# Программирование, ориентированное на планирование: рабочий процесс программирования на большом языковой модели

Дата: 2025-01-09 00:00:00

Ссылка на исследование: https://arxiv.org/pdf/2411.14503

Рейтинг: 85
Адаптивность: 90

## Ключевые выводы:

Исследование представляет новый рабочий процесс программирования с использованием больших языковых моделей (LPW), состоящий из двух фаз: генерации решения и реализации кода. Основная цель - улучшить как начальную генерацию кода, так и последующие уточнения. Результаты показывают значительное улучшение точности Pass@1 до 16.4% на различных бенчмарках по сравнению с существующими методами.

## Объяснение метода:

Исследование предлагает двухфазный подход к генерации кода - планирование решения с верификацией и последующую реализацию с отладкой. Метод значительно повышает точность кода, легко адаптируется к стандартным чатам без API, помогает пользователям структурировать запросы и понимать ошибки. Подход применим не только к программированию, но и к другим задачам, требующим пошагового планирования и проверки.

## Ключевые аспекты исследования: 1. **Двухфазный рабочий процесс для генерации кода (LPW)**: Исследование представляет структурированный подход к генерации кода с помощью больших языковых моделей (LLM), разделенный на фазу генерации решения и фазу реализации кода.

**Верификация плана решения**: Ключевая инновация заключается в проверке плана решения на тестовых примерах перед написанием кода. Это позволяет LLM понять логику решения и проверить ее корректность.

**Пошаговая отладка на основе плана**: При возникновении ошибок в коде, система сравнивает фактическое выполнение с ожидаемым поведением из верифицированного плана, что позволяет точно локализовать и исправить ошибки.

**Автономная генерация информации для обратной связи**: Вся дополнительная информация (план решения, верификация, объяснение кода) генерируется самой моделью LLM без необходимости в дополнительном обучении или аннотированных корпусах.

**Значительное улучшение точности генерации кода**: На различных бенчмарках метод демонстрирует существенное повышение точности (Pass@1) по сравнению с существующими подходами, особенно для сложных задач.

## Дополнение:

### Применимость методов исследования в стандартном чате

Исследование LPW (Large Language Model Programming Workflow) представляет методы, которые **не требуют дообучения или специального API** и могут быть полностью реализованы в стандартном чате с LLM.

Ключевые концепции, которые можно применить:

**Двухфазный подход**: Разделение работы на планирование решения и реализацию кода. Пользователь может явно запросить: "Сначала составь пошаговый план решения задачи" "Теперь проверь этот план на следующем тестовом примере..." "Теперь напиши код, основываясь на проверенном плане"

**Верификация плана** перед написанием кода:

Попросить LLM проверить план на конкретных примерах с пошаговым выполнением Запросить анализ промежуточных значений, чтобы убедиться в правильности логики

**Структурированная отладка**:

При ошибках в коде, попросить LLM сравнить фактическое выполнение с ожидаемым поведением из верифицированного плана Запросить анализ расхождений и предложения по исправлению

**Объяснение кода**:

Запрашивать подробные объяснения каждой строки кода для лучшего понимания Ожидаемые результаты от применения этих концепций: - Значительное повышение качества генерируемого кода - Меньшее количество итераций отладки - Лучшее понимание логики решения - Более точная локализация ошибок

Эти подходы особенно полезны для сложных задач программирования, но концепция "план → верификация → реализация → отладка на основе плана" может быть адаптирована практически для любой сложной задачи, где важна точность выполнения.

## Анализ практической применимости: 1. **Двухфазный рабочий процесс для генерации кода** - **Прямая применимость**: Высокая. Пользователи могут адаптировать этот подход в обычных чатах с LLM, сначала запрашивая план решения, затем его проверку на тестовых случаях, и только потом генерацию кода. - **Концептуальная ценность**: Очень высокая. Подход демонстрирует, что разбиение сложной задачи на этапы планирования и реализации значительно повышает качество результата. - **Потенциал для адаптации**: Высокий. Пользователи могут применять эту концепцию не только для программирования, но и для других задач, требующих пошагового планирования и проверки.

**Верификация плана решения Прямая применимость**: Высокая. Пользователи могут запрашивать LLM проверить предложенный план на конкретных примерах перед реализацией. **Концептуальная ценность**: Исключительная. Этот подход демонстрирует важность проверки логики решения до начала его реализации, что помогает избежать ошибок на ранних стадиях. **Потенциал для адаптации**: Высокий. Концепцию можно применять в различных областях, где требуется проверка логики рассуждений.

**Пошаговая отладка на основе плана**

**Прямая применимость**: Средняя. Требует структурированного подхода к отладке, но может быть реализована в стандартном чате. **Концептуальная ценность**: Высокая. Подход показывает, как можно эффективно использовать информацию о желаемом поведении для точной локализации ошибок. **Потенциал для адаптации**: Средний. Требует некоторых технических знаний для применения в других контекстах.

**Автономная генерация информации для обратной связи**

**Прямая применимость**: Высокая. Не требует дополнительных инструментов или обучения. **Концептуальная ценность**: Высокая. Демонстрирует, как можно использовать сами LLM для генерации вспомогательной информации. **Потенциал для адаптации**: Высокий. Подход можно применять в различных задачах, где требуется детальное объяснение или проверка.

**Значительное улучшение точности генерации кода**

**Прямая применимость**: Высокая. Подход непосредственно повышает качество генерируемого кода. **Концептуальная ценность**: Высокая. Показывает, насколько важен структурированный подход к сложным задачам. **Потенциал для адаптации**: Средний. Конкретные улучшения точности специфичны для программирования, но общий принцип применим шире.

# Prompt:

Использование исследования LPW в промптах для GPT ## Ключевые идеи исследования для промптов

Исследование "Программирование, ориентированное на планирование" предлагает двухфазный подход к генерации кода с использованием LLM: 1. **Фаза планирования** - создание и верификация плана решения 2. **Фаза реализации** - написание кода на основе плана и его итеративное улучшение

## Пример промпта на основе методологии LPW

[======] # Задача программирования: [описание задачи]

## Инструкции: Я хочу, чтобы ты решил эту задачу программирования, используя двухфазный подход:

### ФАЗА 1: ПЛАНИРОВАНИЕ РЕШЕНИЯ 1. Проанализируй задачу и создай детальный план решения 2. Определи ключевые алгоритмы и структуры данных 3. Перечисли шаги с ожидаемыми промежуточными результатами 4. Верифицируй план на примерах из условия задачи, "пройдя" через него вручную

### ФАЗА 2: РЕАЛИЗАЦИЯ КОДА 1. Напиши код на [язык программирования] в соответствии с планом 2. Добавь комментарии, объясняющие ключевые части кода 3. Проверь код на тестовых примерах 4. Если найдены ошибки, локализуй их точно и предложи исправления

Примеры для проверки: [Входные данные 1] -> [Ожидаемый результат 1] [Входные данные 2] -> [Ожидаемый результат 2] [======]

## Как работает этот подход

**Улучшение понимания задачи**: Заставляя модель сначала создать и верифицировать план, мы помогаем ей лучше понять суть проблемы до начала кодирования.

**Локализация ошибок**: Сравнивая ожидаемые промежуточные результаты из плана с фактическими результатами кода, модель может точнее определить источник ошибок.

**Структурированное мышление**: Двухфазный подход предотвращает "прыжки к решению" и заставляет модель мыслить более методично.

**Эффективное использование токенов**: Такой подход демонстрирует лучшее соотношение точности к затратам токенов, особенно для сложных задач.

Этот промпт можно адаптировать для различных сценариев программирования, от простых алгоритмических задач до сложных проектов разработки.