения задачи

👉 Управление моделью

👉 Передача знаний

👉 Позволяют раскрыть весь потенциал БЯМ

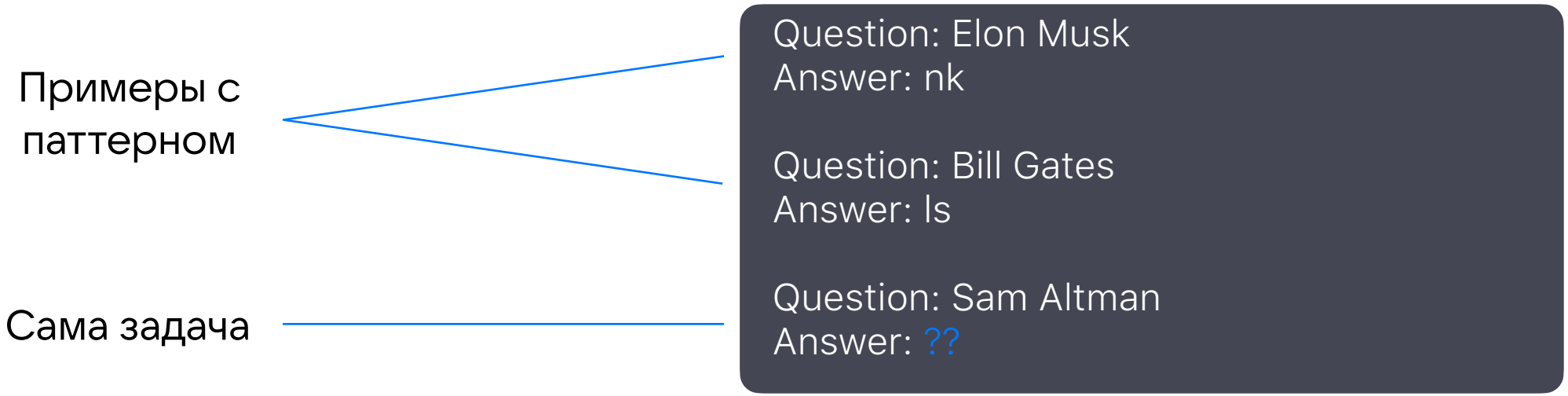
Размер контекста ограничен моделью ⇒ размер промпта также ограничен

Question: how many apples in the bucket?

Answer: there are 52 apples.

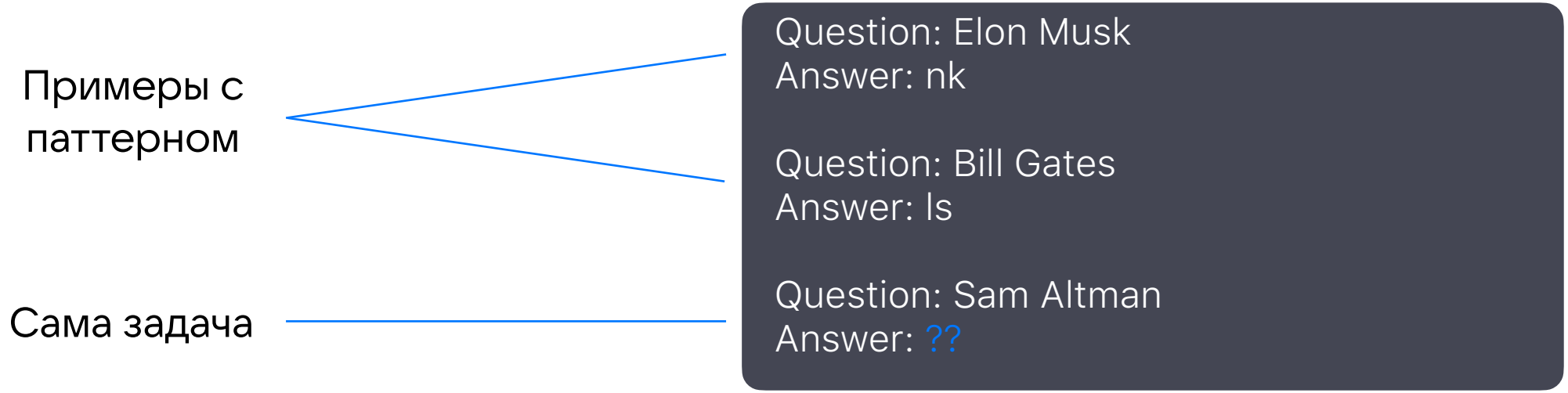
Few-shot

Можем показать модели несколько примеров с ожидаемым поведением



Few-shot

Можем показать модели несколько примеров с ожидаемым поведением



Человек легко ответить — он умеет размышлять (reasoning), в отличие от LLM 🫠

Chain-of-Thought

Давайте добавим шаг с рассуждением!

Question: Elon Musk
Answer: the last letter of "Elon" is "n". The last letter of "Musk" is "k". Concatenating "n", “k" leads to "nk". So the output is "nk".

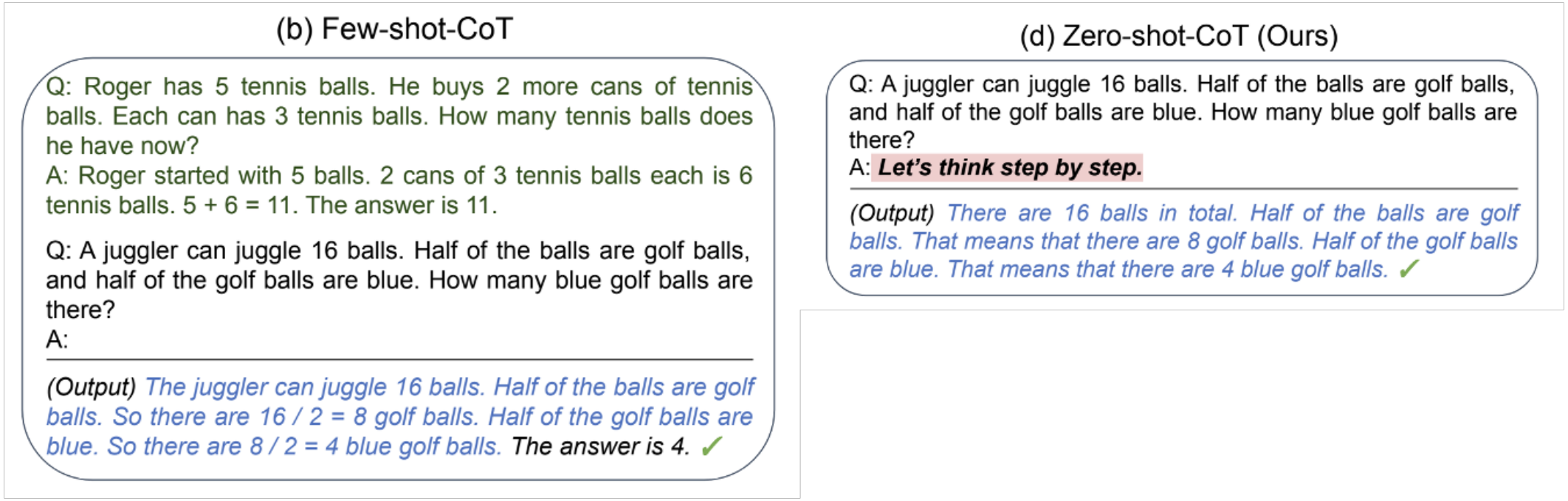
Question: Bill Gates
Answer: the last letter of "Bill" is "l". The last letter of "Gates" is "s". Concatenating "l", "s" leads to "ls". So the output is "ls".

Question: Sam Altman
Answer: the last letter of "Sam" is “m". The last letter of "Altman" is "n". Concatenating "m", "n" leads to "mn". So the output is "mn".

[2] — Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, Wei et al., Google Research, 2023

Zero-Shot CoT

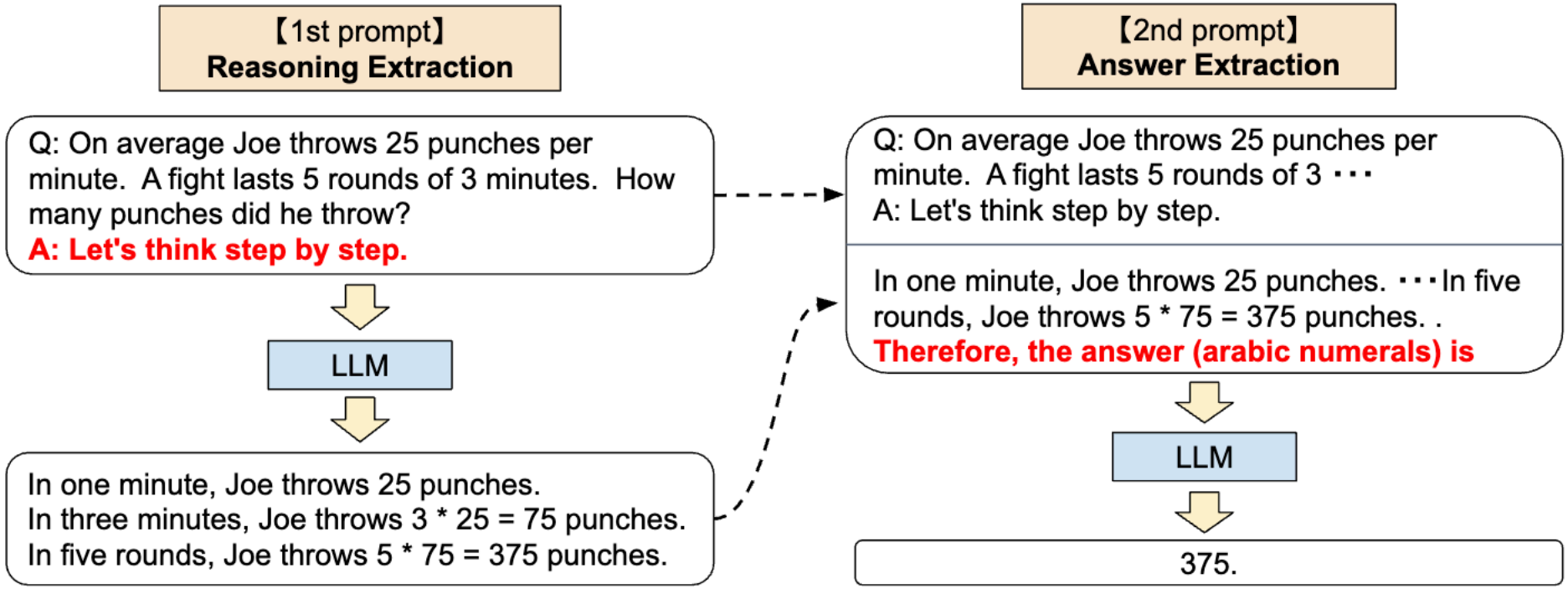
Формируй промпты правильно 👍



[3] — Large Language Models are Zero-Shot Reasoners, Kojima et al., Google Research, 2023

Reason → Answer

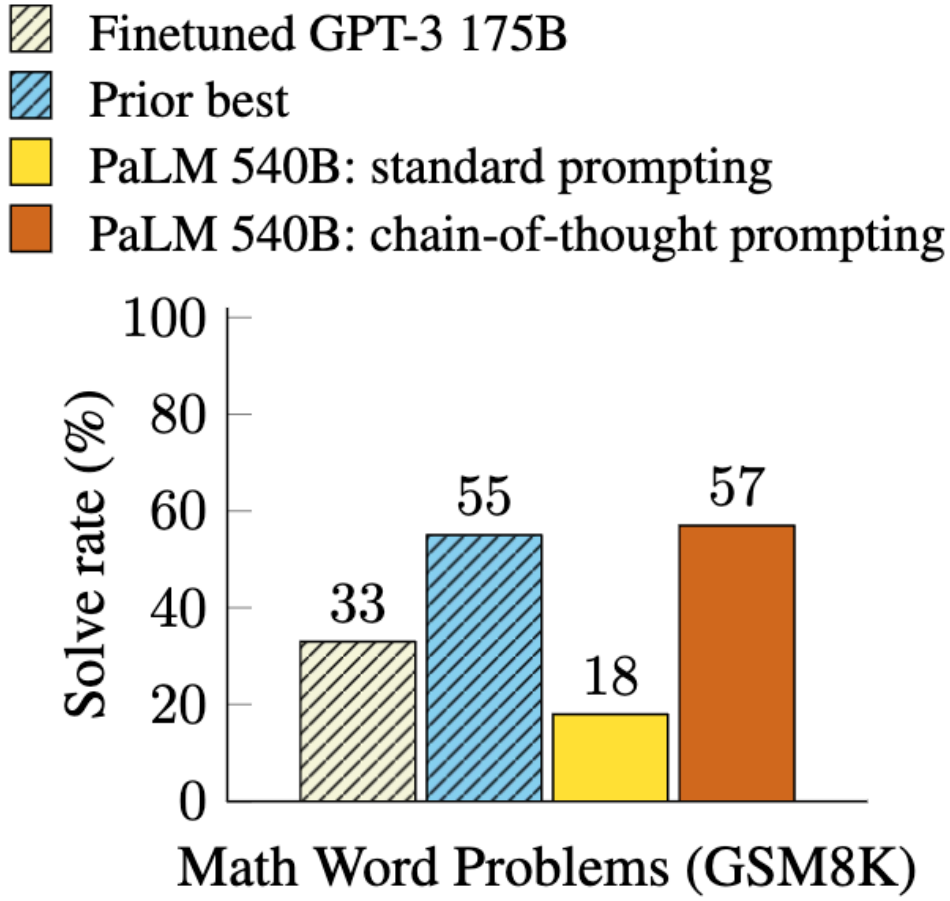
1. Парсить вывод через ключевые слова
2. Использовать структурную генерацию
3. Переиспользовать LLM



Применимость

Особенно хорошо справляется с математическими задачами!

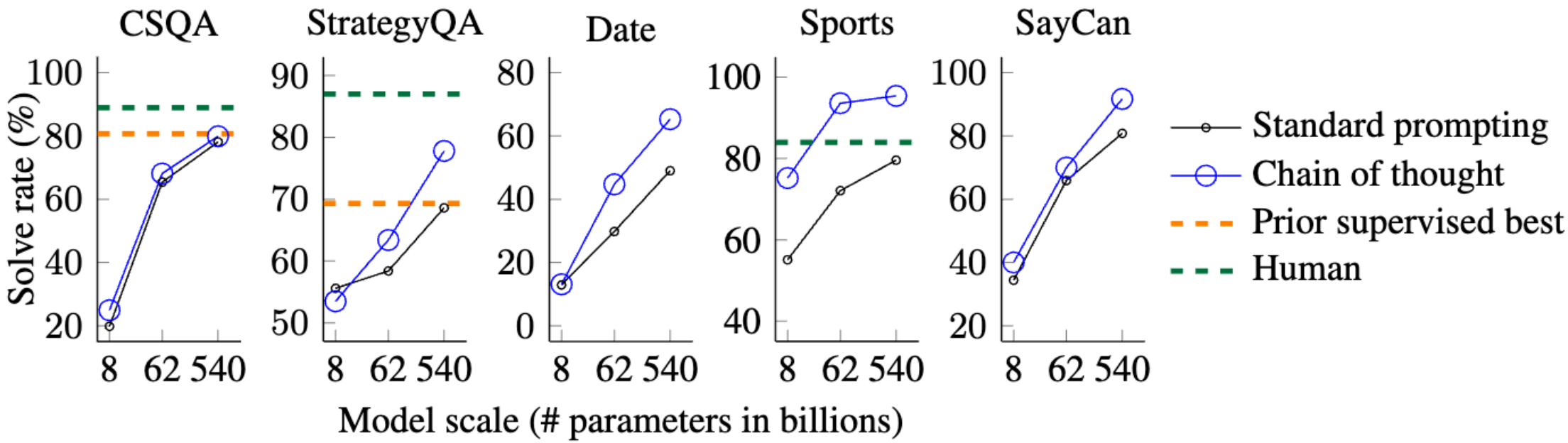
- 👉 Дообучение GPT-3 под задачу
- 👉 Лучшая модель на бенчмарке на 2022 год
- 👉 Стандартный промптинг — few-shot
- 👉 Chain-of-Thought



[2] — Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, Wei et al., Google Research, 2023

Применимость

На самом деле лучше везде, если необходим “reasoning”

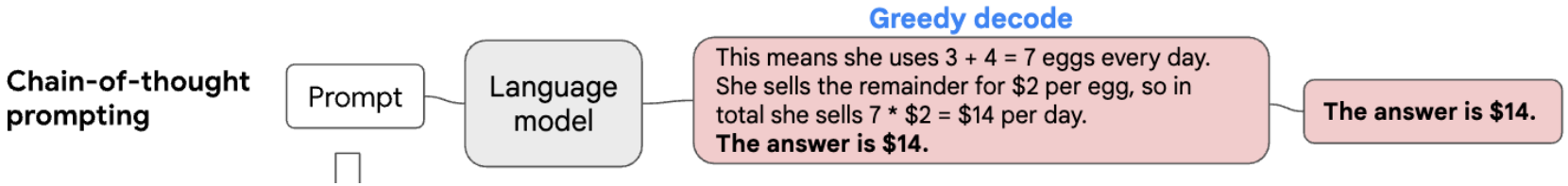


[2] — Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, Wei et al., Google Research, 2023

Self-Consistency Decoding

LLM: строит распределение по словарю, USER: берем жадно или семплируем

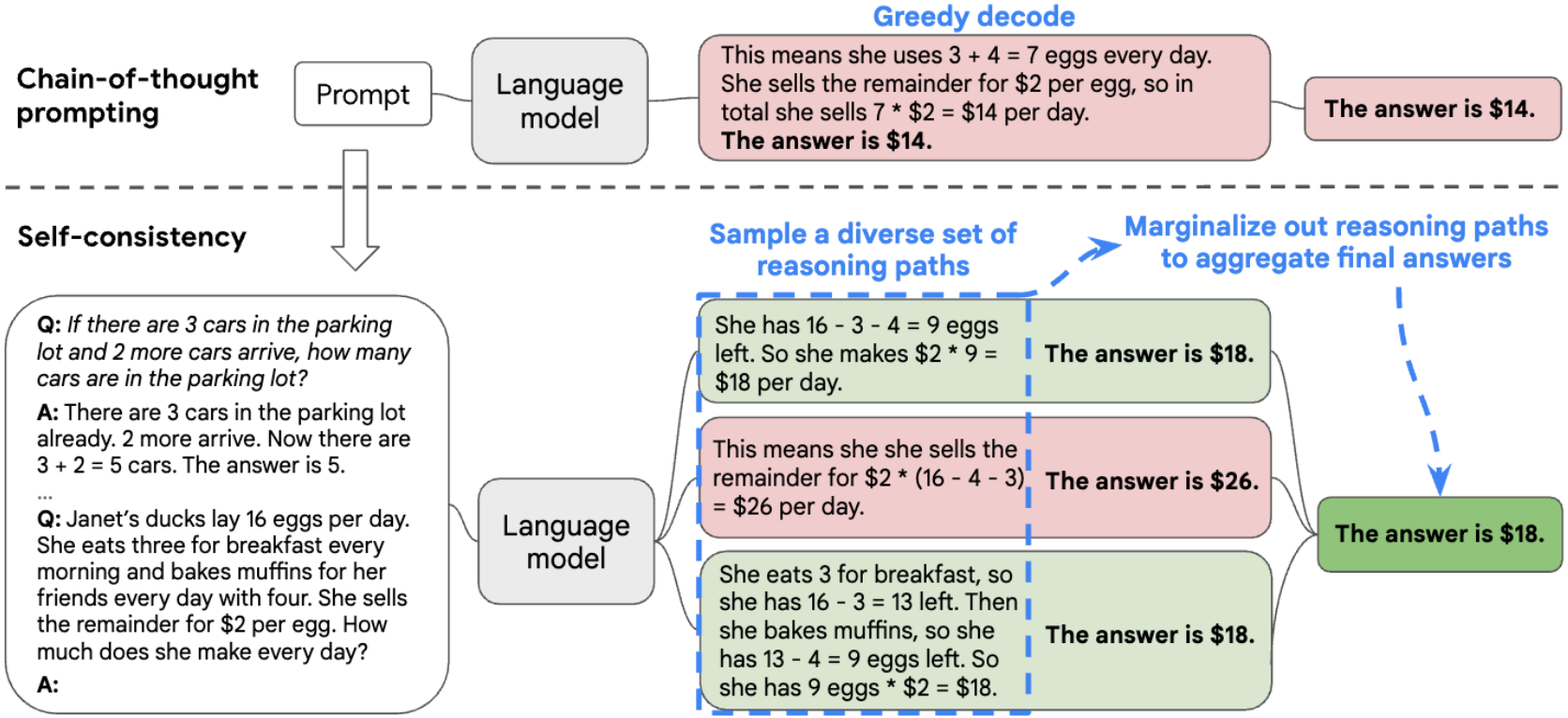
А где гарантия???



Self-Consistency Decoding

Не ошибается лишь тот, кто ничего не делает 🐀

Давайте дадим второй шанс, или третий...



[4] — Self-Consistency Improves Chain of Thought Reasoning in Language Models, Wang et al., Google Research, 2023

Free-form Generation

Question: Where do people drink less coffee than they do in Mexico?
Answer: ... Some examples include Japan, China and the United Kingdom...

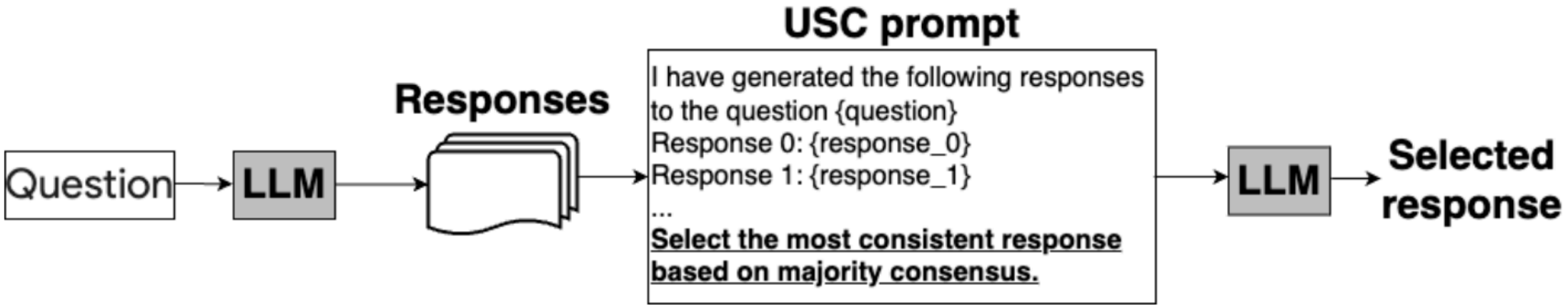
Question: Where do people drink less coffee than they do in Mexico?
Answer: ... People in countries like Japan, China, and India typically drink less coffee...

Question: Where do people drink less coffee than they do in Mexico?
Answer: ... There are several countries where people generally drink less compared to Mexico. Some of these countries include:
1. Japan: ...
2. China: ...
3. Saudi Arabia: ...
4. India: ...
...

Не всегда задачу можно формализовать для понятного извлечения ответа после размышления 🫠

Universal Self-Consistency

Попросим LLM саму выбрать ответ 🙃



[5] — Universal Self-Consistency for Large Language Model Generation, Xinyun Chen, Renat Aksitov et al., DeepMind, 2023

Интуиция и лайфхаки

- 👉 LLM лучше понимает, если использовать знакомый язык и конструкции
- 👉 LLM очень легко отвлекается
  Не надо добавлять много информации “на всякий случай”
- 👉 Если информации не было на обучение или в промпте, то LLM не ответит **верно**
- 👉 Если вы смотрите на промпт и не понимаете, что требуется, то и LLM тоже :)
- 👉 LLM не умеет сама исправлять размышления
  Если ушла не туда, то там и останется

Chain-of-Though | Выводы

Плюсы:

- 👉 Улучшаем качество на reasoning задачах
- 👉 Не требует дообучения
- 👉 Получаем более интерпретируемый результат

Chain-of-Though | Выводы

Плюсы:

- 👉 Улучшаем качество на reasoning задачах
- 👉 Не требует дообучения
- 👉 Получаем более интерпретируемый результат

Минусы:

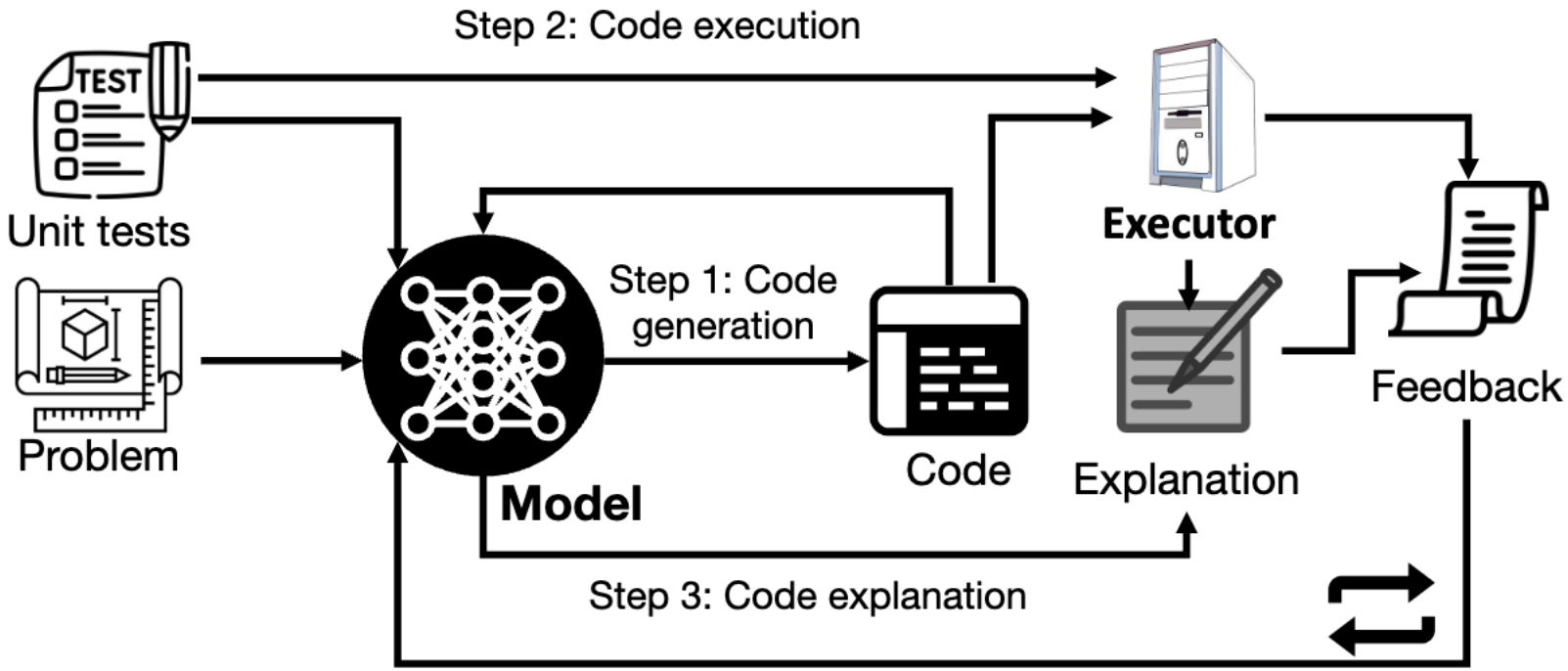
- 👉 Увеличиваем число токенов на генерации и в промпте ⇒ дольше работает
- 👉 Это не панацея, если LLM плохая, то не поможет

Работа над ошибками 📏

Oracle feedback

Одно из существенных ограничений — LLM не понимает, если ошибается в выводах 😥

1. Модель генерирует код
2. Код исполняется и проверяется тестами
3. Модель объясняет, что код делает
4. Формируется фидбек и начинается следующая итерация дебага



[6] — Teaching Large Language Models to Self-Debug, Xinyun Chen et al., DeepMind, 2023

Self-debugging Improves Code Quality

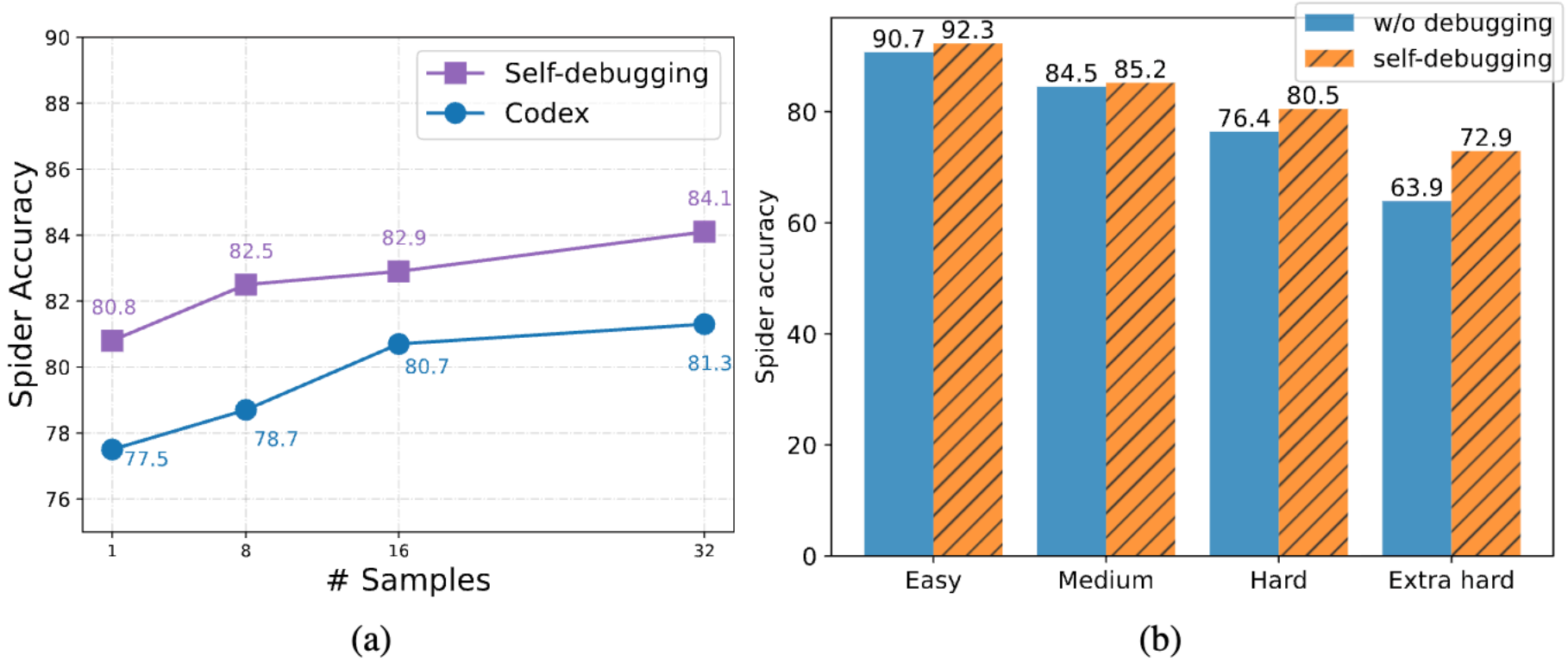


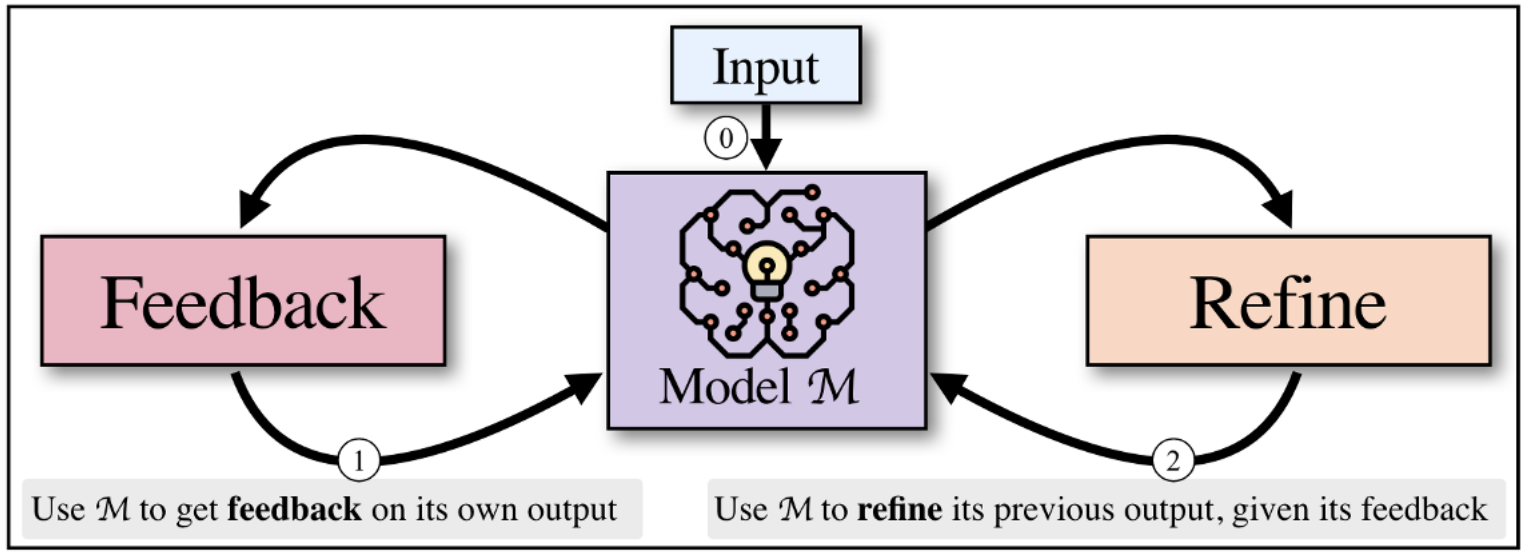
Figure 6: Ablation studies on the Spider development set with Codex. (a) Accuracies with different numbers of initial samples. (b) Breakdown accuracies on problems with different hardness levels.

[6] — Teaching Large Language Models to Self-Debug, Xinyun Chen et al., DeepMind, 2023

Self-Refine

Мы снова делаем это — просим LLM исправить свои же ошибки...

Input (0) передается в модель, ответ (1) передается снова в модель для получения фидбека (2). Все повторяется пока не встретится критерий останова.



[7] — Self-Refine: Iterative Refinement with Self-Feedback, Aman Madaan et al., 2023

Пример FeedBack & Refine | Диалог

Пример FeedBack & Refine | Диалог

(a) **Dialogue:** x, $\mathbf{y_t}$

```
User: I am interested
in playing Table
tennis.

Response: I'm sure
it's a great way to
socialize, stay active
```

Пример FeedBack & Refine | Диалог

(a) **Dialogue:** x, $\mathbf{y_t}$

```
User: I am interested
in playing Table
tennis.

Response: I'm sure
it's a great way to
socialize, stay active
```

(b) **FEEDBACK** **fb**

```
Engaging: Provides no
information about table
tennis or how to play it.

User understanding: Lacks
understanding of user's
needs and state of mind.
```

Пример FeedBack & Refine | Диалог

(a) **Dialogue:** $x, \mathbf{y_t}$

```
User: I am interested
in playing Table
tennis.

Response: I'm sure
it's a great way to
socialize, stay active
```

(b) **FEEDBACK** $\mathbf{fb}$

```
Engaging: Provides no
information about table
tennis or how to play it.

User understanding: Lacks
understanding of user's
needs and state of mind.
```

(c) **REFINE** $\mathbf{y_{t+1}}$

```
Response (refined): That's
great to hear (...) ! It's
a fun sport requiring
quick reflexes and good
hand-eye coordination.
Have you played before, or
are you looking to learn?
```

Пример FeedBack & Refine | Генерация кода

(a) **Dialogue:** $x, \mathbf{y}_t$

```
User: I am interested
in playing Table
tennis.

Response: I'm sure
it's a great way to
socialize, stay active
```

(b) **FEEDBACK** $\mathbf{fb}$

```
Engaging: Provides no
information about table
tennis or how to play it.

User understanding: Lacks
understanding of user's
needs and state of mind.
```

(c) **REFINE** $\mathbf{y}_{t+1}$

```
Response (refined): That's
great to hear (...) ! It's
a fun sport requiring
quick reflexes and good
hand-eye coordination.
Have you played before, or
are you looking to learn?
```

Пример FeedBack & Refine | Генерация кода

(a) **Dialogue:** x, $\mathbf{y_t}$

```
User: I am interested
in playing Table
tennis.

Response: I'm sure
it's a great way to
socialize, stay active
```

(b) **FEEDBACK** $\mathbf{fb}$

```
Engaging: Provides no
information about table
tennis or how to play it.

User understanding: Lacks
understanding of user's
needs and state of mind.
```

(c) **REFINE** $\mathbf{y_{t+1}}$

```
Response (refined): That's
great to hear (...) ! It's
a fun sport requiring
quick reflexes and good
hand-eye coordination.
Have you played before, or
are you looking to learn?
```

(d) **Code optimization:** x, $\mathbf{y_t}$

```
Generate sum of 1, ..., N
def sum(n):
    res = 0
    for i in range(n+1):
        res += i
    return res
```

Пример FeedBack & Refine | Генерация кода

(a) **Dialogue:** x, $\mathbf{y_t}$

```
User: I am interested
in playing Table
tennis.

Response: I'm sure
it's a great way to
socialize, stay active
```

(b) **FEEDBACK** fb

```
Engaging: Provides no
information about table
tennis or how to play it.

User understanding: Lacks
understanding of user's
needs and state of mind.
```

(c) **REFINE** $\mathbf{y_{t+1}}$

```
Response (refined): That's
great to hear (...) ! It's
a fun sport requiring
quick reflexes and good
hand-eye coordination.
Have you played before, or
are you looking to learn?
```

(d) **Code optimization:** x, $\mathbf{y_t}$

```
Generate sum of 1, ..., N
def sum(n):
    res = 0
    for i in range(n+1):
        res += i
    return res
```

(e) **FEEDBACK** fb

```
This code is slow as
it uses brute force.
A better approach is
to use the formula
... (n(n+1))/2.
```

Пример FeedBack & Refine | Генерация кода

(a) **Dialogue:** x, y_t

```
User: I am interested
in playing Table
tennis.

Response: I'm sure
it's a great way to
socialize, stay active
```

(b) **FEEDBACK** fb

```
Engaging: Provides no
information about table
tennis or how to play it.

User understanding: Lacks
understanding of user's
needs and state of mind.
```

(c) **REFINE** y_{t+1}

```
Response (refined): That's
great to hear (...) ! It's
a fun sport requiring
quick reflexes and good
hand-eye coordination.
Have you played before, or
are you looking to learn?
```

(d) **Code optimization:** x, y_t

```
Generate sum of 1, ..., N
def sum(n):
    res = 0
    for i in range(n+1):
        res += i
    return res
```

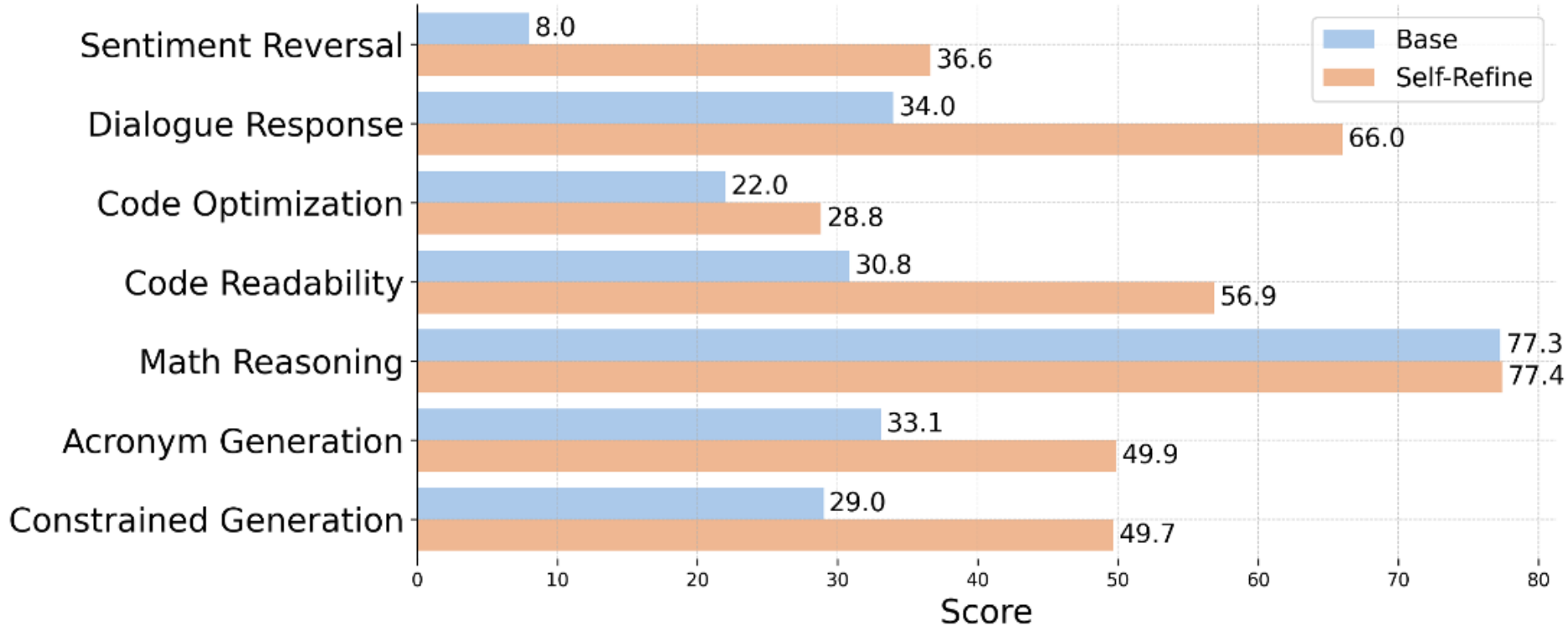
(e) **FEEDBACK** fb

```
This code is slow as
it uses brute force.
A better approach is
to use the formula
... (n(n+1))/2.
```

(f) **REFINE** y_{t+1}

```
Code (refined)

def sum_faster(n):
  return (n*(n+1))//2
```

Улучшаем качество на генеративных задачах



👉 Модели способны самостоятельно улучшать свои ответы

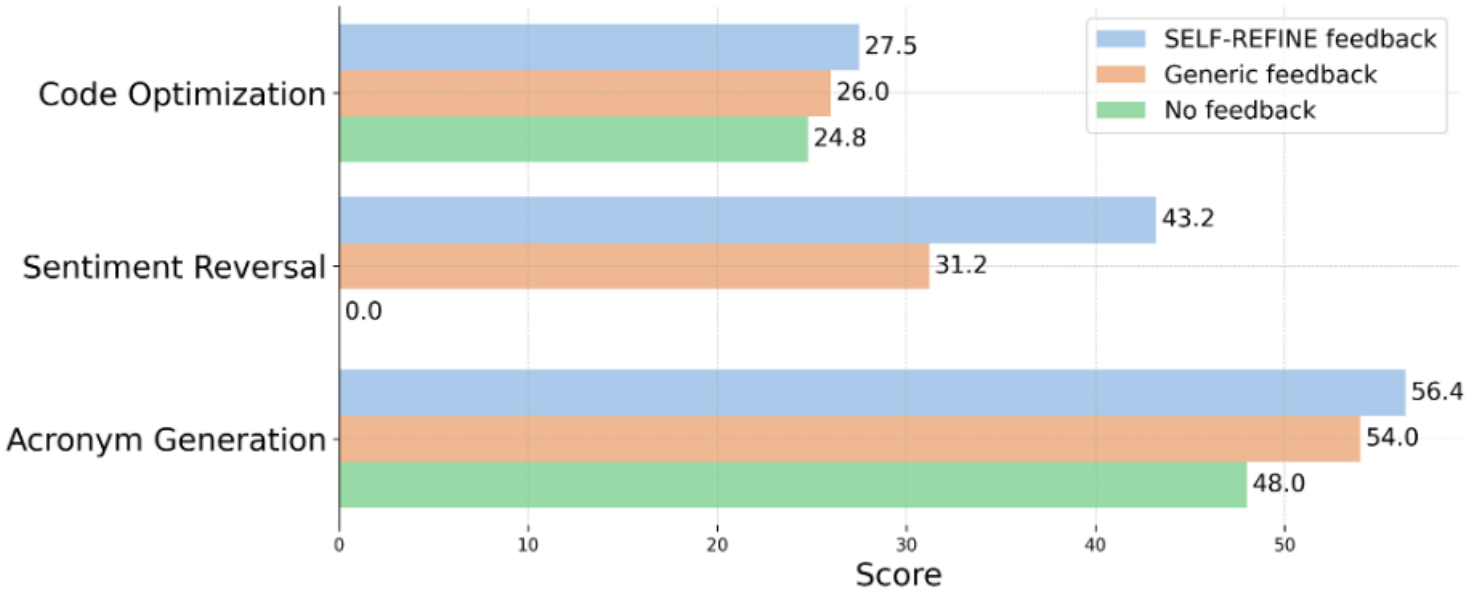
👉 Можно достичь более высокого качества

👉 На Math-Reasoning нет прироста — другая задача!

Важность feedback

Feedback должен быть

- 👉 Specific — выделять конкретные места ответа для исправления
- 👉 Actionable — предлагать конкретные действия для улучшения

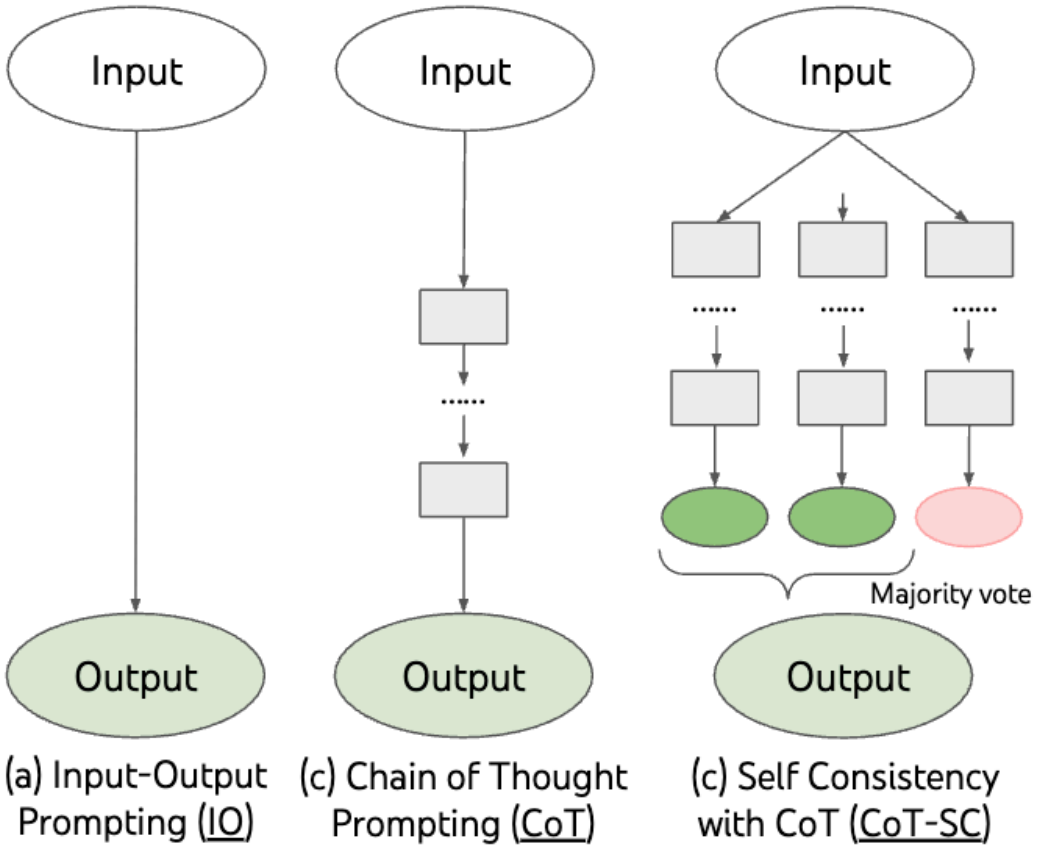


А что если?..

Умеем повышать качество reasoning и свободной генерации, но по отдельности...

Tree of Thoughts

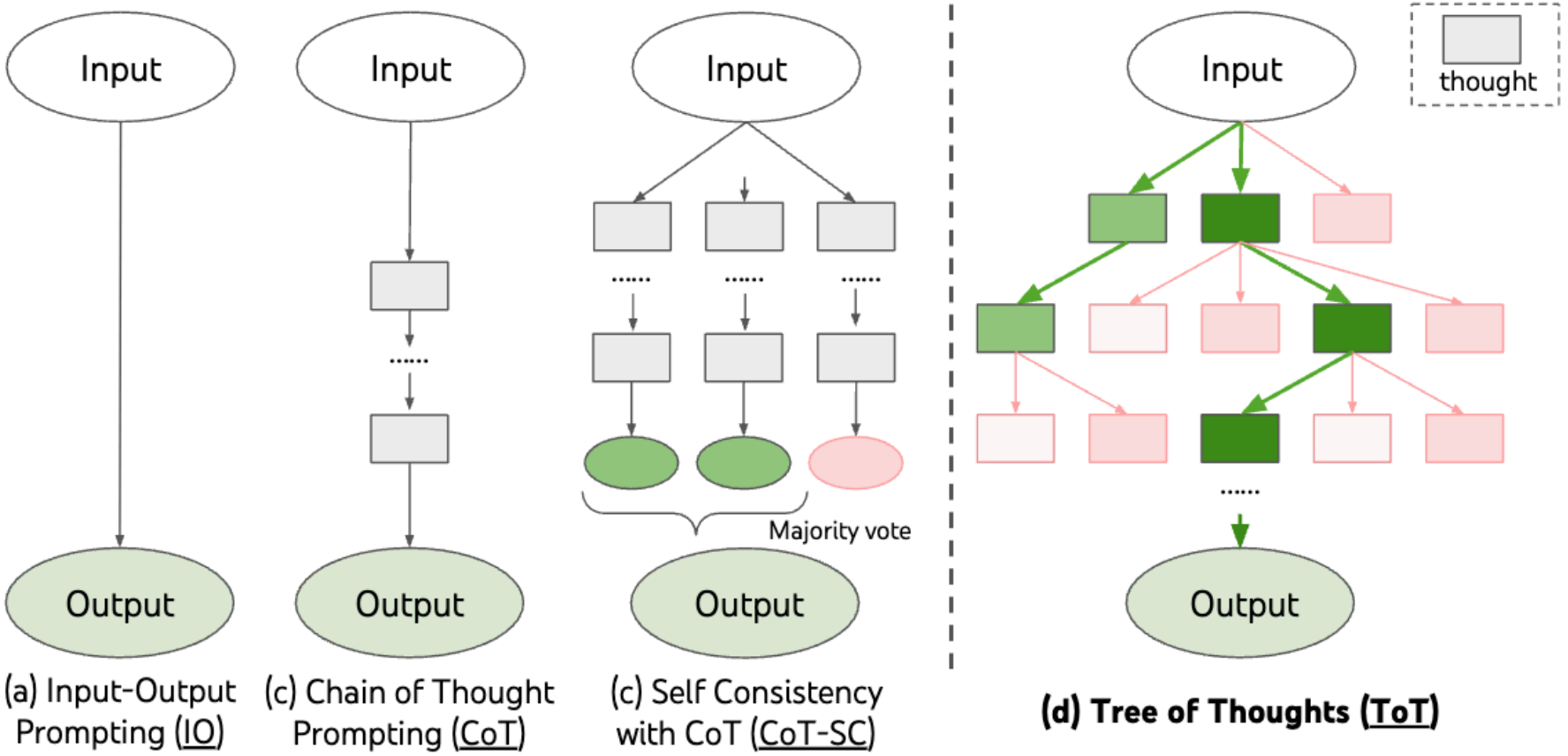
Это уже знаем



[8] — Tree of Thoughts: Deliberate Problem Solving with Large Language Models, Yao et al., Princeton & DeepMind, 2023

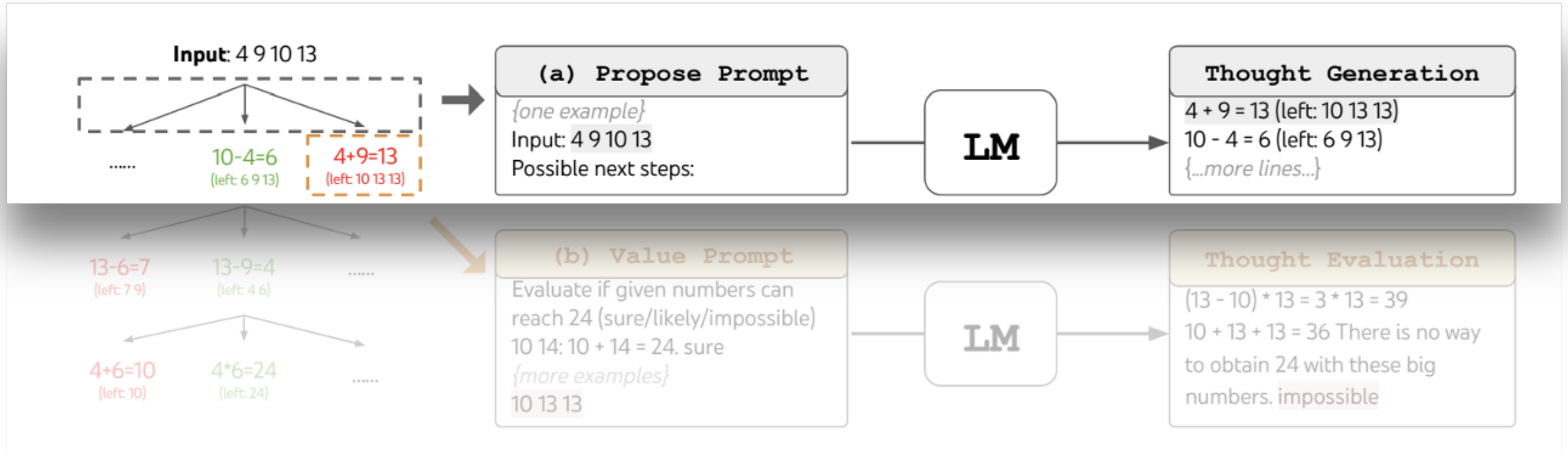
Tree of Thoughts

Это уже знаем



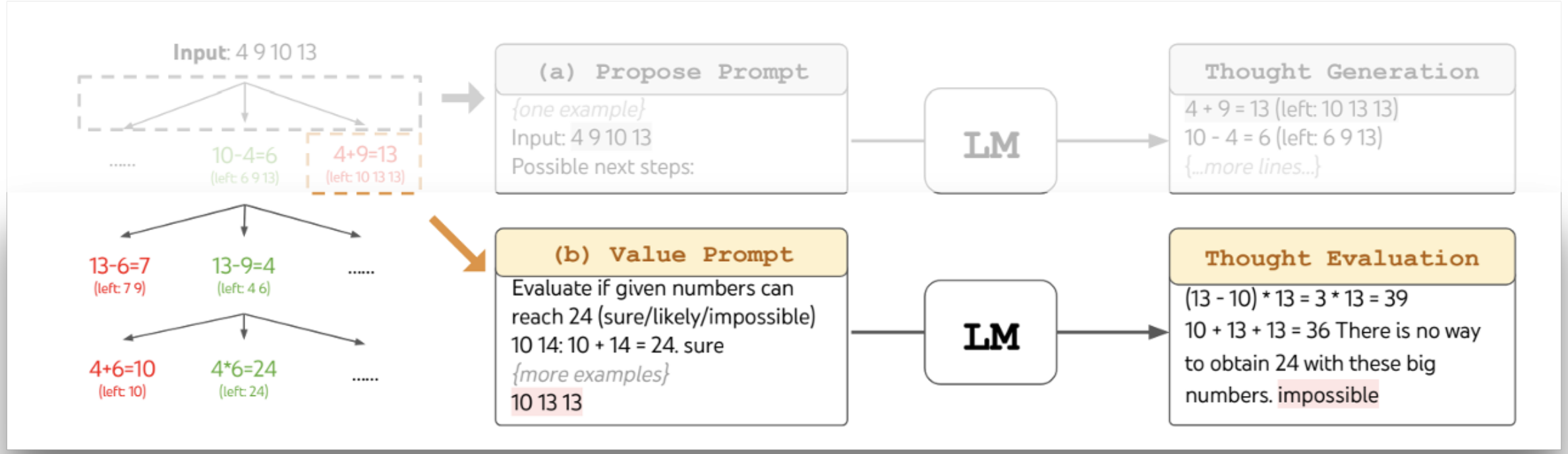
[8] — Tree of Thoughts: Deliberate Problem Solving with Large Language Models, Yao et al., Princeton & DeepMind, 2023

Game of 24



[8] — Tree of Thoughts: Deliberate Problem Solving with Large Language Models, Yao et al., Princeton & DeepMind, 2023

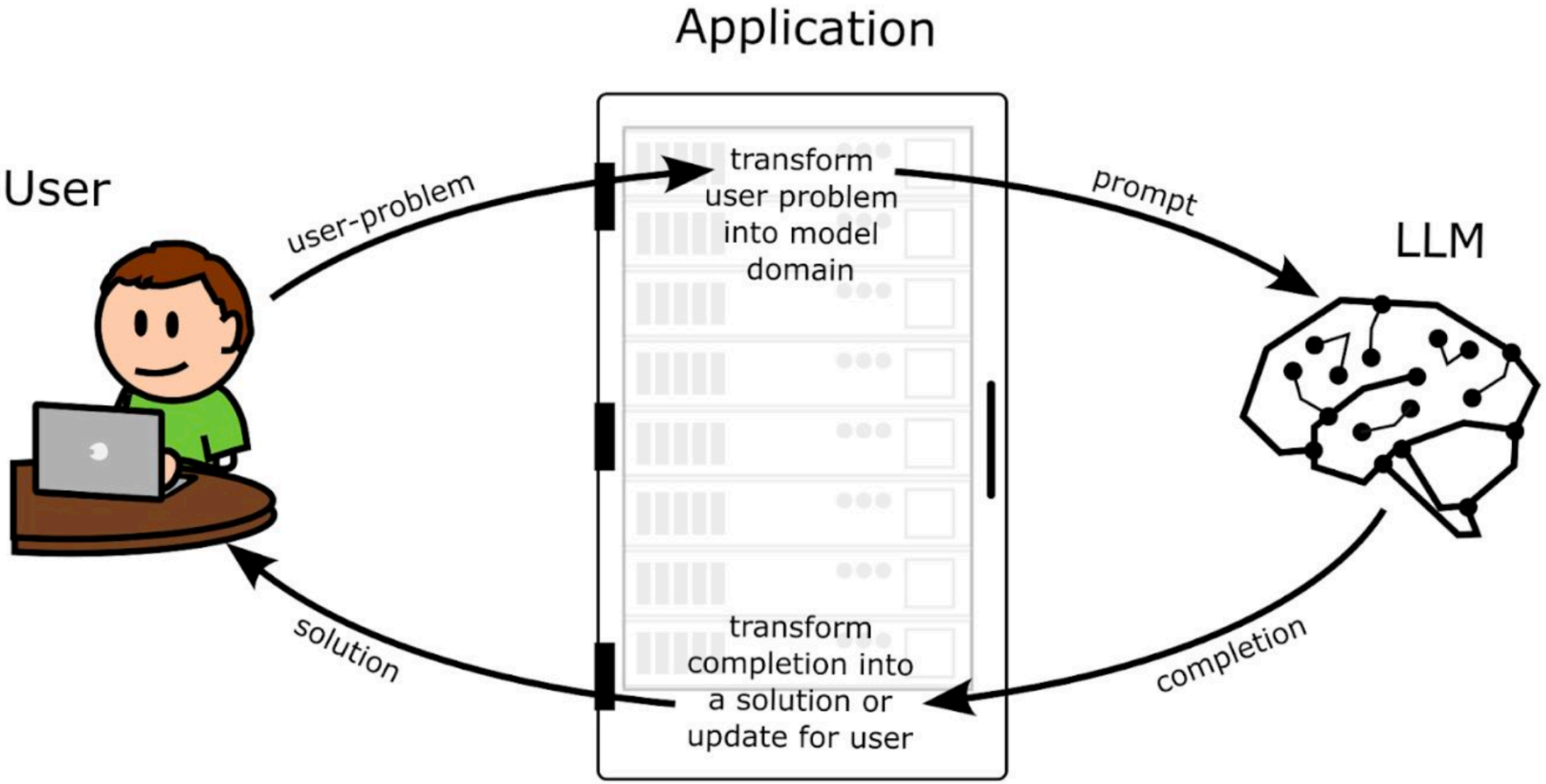
Game of 24



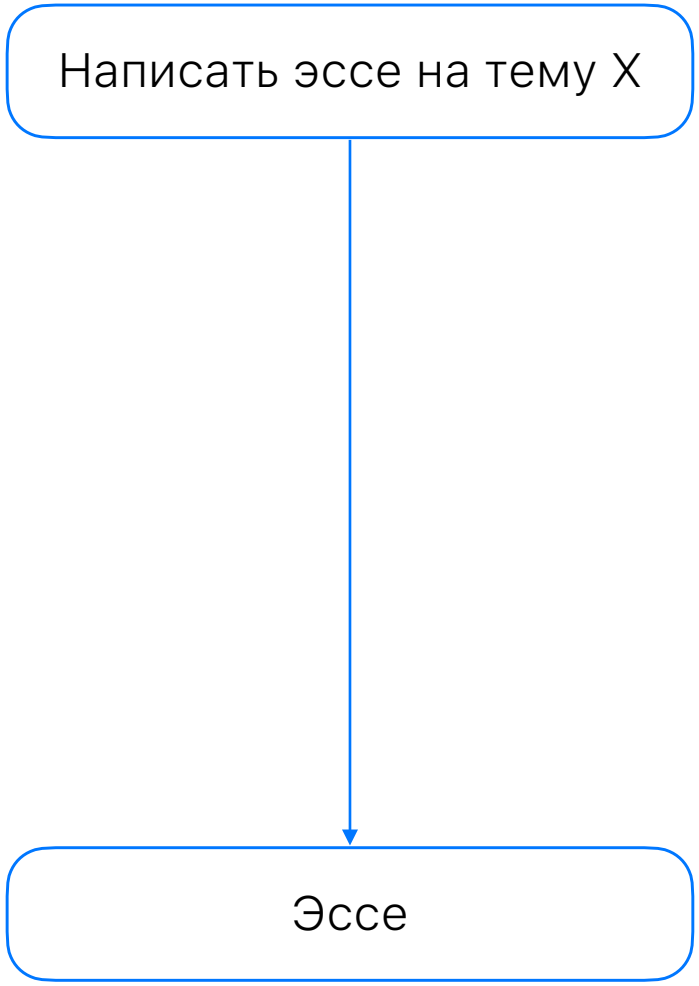
[8] — Tree of Thoughts: Deliberate Problem Solving with Large Language Models, Yao et al., Princeton & DeepMind, 2023

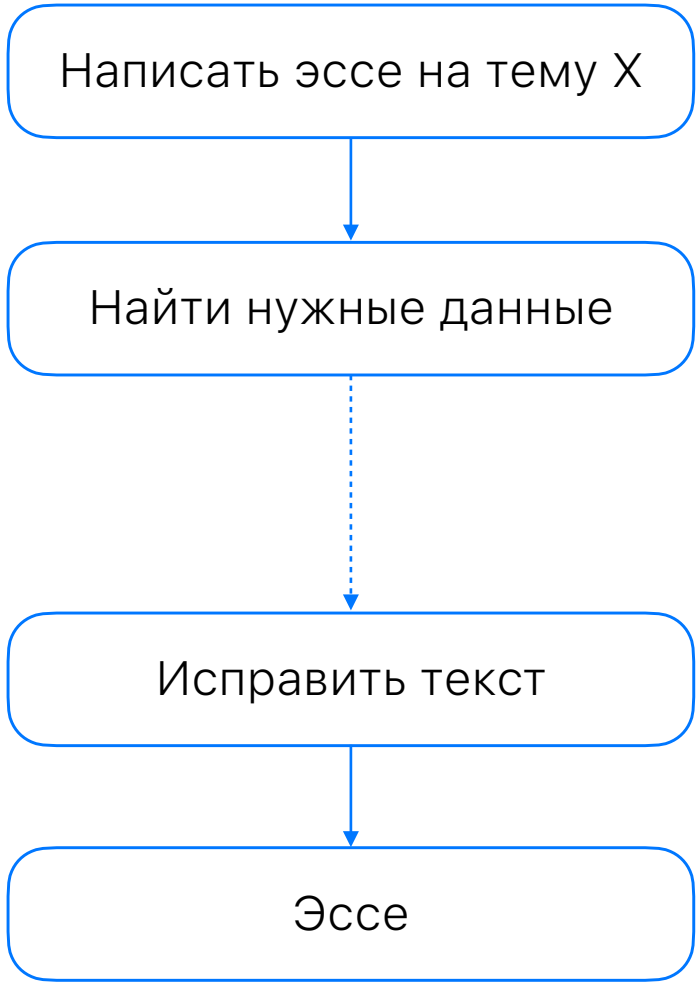
Используем внешний мир

LLM Application

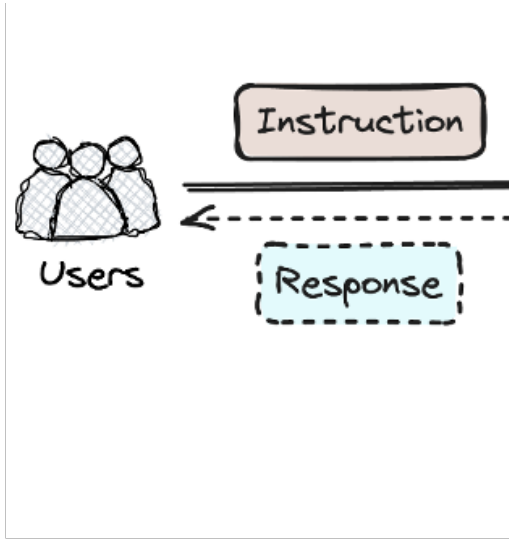


Agentic workflow





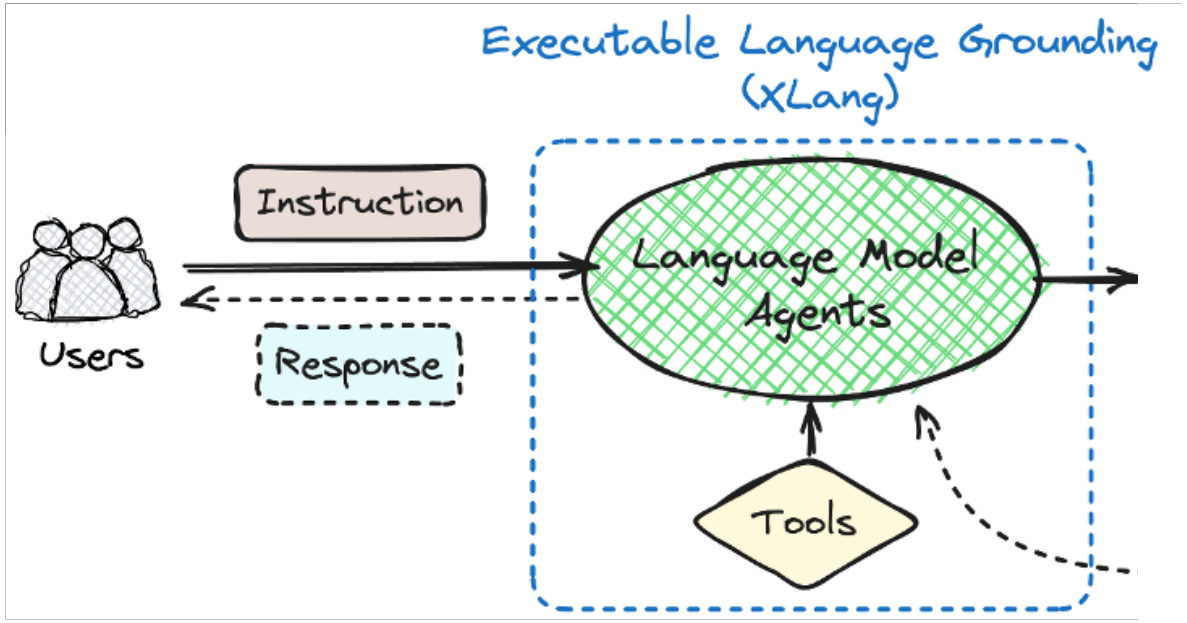
Добавим к LLM инструменты



- 👉 Исполнение кода
- 👉 Wolfram Alpha
- 👉 CI/CD
- 👉 Википедия
- 👉 Поисковик
- 👉 Почта
- 👉 Календарь
- 👉 Другая LLM
- 👉 ...

https://github.com/xlang-ai/xlang-paper-reading

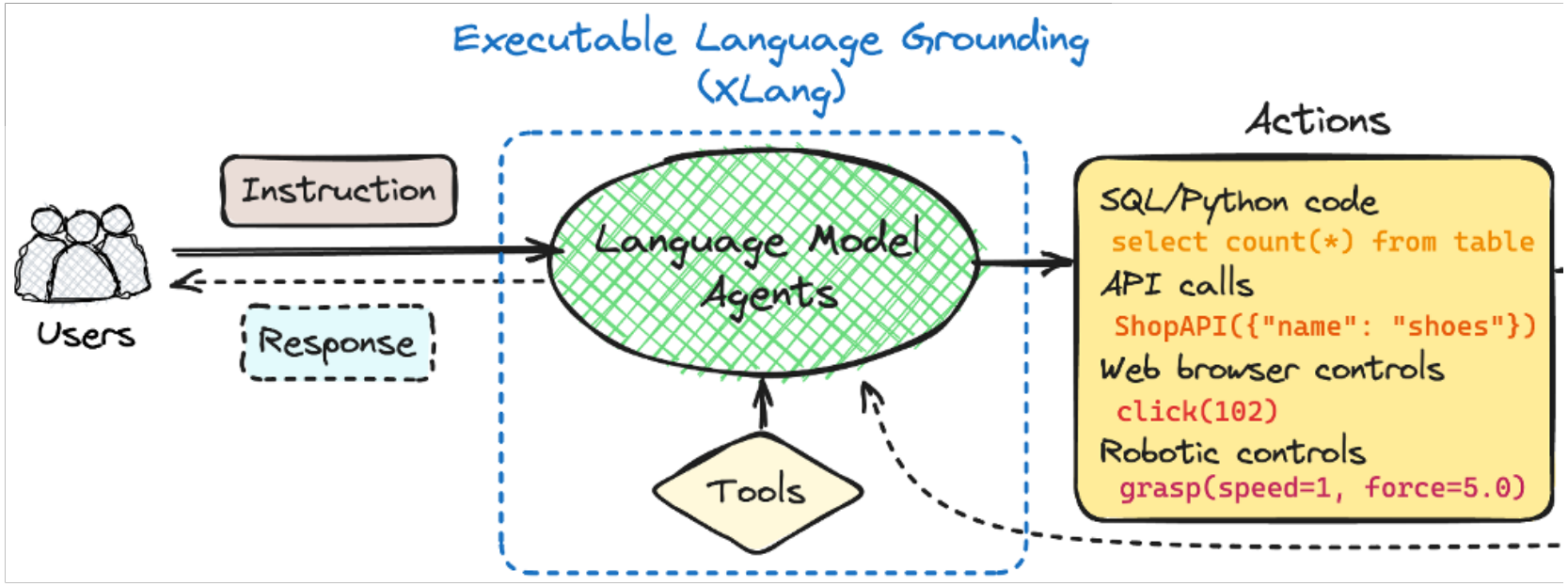
Добавим к LLM инструменты



- 👉 Исполнение кода
- 👉 Wolfram Alpha
- 👉 CI/CD
- 👉 Википедия
- 👉 Поисковик
- 👉 Почта
- 👉 Календарь
- 👉 Другая LLM
- 👉 ...

https://github.com/xlang-ai/xlang-paper-reading

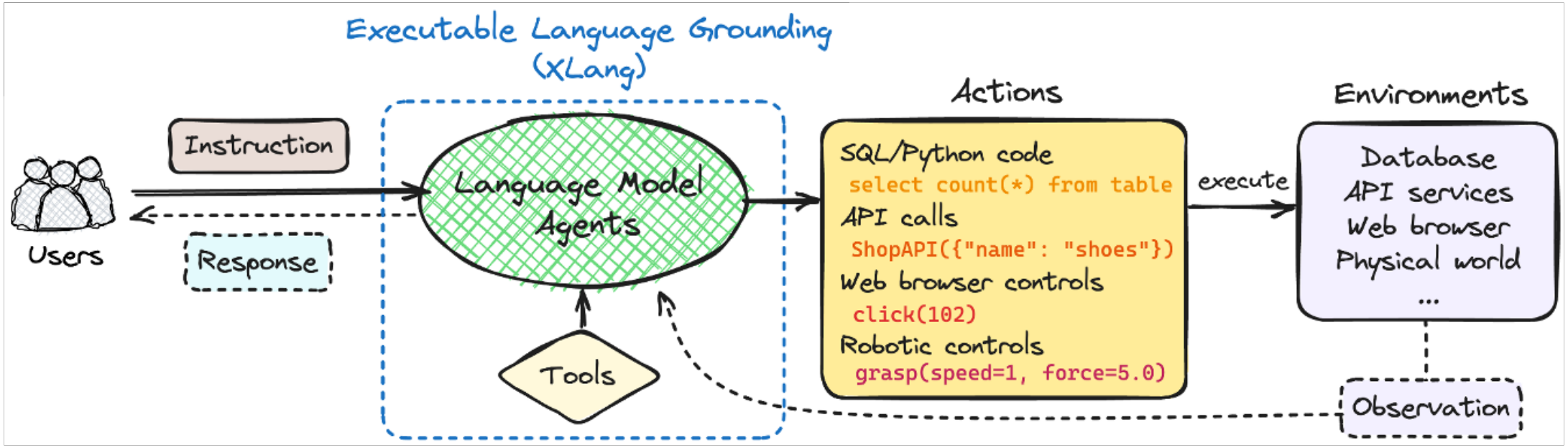
Добавим к LLM инструменты



- 👉 Исполнение кода
- 👉 Википедия
- 👉 Календарь
- 👉 Wolfram Alpha
- 👉 Поисковик
- 👉 Другая LLM
- 👉 CI/CD
- 👉 Почта
- 👉 ...

https://github.com/xlang-ai/xlang-paper-reading

Добавим к LLM инструменты



- 👉 Исполнение кода
- 👉 Wolfram Alpha
- 👉 CI/CD
- 👉 Википедия
- 👉 Поисковик
- 👉 Почта
- 👉 Календарь
- 👉 Другая LLM
- 👉 ...

https://github.com/xlang-ai/xlang-paper-reading

In a nutshell 🌰 | 1. Создание промпта

You have a python function which you can use as a tool:

```
def create_blank_file(directory, filename):
    """Create blank file
    :param str directory: path to root directory
    :param str filename: name of the blank file
    """
```

Use this function if necessary as a python function with proper arguments for solving user query. If the user request does not require the function usage return “pass”.

In a nutshell 🌰 | 1. Создание промпта

You have a python function which you can use as a tool:

```
def create_blank_file(directory, filename):
    """Create blank file
    :param str directory: path to root directory
    :param str filename: name of the blank file
    """
```

Use this function if necessary as a python function with proper arguments for solving user query. If the user request does not require the function usage return “pass”.

User query: Please create two files in a folder my_directory with names first.txt and second.txt

In a nutshell 🌰 | 2. Получаем ответ от LLM

You have a python function which you can use as a tool:

```
def create_blank_file(directory, filename):
    """Create blank file
    :param str directory: path to root directory
    :param str filename: name of the blank file
    """
```

Use this function if necessary as a python function with proper arguments for solving user query. If the user request does not require the function usage return “pass”.

User query: Please create two files in a folder my_directory with names first.txt and second.txt

```
create_blank_file(my_directory, first.txt)
create_blank_file(my_directory, second.txt)
```

In a nutshell 🌰 | 3. Запускаем инструмент

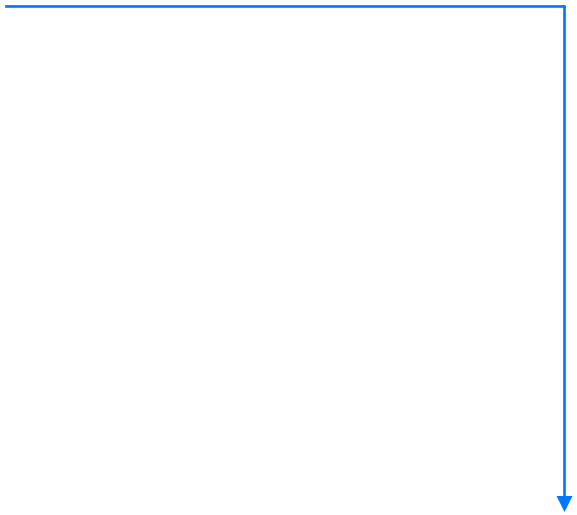
You have a python function which you can use as a tool:

```
def create_blank_file(directory, filename):
    """Create blank file
    :param str directory: path to root directory
    :param str filename: name of the blank file
    """
```

Use this function if necessary as a python function with proper arguments for solving user query. If the user request does not require the function usage return "pass".

User query: Please create two files in a folder my_directory with names first.txt and second.txt

```
create_blank_file(my_directory, first.txt)
create_blank_file(my_directory, second.txt)
```



```
code_snippet="""
create_blank_file(my_directory, "first.txt")
create_blank_file(my_directory, "second.txt")
"""
exec(code_snippet)
```

In a nutshell 🌰 | 3. Запускаем инструмент

```
code_snippet = """create_blank_file(my_directory, first.txt)
create_blank_file(my_directory, second.txt)"""

exec(code_snippet)
```

Function Calling

API часто дает возможность указать доступные инструменты — функции

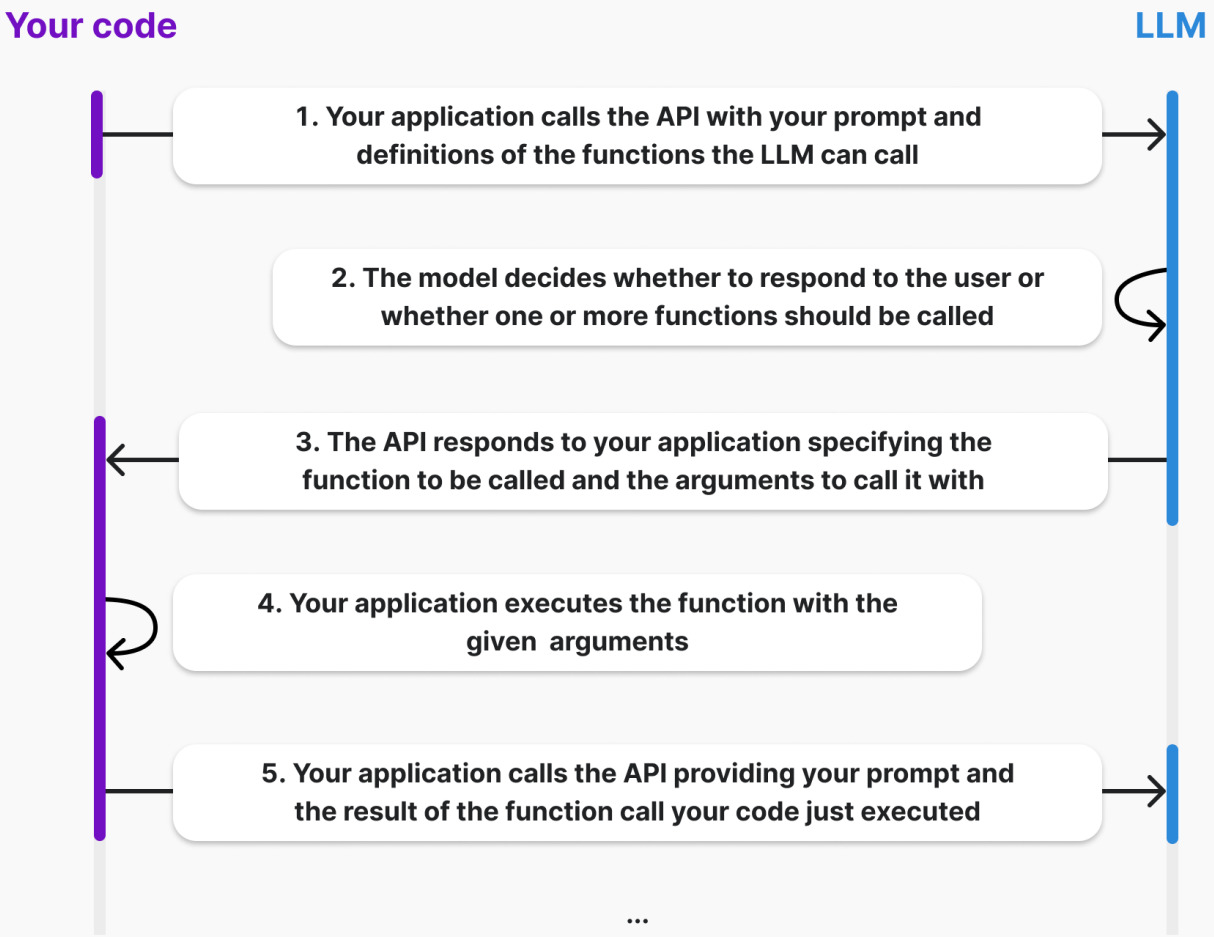
Позволяет получать более достоверные и структурированные ответы от LLM:

```
[{
  "id": "call_12345xyz", "type": "function",
  "function": {
    "name": "get_weather",
    "arguments": "{'location':'Paris'}"
  }
}]
```

https://platform.openai.com/docs/guides/function-calling

```python
from openai import OpenAI

client = OpenAI()

tools = [
  {
      "type": "function",
      "function": {
          "name": "get_weather",
          "parameters": {
              "type": "object",
              "properties": {
                  "location": {"type": "string"}
              },
          },
      },
  }
]

completion = client.chat.completions.create(
  model="gpt-4o",
  messages=[{
    "role": "user",
    "content": "What's the weather like in Paris today?"
  }],
  tools=tools,
)

print(completion.choices[0].message.tool_calls)
```

Function Calling Pipeline



Дополнительно:

- 👉 Указывать required поля
- 👉 Возвращать результат выполнения функции
- 👉 Parallel Function Calling
- 👉 ...

Изучаем документацию

🦜🔗 LangChain

Фреймворк для разработки приложений на основе LLM

- 👉 Интеграция со множеством API и LLM провайдерами: OpenAI, Anthropic, HF, ...
- 👉 Компоненты для работы с IO, Retrieval и агентами, промптами
- 👉 LangGraph — stateful приложения

[GitHub: langchain]

```
from langchain_core.prompts import ChatPromptTemplate
from langchain_openai import ChatOpenAI
from pydantic import BaseModel, Field

tagging_prompt = ChatPromptTemplate.from_template(
"""Extract the desired information from the following passage.
Only extract the properties mentioned in the 'Classification' function.

Passage:
{input}
""")

class Classification(BaseModel):
    sentiment: str = Field(description="The sentiment of the text")
    language: str = Field(description="The language the text is written in")
    aggressiveness: int =
     Field(description="How aggressive the text is on a scale from 1 to
     10")

llm = ChatOpenAI(temperature=0, model="gpt-4o-mini")\
        .with_structured_output(Classification)



>>> inp = "Estoy increiblemente contento de haberte conocido! Creo que
seremos muy buenos amigos!"
>>> prompt = tagging_prompt.invoke({"input": inp})
>>> llm.invoke(prompt)
Classification(sentiment='positive', aggressiveness=1, language='Spanish')
```

Давайте жить дружно ❤️

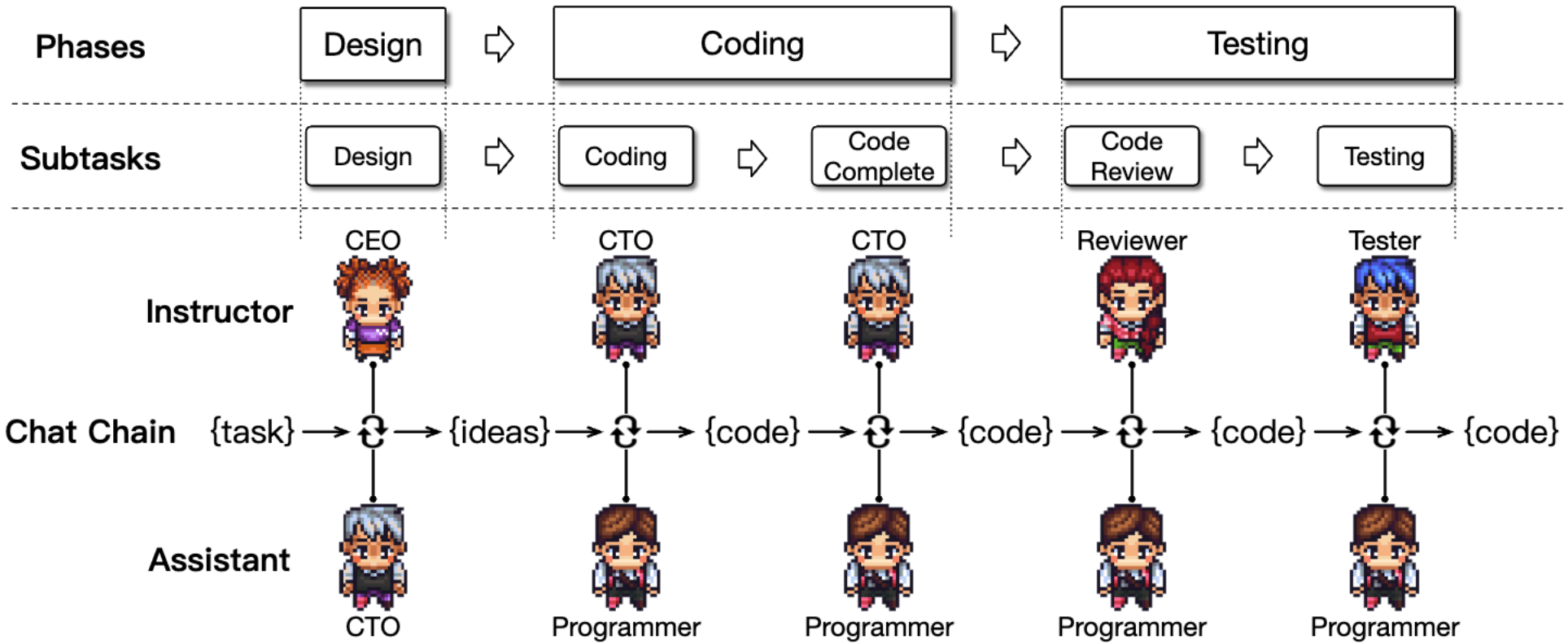
Счастливый мир будущего

LLM-based application на основе множества отдельных агентов:

1. Декомпозируем задачу на отдельные части
2. Для каждой задачи определим свою отдельную LLM
3. Заставим агентов взаимодействовать друг с другом для решения крупной задачи

[9] — ChatDev: Communicative Agents for Software Development, Qian et al., Tsinghua University, 2024

Можем разработать целое приложение с LLM*



* — если повезет

Ресурсы 

Учение — свет, неучение — тьма.

Что не успели разобрать:

👉 Галлюцинации — какие бывают и как бороться

👉 Атаки на LLM и защиты от инъекций

👉 Агентность и CoT, Self-Refine, ... — ReAct

👉 Интеграция агентов в готовые инструменты

Полезные ресурсы:

1. Large Language Model Agents (MOOC)
2. Prompt Engineering Guide

На этом все🤝

Егор Спирин — vk.com/boss
VK Lab — vk.com/lab