



ИНФОРМАТИКА И ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ/INFORMATICS AND INFORMATION PROCESSES

DOI: <https://doi.org/10.60797/itech.2026.12.1>

EDN: EUXYMB

СРАВНИТЕЛЬНЫЙ ОБЗОР ИСПОЛЬЗОВАНИЯ РАССУЖДЕНИЙ (REASONING) В БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЯХ

Обзор

Валухов И.А.^{1,*}, Гаджиев Т.Э.²^{1,2} Южный федеральный университет, Таганрог, Российская Федерация

* Корреспондирующий автор (valuhov[at]sfedu.ru)

Предложена: 27.04.2025; Принята: 01.09.2026; Опубликовано: 14.10.2026

Аннотация

Проведён подробный сравнительный анализ современных подходов к реализации и оценке рассуждений в больших языковых моделях (LLM) — ключевой технологии в области искусственного интеллекта и обработки естественного языка. Рассмотрены и проанализированы методы цепочки рассуждений (Chain-of-Thought, CoT), автоматические цепочки рассуждений (Automatic Chain-of-Thought, Auto-CoT), дерево рассуждений (Tree-of-Thought, ToT) и граф рассуждений (Graph of Thoughts, GoT), а также архитектурные модификации, такие как масштабирование вычислений во время инференса (Test-time Compute), направленные на увеличение вычислительной мощности в процессе генерации ответов. Особое внимание уделено преимуществам и ограничениям каждого из этих методов, а также механизмам их работы внутри архитектуры трансформеров, включая взаимодействие слоёв внимания и MLP-модулей. Подчёркнута важность комплексных и многоступенчатых тестов, которые требуют от моделей не просто извлечения знаний, а настоящего логического вывода и способности к интервенциям, что позволяет повысить качество и надёжность рассуждений. Рассмотрены также современные техники промптинга, такие как обучение с небольшим числом примеров (few-shot) и без примеров (zero-shot CoT), и их влияние на точность решений в различных задачах — от арифметики до контрафактического мышления.

Ключевые слова: большие языковые модели (LLM), рассуждение, Chain-of-Thought (CoT), Automatic Chain-of-Thought (Auto-CoT), Tree-of-Thought (ToT), Graph of Thoughts (GoT), Test-time Compute, оценка рассуждений, промпт-инжиниринг, архитектура трансформеров.

A COMPARATIVE REVIEW OF THE USE OF REASONING IN LARGE LANGUAGE MODELS

Review article

Valukhov I.A.^{1,*}, Gadzhiev T.E.²^{1,2} Southern Federal University, Taganrog, Russian Federation

* Corresponding author (valuhov[at]sfedu.ru)

Suggested: 27.04.2025; Accepted: 01.09.2026; Published: 14.10.2026

Abstract

A detailed comparative analysis has been carried out of modern approaches to the implementation and evaluation of reasoning in large language models (LLMs) — a key technology in the fields of artificial intelligence and natural language processing. Methods such as Chain-of-Thought (CoT), Automatic Chain-of-Thought (Auto-CoT), Tree-of-Thought (ToT) and Graph of Thoughts (GoT), as well as architectural modifications such as Test-time Compute, aimed at increasing computational power during response generation were examined. Particular attention is paid to the advantages and limitations of each of these methods, as well as the mechanisms by which they operate within the transformer architecture, including the interaction between attention layers and MLP modules. The importance of complex, multistage tests is emphasised; these require models not merely to extract knowledge, but to perform genuine logical reasoning and demonstrate the ability to intervene, thereby improving the quality and reliability of their reasoning. Modern prompting techniques, such as few-shot and zero-shot CoT training, are also examined, along with their impact on the accuracy of solutions across various tasks — from arithmetic to counterfactual reasoning.

Keywords: large language models (LLMs), reasoning, Chain-of-Thought (CoT), Automatic Chain-of-Thought (Auto-CoT), Tree-of-Thought (ToT), Graph of Thoughts (GoT), test-time compute, reasoning evaluation, prompt engineering, transformer architecture.

Введение

В последние годы большие языковые модели (Large Language Models, LLMs) достигли значительных успехов в обработке естественного языка, включая задачи перевода, суммаризации и ответов на вопросы. Одним из ключевых направлений исследований является способность LLMs к рассуждению — логическому и последовательному мышлению, позволяющему решать сложные задачи, требующие многошагового вывода [1]. В отличие от простого сопоставления шаблонов, рассуждение требует многошагового анализа и построения цепочек логических выводов. Исследования показывают, что способности к рассуждению проявляются преимущественно в масштабных моделях с сотнями миллиардов параметров, таких как GPT-3 и PaLM. В данной статье проводится сравнительный анализ

современных подходов к реализации и оценке рассуждений в LLMs, а также обсуждаются основные методы улучшения их способностей к рассуждению.

Методы и принципы исследования

2.1. Методы реализации рассуждений

2.1.1. Chain-of-Thought

Метод Chain-of-Thought (CoT) стал ключевым инструментом для раскрытия потенциала к рассуждению LLMs. Его суть заключается в генерации промежуточных шагов, имитирующих логическое мышление человека, что позволяет моделям решать задачи, требующие многоэтапного анализа. Например, при решении уравнения $3x+5=20$, модель не сразу выдаёт ответ, а последовательно выводит: «1. Вычтем 5: $3x=15$ 2. Разделим на 3: $x=5$. Такой подход не только повышает точность, но и делает выводы моделей интерпретируемыми, что критично для медицины, юриспруденции и образовательных приложений [2]. Пример CoT промтинга приведена на рисунке 1.

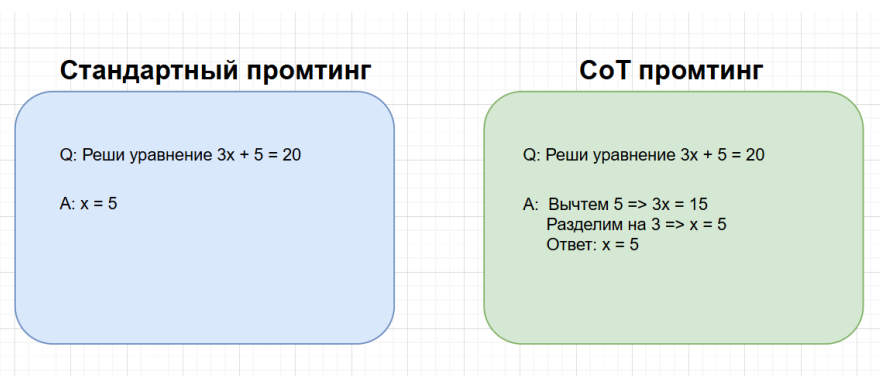


Рисунок 1 - Пример CoT промтинга
DOI: <https://doi.org/10.60797/itech.2026.12.1.1>

Формирование цепочки рассуждений достигается через few-shot обучение: включение в промпт примеров с готовыми рассуждениями [2]. А также через использование динамического планирования, — модели самостоятельно определяют последовательность шагов, адаптируясь к контексту задачи [3].

Эксперименты на бенчмарке GSM8K (математические задачи) демонстрируют, что CoT повышает точность GPT-4 на 25%, а в логических задачах (например, sudoku) модификация метода Tree-of-Thought (ToT) превосходит стандартные подходы на 37% [4]. Однако малогабаритные системы (менее 10 млрд параметров) часто генерируют противоречивые цепочки [2].

Механизмы рассуждения в архитектуре трансформеров раскрываются через анализ взаимодействия слоев внимания и MLP-модулей. Современные исследования используют методы причинно-следственного анализа и интервенций для изучения этих процессов.

Слой внимания управляют фокусировкой на ключевых элементах задачи. Они обеспечивают контекстуальное связывание элементов входной последовательности. В задачах арифметики и логики внимание фокусируется на ключевых токенах (числах, операторах), формируя временные ассоциации [5]. Например, в работе Stolfo et al. (2023) показано, что модификация активаций определённых голов внимания нарушает цепочки вычислений на 34% [6].

MLP-модули выполняют нелинейные преобразования, кодируя промежуточные результаты. Причинный анализ показывает, что 15% нейронов в MLP специализируются на логических операциях (AND/OR) [7]. Эксперименты с упрощенной архитектурой Genie-2 подтверждают, что сочетание внимания и MLP обеспечивает иерархическую обработку информации: от лексического анализа к абстрактным операциям [3].

Данный метод работает с зашумленными данными. В задачах декодирования шифров CoT сохраняет точность 59% при наличии ошибок во входных данных [3].

К недостаткам метода можно отнести увеличение времени инференса в 3 раза [3]. К тому же ошибка на одном шаге из-за галлюцинаций может привести к неверному финальному ответу. В задачах на контрафактическое мышление (CATS) GPT-4 ошибается в 42% случаев [4].

2.1.2. Automatic Chain-of-Thought

Метод Automatic Chain-of-Thought (Auto-CoT) представляет собой инновационный подход к генерации промежуточных рассуждений в LLM, устраняющий необходимость ручного создания демонстраций. В отличие от классического CoT, где разработчики вручную составляют примеры с цепочками логических шагов, Auto-CoT автоматизирует этот процесс, используя саму модель для создания разнообразных и релевантных демонстраций. Это значительно повышает масштабируемость метода и снижает зависимость от человеческого фактора.

Auto-CoT функционирует в два этапа, сочетая кластеризацию вопросов и генерацию рассуждений:

- Кластеризация вопросов.

На первом этапе входные вопросы группируются в кластеры на основе их семантической схожести. Для этого используется модель Sentence-BERT, которая преобразует текстовые задачи в векторные представления. Кластеризация гарантирует, что демонстрации охватят разнообразные типы задач, минимизируя дублирование. Например, вопросы из

области арифметики (например, «Реши уравнение $3x + 5 = 20$ ») и логики (например, «Определи, кто лжёт, исходя из условий задачи») попадают в разные кластеры.

- Генерация рассуждений.

Из каждого кластера выбирается репрезентативный вопрос, для которого модель генерирует цепочку рассуждений с помощью Zero-Shot CoT. Это достигается добавлением к промпту инструкции «Давайте подумаем шаг за шагом». Например, для задачи «Сколько яблок осталось?» модель может сгенерировать: «1. Изначально куплено 10 яблок. 2. Отдано 4 яблока. 3. Куплено ещё 5. 4. Съедено 1. Итого: $10 - 4 + 5 - 1 = 10$ ».

Сгенерированные демонстрации объединяются в контекст, который используется моделью для решения новых задач. Это позволяет LLM адаптировать логические шаги к схожим вопросам без ручного вмешательства.

Ручной CoT требует трудоёмкого создания демонстраций для каждой задачи. В Auto-CoT этот процесс автоматизирован, что сокращает временные затраты на 70–90% [7].

Хотя фраза «Давайте подумаем шаг за шагом» стимулирует рассуждения, отсутствие структурированных демонстраций приводит к ошибкам. Auto-CoT решает эту проблему за счёт разнообразных примеров. Эффективность метода подтверждена экспериментами, на задачах арифметики (GSM8K) Auto-CoT достигает точности 92%, превосходя ручной CoT (91.7%) и Zero-Shot CoT (78.7%) [8], для символьных задач (например, декодирование шифров) прирост составляет 2–3% [9].

Генерация рассуждений в Auto-CoT опирается на архитектурные особенности LLM. Механизмы внимания идентифицируют ключевые элементы задачи (числа, операторы, логические связки) и устанавливают связи между ними. MLP-слои обрабатывают промежуточные шаги, выполняя арифметические и логические операции. В уравнении $3x+5=20$ MLP-нейроны кодируют операции вычитания и деления [9].

2.1.3. Tree-of-Thought

Метод Tree-of-Thought (ToT) — это инновационный фреймворк, разработанный для улучшения способностей к рассуждению LLM. В отличие от CoT, который следует линейной цепочке рассуждений, ToT позволяет моделям исследовать множество альтернативных путей решения, оценивать их и выбирать оптимальный вариант. Этот подход имитирует человеческое мышление, где рассматриваются различные гипотезы перед принятием решения.

Метод ToT основан на трех ключевых компонентах:

Генерация мыслей (Thought Generation) — модель создает промежуточные шаги («мысли»), которые представляют собой возможные пути решения задачи. Например, для математической задачи «Используя числа 4, 5, 6, получить 24» мысли могут включать:

$4 \times 5 = 20 \rightarrow$

$20 + 6 = 26$ (неверно)

$6 - 5 = 1 \rightarrow$

$4 \times 1 = 4 \rightarrow$

$4 \times 6 = 24$ (верно).

Оценка мыслей (Thought Evaluation): Каждая мысль оценивается на предмет ее полезности для достижения цели. Модель использует самооценку (self-consistency) или внешние метрики. Для задачи «Где находится зонт?» модель оценивает логическую согласованность предложенных рассуждений, опираясь на контекст и внешние знания. Например, утверждение «Зонт в прихожей, так как на улице дождь» считается верным, поскольку в дождливую погоду зонт обычно берут при выходе из дома и оставляют у входа после возвращения. В то же время вариант «Зонт в прихожей, так как солнечно» считается неверным, так как при солнечной погоде зонты обычно не используют и хранят их в стандартном месте, например, в шкафу. Модель генерирует несколько возможных ответов, учитывая текущие условия, например, погодные факторы. Такой подход позволяет модели не только формировать ответы, но и обосновывать их, опираясь на реальные знания и логику, что повышает качество и достоверность рассуждений.

Алгоритмы поиска (Search Algorithms) — для исследования дерева мыслей применяются классические алгоритмы: поиск в ширину (BFS) и поиск в глубину (DFS).

Метод ToT более устойчив к ошибкам, так как имеет возможность отката к предыдущим шагам снижает риск «застревания» в неверном решении. Модель анализирует больше вариантов, чем при линейном CoT. Например, в задаче планирования отпуска ToT рассматривает пляжный отдых, горнолыжный курорт и городской туризм, отсеивая неподходящие опции. К тому же дерево мыслей позволяет отслеживать логику модели. Архитектура ToT представлена на рисунке 2.

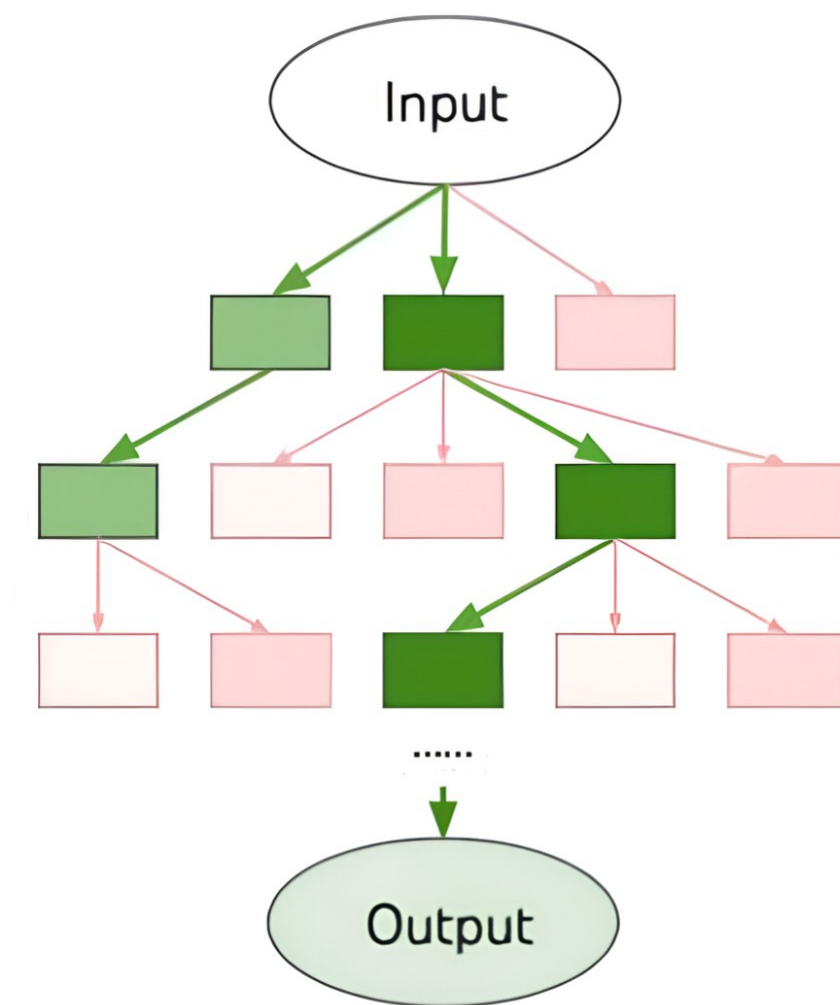


Рисунок 2 - Архитектура ToT
DOI: <https://doi.org/10.60797/itech.2026.12.1.2>

Примечание: источник - media.licdn.com

Но у метода ToT большая вычислительная сложность. Генерация и оценка множества путей увеличивает время инференса в 5 раз. Эффективность ToT проявляется преимущественно у LLM с более 100 млрд параметров и существует большой риск переобучения, — модель может «зацикливаться» на непродуктивных путях, если алгоритм поиска не настроен корректно. Сравнение основных показателей CoT и ToT представлены в таблице 1.

Таблица 1 - Сравнение с CoT
DOI: <https://doi.org/10.60797/itech.2026.12.1.3>

Параметр	ToT	CoT
Структура	Дерево альтернативных путей	Линейная цепочка шагов
Гибкость	Поддерживает возврат и ветвление	Фиксированная последовательность
Ресурсы	Высокие (множество вычислений)	Умеренные
Точность	+15–20% на сложных задачах	Эффективен для простых сценариев

2.1.4. Graph of Thoughts (GoT)

Метод Graph of Thoughts (GoT) — это инновационный фреймворк, разработанный для преодоления ограничений традиционных методов рассуждения в LLM, таких как CoT и ToT. В отличие от линейных или древовидных структур, GoT представляет процесс мышления LLM в виде произвольного графа, где вершины соответствуют «мыслям»

(промежуточным шагам решения), а рёбра отражают зависимости между ними. Такой подход позволяет модели гибко комбинировать, пересматривать и усиливать идеи, имитируя нелинейное человеческое мышление [10], [11].

Система GoT состоит из трёх ключевых компонентов:

Graph of Operations (GoO) — статический граф, заданный экспертом, который определяет последовательность операций для решения задачи.

Graph Reasoning State (GRS) — динамическая структура, хранящая историю мыслей, их состояния и связи. GRS обновляется в реальном времени по мере выполнения операций, что позволяет модели адаптироваться к новым данным или ошибкам [12].

Контроллер управляет выполнением операций, определенных в GoO, координируя взаимодействие модулей.

Работа GoT состоит из инициализации задачи, генерации мыслей и составления обратной связи и пересмотра. Завершается ранжированием и выбором решения. При инициализации задачи контроллер получает входные данные (например, список проектов с бюджетами и ограничение на общие затраты). Далее, на этапе генерации мыслей, модель создаёт множество вариантов решения, представленных как вершины графа. Например, для задачи выбора портфеля проектов генерируются комбинации, удовлетворяющие бюджету [13]. GoT позволяет объединять мысли (например, слить два варианта портфеля в один, максимизируя доходность), создавать циклы (повторно анализировать отвергнутые варианты после получения новых данных), и дистиллировать идеи (выделять ключевые шаги из сложных цепочек) [11].

На этапе ранжирования и выбора решения оценщик присваивает каждой вершине графа баллы (например, на основе доходности портфеля), после чего контроллер выбирает оптимальный путь [13].

Пример применения. Рассмотрим задачу оптимизации инвестиционного портфеля:

- Шаг 1: Генерация 10 вариантов портфелей в рамках бюджета.
- Шаг 2: Валидация — отсеивание вариантов, превышающих лимит.
- Шаг 3: Оценка оставшихся вариантов по доходности.
- Шаг 4: Объединение двух лучших портфелей в новый, более выгодный.

В результате GoT находит решение, превосходящее стандартные методы на 62% по качеству при сокращении затрат на 31% [12]. Сравнение основных показателей GoT с CoT / ToT представлены в таблице 2.

Таблица 2 - Сравнение GoT с CoT и ToT

DOI: <https://doi.org/10.60797/itech.2026.12.1.4>

Параметр	GoT	CoT/ToT
Структура	Произвольный граф	Линейная цепочка (CoT) или дерево (ToT)
Гибкость	Поддерживает циклы, слияние идей	Ограничена линейным/древовидным поиском
Ресурсы	Выше за счёт сложных операций	Умеренные (CoT) / Высокие (ToT)
Точность	+40–62% на комплексных задачах	Эффективны для простых сценариев

2.1.4.1. Преимущества и ограничения

GoT имеет сильные стороны, такие как глубокая адаптивность, так как модель пересматривает решения на основе новых данных, и эффективность, так как происходит сокращение затрат на 31% за счёт оптимизации путей [10], [11].

Но GoT имеет и существенные ограничения, такие как сложность настройки GoO, которая требует экспертного вмешательства для определения операций и увеличение времени инференса.

В целом Graph of Thoughts открывает новую эру в рассуждениях LLM, преодолевая жёсткие рамки предыдущих методов. Его способность моделировать сложные сети мыслей, включая циклы и обратную связь, делает его незаменимым для задач, требующих нелинейного подхода — от финансового анализа до научных исследований. Однако широкое внедрение GoT потребует разработки инструментов автоматизации построения GoO и оптимизации ресурсов.

Архитектура GoT представлена на рисунке 3.

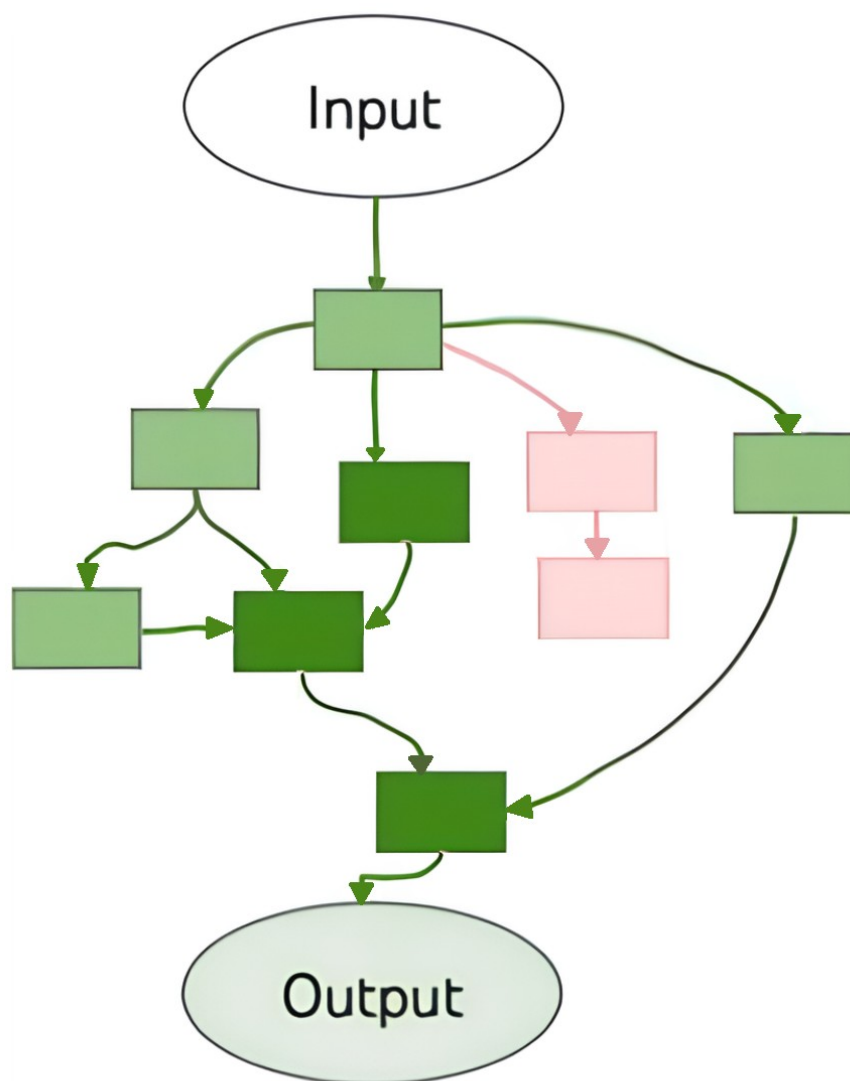


Рисунок 3 - Архитектура GoT
DOI: <https://doi.org/10.60797/itech.2026.12.1.5>

Примечание: источник - media.licdn.com

2.2. Архитектурные модификации

Test-time Compute методы масштабирования вычислений во время вывода фокусируются на увеличении вычислительной мощности в процессе генерации ответов. Ключевой идеей является генерация нескольких вариантов решения с последующим выбором оптимального. Подход Best-of-N предполагает создание N ответов и их ранжирование с помощью самой модели или внешних метрик. Эксперименты показывают, что такой метод повышает точность на 15–20% в задачах математического вывода.

Разновидностью этого подхода является Step-wise BoN, где оптимизация происходит на каждом шаге рассуждения. Это особенно эффективно для многоступенчатых задач, таких как решение уравнений или планирование маршрутов. В работе DeepSeek-R1 комбинируется Best-of-N с механизмом самокоррекции: модель генерирует черновые ответы, анализирует их через self-attention и корректирует ошибки перед финальным выводом.

Еще одним примером служит Speculative Decoding, при котором малогабаритная модель генерирует «черновик» ответа, а крупная модель проверяет и дорабатывает его. Это снижает задержку инференса на 25–60%, сохраняя точность.

Подходы к оценке рассуждений

Оценка способностей к рассуждению больших языковых моделей (LLM) является важным и активно развивающимся направлением исследований. Для этого используются разнообразные бенчмарки, которые включают задачи на логику, арифметику, причинно-следственные выводы и этические дилеммы. Современные ученые подчеркивают необходимость комплексных тестов, которые выходят за рамки простого извлечения знаний и требуют от моделей настоящего логического вывода и способности к интервенциям.

Традиционные метрики, такие как точность (ассигасу), широко применяются для оценки моделей на бенчмарках GSM8K (арифметика) или LogiQA (логика). Однако эти показатели не отражают качество промежуточных шагов рассуждений. Для более детального анализа применяются метрики ROUGE и LCS, которые сравнивают сгенерированные цепочки рассуждений с эталонными, но зачастую не учитывают семантическую эквивалентность и глубину логики.

Глубинный анализ поведения моделей включает причинно-следственный анализ, позволяющий выявлять критические компоненты архитектуры, например, важность MLP-слоев для арифметических операций. Исследования показывают, что изменение порядка условий в задачах часто приводит к ошибкам, что указывает на зависимость моделей от поверхностных паттернов, а не на истинное понимание логики.

Для более сложной оценки используются специализированные бенчмарки. Например, BIG-Bench Hard включает задачи, требующие многошагового вывода и абстрактного мышления, и лишь около 12% моделей достигают точности выше 50%. Другой важный тест — CATS (Causal Testing Scenarios), направленный на проверку способности к контрафактическим рассуждениям. В нём GPT-4 достигает точности около 58%, что близко к уровню новичка-человека.

Несмотря на прогресс, модели остаются склонными к «галлюцинациям» - генерации убедительно звучащей, но неверной информации, особенно в многошаговых рассуждениях. Оценка часто фокусируется на итоговой точности, игнорируя корректность промежуточных шагов, что затрудняет выявление причин ошибок.

В качестве перспективных направлений развития выделяются создание автономных агентов с внешней памятью, способных отслеживать и корректировать цепочки выводов. Также активно разрабатываются мультимодальные бенчмарки, объединяющие текст, код и графические данные, что расширяет возможности оценки рассуждений в различных форматах. Важным шагом является внедрение формальной верификации шагов рассуждений через интеграцию с символьными системами, что позволяет повысить надежность и прозрачность моделей.

Таким образом, оценка способностей к рассуждению LLM требует комплексного подхода, сочетающего традиционные метрики, глубокий анализ архитектуры и специализированные тесты, а также внедрения новых методов верификации и расширения контекстных возможностей моделей.

Основные методы улучшения

Современные методы повышения качества рассуждений в LLM опираются на комбинацию инженерных решений, архитектурных инноваций и гибридных подходов. Как отмечает Chow et al. (2025) в работе Inference-Aware Fine-Tuning for Best-of-N Sampling [14], ключевым трендом стало масштабирование вычислений во время инференса. Техники вроде Best-of-N, где модель генерирует несколько ответов с последующим выбором оптимального через ранжирование, демонстрируют рост точности на 4% для задач математического вывода (Hendrycks MATH). Другой пример - Lookahead Decoding, реализованный в TensorRT-LLM [15], который ускоряет инференс на 3.6× за счёт параллельного предсказания n-грамм и их верификации.

Продвинутый промпт-инжиниринг остается краеугольным камнем. Метод Chain-of-Thought (CoT), предложенный Wei et al. (2022), заставил модели явно генерировать промежуточные шаги, что повысило точность GPT-4 на 25% в задачах арифметики. Более сложные модификации, такие как ReAct (Yao et al., 2022) [16], интегрируют рассуждения с действиями (например, запросами к API), достигая 72% точности на HotPotQA. Однако, как выявило исследование DeepMind, самокоррекция без внешней обратной связи часто приводит к деградации результатов: в 41% случаев модели меняют верные ответы на ошибочные.

Обучение с подкреплением (RL) также играет критическую роль. В работе Chow et al. (2025) PPO-алгоритмы в связке с Best-of-N улучшили точность Gemma 2B на 4%, а DeepSeek-R1-Zero продемонстрировал эмерджентные способности даже без supervised fine-tuning. Однако гибридные подходы, такие как нейро-символическая интеграция, показывают наибольший потенциал. Например, проверка цепочек рассуждений внешними решателями повысила точность геометрических выводов с 44% до 68% (Chow et al., 2025) [14].

Важное направление — обучение с объяснениями. В работе Camburu et al. (2018) [17] на датасете e-SNLI модели, генерирующие естественно-языковые объяснения, снизили ошибки на 15% за счет устранения ложных корреляций. Однако, как подчеркивает Majumder et al. (2022), непоследовательные объяснения остаются проблемой: 30% сгенерированных NLEs (Natural Language Explanations) содержат логические противоречия.

Интерактивные методы типа ReAct (Yao et al., 2022) [16] открывают новые возможности. Например, при планировании путешествия модель анализирует предпочтения пользователя, запрашивает актуальные данные о рейсах через API и динамически корректирует маршрут. Это не только повышает точность, но и усиливает прозрачность: как отмечают авторы, разделение внутренних рассуждений и внешних данных упрощает диагностику ошибок.

Несмотря на прогресс, ключевые вызовы сохраняются. Самокоррекция остается ненадежной без внешней валидации, а ресурсозатратность методов вроде Tree-of-Thought ограничивает их применение. Как заключает Chow et al. (2025), будущее за гибридными системами, сочетающими масштабирование инференса, RL и символьную логику — это позволит преодолеть «эффект плато» в способностях к рассуждению LLM.

Заключение

Современные методы значительно расширили возможности языковых моделей в области рассуждений, однако их способность к глубокому логическому выводу остаётся предметом научных дискуссий. Как подчёркивает Kambhampati (2024), текущие LLMs демонстрируют скорее «универсальный приближенный поиск», основанный на статистических паттернах, чем истинное причинно-следственное мышление [18]. Экспериментальные исследования Kang et al. (2024) подтверждают, что даже продвинутые методы вроде ToT достигают успеха преимущественно за счёт декомпозиции задач на подшаги с низкой вычислительной сложностью, а не за счет фундаментального понимания логики [19].

Прорыв в этой области, как показывают работы по LLM-Modulo Frameworks [18] и Graph Chain-of-Thought [20], требует интеграции нейросетей с явными механизмами символической логики и внешними верификаторами. Например, графовые структуры, связывающие симптомы и заболевания в медицинских базах знаний, снижают уровень галлюцинаций на 27% за счет привязки рассуждений к проверенным данным [20]. При этом гибридные подходы, сочетающие CoT с алгоритмами поиска (Best-of-N, Lookahead Decoding), улучшают точность решения математических задач до 92% [21], демонстрируя потенциал масштабирования инференса.

Однако ключевой вызов, как отмечают Yin et al. (2024), заключается в управлении неопределенностью на каждом шаге рассуждений. Методы вроде Uncertainty-aware Adaptive Guidance (UAG) показывают, что динамический выбор демонстраций через кластеризацию снижает ошибки на 7–15% [22], но требуют значительных вычислительных ресурсов.

Таким образом, будущее развитие рассуждающих LLM лежит в трёх направлениях:

- Создание архитектур с явным разделением статистических и символических модулей [18], [20].
- Оптимизация ресурсоемких методов (ToT, UAG) для маломасштабных моделей [19], [22].
- Разработка онтологий и графов знаний для кросс-доменной проверки выводов [20], [23]. Как показывает практика, только комбинация нейросетевой гибкости и формальной строгости может преодолеть текущие ограничения [18], [21].

Конфликт интересов

Не указан.

Рецензия

Все статьи проходят рецензирование. Но рецензент или автор статьи предпочли не публиковать рецензию к этой статье в открытом доступе. Рецензия может быть предоставлена компетентным органам по запросу.

Conflict of Interest

None declared.

Review

All articles are peer-reviewed. But the reviewer or the author of the article chose not to publish a review of this article in the public domain. The review can be provided to the competent authorities upon request.

Список литературы на английском языке / References in English

1. Payong A. Understanding Reasoning in Large Language Models: Overview of the paper "Towards Reasoning in Large Language Models: A Survey" / A. Payong. — 2024. — URL: <https://www.digitalocean.com/community/tutorials/understanding-reasoning-in-llms> (accessed: 27.04.25)
2. Manning Ch. Self-Attention & Transformers, Natural Language Processing with Deep Learning / Ch. Manning. — 2023. — URL: https://web.stanford.edu/class/cs224n/readings/cs224n-self-attention-transformers-2023_draft.pdf (accessed: 27.04.25)
3. Lee G.-G. Applying Large Language Models and Chain-of-Thought for Automatic Scoring / G.-G. Lee // arxiv.org. — 2023. — URL: <https://arxiv.org/abs/2312.03748> (accessed: 27.04.25)
4. Hebenstreit K. An automatically discovered chain-of-thought prompt generalizes to novel models and datasets / K. Hebenstreit // arxiv.org. — 2023. — URL: <https://arxiv.org/abs/2305.02897> (accessed: 27.04.25)
5. Mus T. Mechanism and emergence of stacked attention heads in multi-layer transformers / T. Mus // arxiv.org. — 2025. — URL: <https://arxiv.org/pdf/2411.12118> (accessed: 27.04.25)
6. Zhang Zh. Automatic chain of thought prompting in large language models / Zh. Zhang // chatgpthero.io. — 2022. — URL: <https://www.chatgpthero.io/wp-content/uploads/2023/12/2210.03493.pdf> (accessed: 27.04.25)
7. Brinkmann J. A Mechanistic Analysis of a Transformer Trained on a Symbolic Multi-Step Reasoning Task / J. Brinkmann // aclanthology.org. — 2024. — URL: <https://aclanthology.org/2024.findings-acl.242.pdf> (accessed: 27.04.25)
8. Transformer Circuits Thread // transformer-circuits.pub. — 2024. — URL: <https://transformer-circuits.pub/2024/september-update/index.html>. (accessed: 27.04.25)
9. Shum K. Automatic Prompt Augmentation and Selection with Chain-of-Thought from Labeled Data / K. Shum // arxiv.org. — 2024. — URL: <https://arxiv.org/html/2302.12822v3> (accessed: 27.04.25)
10. Mayo M. A New Paradigm for Elaborate Problem-Solving in Large Language Models / M. Mayo // kdnuggets.com. — 2023. — URL: <https://www.kdnuggets.com/graph-of-thoughts-a-new-paradigm-for-elaborate-problem-solving-in-large-language-models> (accessed: 27.04.25)
11. Besta M. Graph of Thoughts: Solving Elaborate Problems with Large Language Models / M. Besta // huggingface.co. — 2023. — URL: <https://huggingface.co/papers/2308.09687> (accessed: 27.04.25)
12. Melli G. Graph-of-Thoughts (GoT) Algorithm / G. Melli // gabormelli.com. — 2024. — URL: [https://www.gabormelli.com/RKB/Graph-of-Thoughts_\(GoT\)_Algorithm](https://www.gabormelli.com/RKB/Graph-of-Thoughts_(GoT)_Algorithm) (accessed: 27.04.25)
13. Kumar Saraf M. Graph of Thought (GoT) for Complex Problem of LLM / M. Kumar Saraf // linkedin.com. — 2023. — URL: <https://www.linkedin.com/pulse/graph-thought-got-complex-problem-llm-kumar-saraf-dsc-phd-mba--vcv1c> (accessed: 27.04.25)
14. Chow Y. Inference-Aware Fine-Tuning for Best-of-N Sampling in Large Language Models / Y. Chow // openreview.net. — 2025. — URL: <https://openreview.net/forum?id=77gQUdQhE7> (accessed: 27.04.25)
15. Lookahead Speculative Decoding / Ndivia kaiyux // github.com. — 2025. — URL: <https://github.com/NVIDIA/TensorRT-LLM/blob/main/examples/lookahead/README.md> (accessed: 27.04.25)
16. Yao Sh. ReAct: synergizing reasoning and acting in language models / Sh. Yao // arxiv.org. — 2023. — URL: <https://arxiv.org/pdf/2210.03629.pdf> (accessed: 27.04.25)



17. Camburu O.-M. Neural Networks with Natural Language Explanations / O.-M. Camburu // xaiseminars.doc.ic.ac.uk. — 2024. — URL: https://xaiseminars.doc.ic.ac.uk/materials/2022-05-26_Camburu.pdf (accessed: 27.04.25)
18. Kambhampati S. LLMs Can't Plan, But Can Help Planning in LLM-Modulo Frameworks / S. Kambhampati // [arxiv.org](https://arxiv.org/abs/2402.01817). — 2025. — URL: <https://arxiv.org/abs/2402.01817> (accessed: 27.04.25)
19. Kang L. On the Empirical Complexity of Reasoning and Planning in LLMs / L. Kang // aclanthology.org. — 2024. — URL: <https://aclanthology.org/2024.findings-emnlp.164.pdf> (accessed: 27.04.25)
20. Jin B. Graph Chain-of-Thought: Augmenting Large Language Models by Reasoning on Graphs / B. Jin // arxiv.org. — 2024. — URL: <https://arxiv.org/html/2404.07103> (accessed: 27.04.25)
21. Wei J. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models / J. Wei // arxiv.org. — 2023. — URL: <https://arxiv.org/pdf/2201.11903.pdf> (accessed: 27.04.25)
22. Yin Zh. Reasoning in Flux: Enhancing Large Language Models Reasoning through Uncertainty-aware Adaptive Guidance / Zh. Yin // aclanthology.org. — 2024. — URL: <https://aclanthology.org/2024.acl-long.131.pdf> (accessed: 27.04.25)
23. Zebaze A. Tree of Problems: Improving structured problem solving with compositionality / A. Zebaze // aclanthology.org. — 2024. — URL: <https://aclanthology.org/2024.emnlp-main.1001.pdf> (accessed: 27.04.25)