Стандартизированный доказательный текст как объект символьных вычислений и источник спецификаций для обучения LLM

И. Б. Казаков*

Рассматривается проблема отсутствия единых стандартов логического изложения математических доказательств. На основе аналогии между математическим текстом и программным кодом, а также принципов «чистого кода» и формальной логики предлагается унифицированный стандарт написания определений, формулировок теорем и доказательств. Обсуждается возможность представления доказательного текста, построенного по предложенным правилам, в виде формального объекта — вывода в грамматике шагов; намечено, как такой объект мог бы синтезироваться символьным решателем и проверяться верификатором. Описание правил стандарта может служить структурированной спецификацией (промптом) для обучения специализированной LLM, генерирующей доказательные наброски под контролем символьной проверки.

Ключевые слова: символьные вычисления, математическая логика, структура доказательства, стандартизация, синтез доказательств, верификация, большие языковые модели, формализация.

Актуальность и цель. Сущностью математической деятельности является проведение доказательств — вывода теорем из аксиом по правилам формальной логики [1]. В программной инженерии код пишется один раз, но читается многократно, поэтому приняты жёсткие стандарты оформления («чистый код») и код-ревью [2]. В математическом сообществе подобные практики формализованы слабо: доказательства часто излагаются в свободном стиле, что затрудняет их проверку и усвоение. Цель работы — разработать унифицированный стандарт математического письма, повышающий логическую прозрачность и верифицируемость текстов, и показать, что он встраивается в парадигму символьных вычислений и может служить спецификацией для обучения специализированных LLM.

* *Казаков Илья Борисович* — к.ф.-м.н., доцент РЭУ им. Г. В. Плеханова; доцент МГТУ им. Н. Э. Баумана, e-mail: i_b_kazakov@mail.ru, ORCID: 0000-0002-3506-7673.

Kazakov Ilia Borisovich — Candidate of Physical and Mathematical Sciences, Associate Professor, Plekhanov Russian University of Economics; Associate Professor, Bauman Moscow State Technical University.

Опубликовано на условиях лицензии CC-BY 4.0
Published under CC-BY 4.0

Предлагаемый стандарт. Стандарт основывается на схеме «Определение — Теорема — Доказательство» и состоит из четырёх частей.

1. *Определения.* Определение вводит новый предикат и состоит из декларации переменных («Пусть...»), названия понятия («Будем говорить, что...») и содержания понятия («...если»). На этапе декларации запрещено накладывать условия на переменные, но допускается зависимая типизация.
2. *Формулировки теорем.* Формулировка подчиняется схеме импликации: декларация переменных («Пусть...»), посылки («Если...») и вывод («То...»).
3. *Доказательства.* Доказательство организуется как иерархическая последовательность нумерованных пунктов-силлогизмов; каждый пункт содержит явные ссылки на посылки и основание вывода. Типы шагов: умозаключение (с указанием типа основания), управление кванторами («Зафиксируем...», «Расфиксируем...», «Выберем...», «Предъявим...»), конструктивные шаги («Положим...») и структурные блоки («Докажем, что...»). Подход адаптирован для доказательства от противного, разбора случаев и индукции.
4. *Связность.* Локальный уровень: определённость терминов; жизненный цикл переменных и временных гипотез (введение при открытии блока, снятие при закрытии); инкапсуляция блоков (наружу экспортируется лишь заявленный вывод — импликация или, для доказательства от противного, отрицание; внутренние пункты и гипотезы снаружи недоступны); обоснованность ссылками на пункты, находящиеся в области видимости на момент выполнения шага; завершённость доказательств. Глобальный уровень: определённость типов, неопределяемые понятия (тип «множество» и отношение принадлежности), аксиомы ZFC как первичные утверждения, запрет циклических ссылок.

Доказательный текст как синтезируемое решение задачи. Обсуждается возможность представления доказательного текста, построенного по предложенным правилам, в виде формального объекта — вывода в грамматике шагов. Линейной последовательности пунктов для такого представления недостаточно: гипотетические рассуждения — доказательство импликаций, доказательство от противного, разбор случаев — по существу требуют вложенных шагов. Блок «Докажем, что...» и есть синтаксическое средство для вложенных шагов: у блока могут быть собственные посылки (их набор может быть, в частности, пустым) и есть

заявленный вывод, а его внутренние пункты недоступны снаружи. Семантически открытию блока соответствует помещение нового контекста (его посылок и переменных) в стек, а закрытию — их снятие; блоки и стек контекстов представляют собой синтаксическую и семантическую стороны одной структуры вложенности. Блок закрывается тогда, когда внутри него доказывается утверждение, тождественное его заявленному выводу. После закрытия блока во внешний контекст экспортируется утверждение $(P_1 \wedge \cdots \wedge P_n) \to C$, где $P_1, \ldots, P_n$ — посылки блока, а C — его вывод. К примеру, для доказательства от противного создается блок с посылкой A и заявленным выводом $\bot$, который экспортирует импликацию $A \to \bot$. Из неё по пропозициональной тавтологии $(A \to \bot) \to \neg A$ в следующем пункте выводится $\neg A$. Намечено, как такой доказательный текст мог бы синтезироваться специализированным решателем и проверяться верификатором, который контролирует ссылки, области видимости переменных и тождественность полученного вывода заявленному. Полная формальная грамматика, правила вывода с условиями применимости и алгоритм верификации выходят за рамки настоящих тезисов и составляют предмет дальнейшей работы.

Пример доказательного текста (теорема Кантора): Пусть X, Y — множества. Если Y является булеаном X, то X не равномощно Y.

Доказательство.

1. Докажем противоречие в предположении, что X равномощно Y.

1.1. Так как X равномощно Y [п.1], то существует функция F из X в Y такая, что F является биекцией. Выберем такую функцию F.

1.2. Положим $D = \{x \in X \mid x \notin F(x)\}$.

1.3. Так как $D \subset X$ [п.1.2], Y — булеан X [по условию], то $D \in Y$.

1.4. Так как $D \in Y$ [п.1.3], $F : X \to Y$ является биекцией [п.1.1], то существует элемент $d \in X$ такой, что $F(d) = D$. Выберем такой элемент $d \in X$.

1.5. Докажем противоречие в предположении, что $d \in D$.

1.5.1. Так как $d \in D$ [п.1.5], $D = \{x \in X \mid x \notin F(x)\}$ [п.1.2], то $d \notin F(d)$.

1.5.2. Так как $d \notin F(d)$ [п.1.5.1], $F(d) = D$ [п.1.4], то $d \notin D$.

1.5.3. Так как $d \in D$ [п.1.5], $d \notin D$ [п.1.5.2], то получено противоречие. Пункт 1.5 доказан.

1.6. Из п.1.5 следует, что выполнено $d \notin D$.

1.7. Так как $d \notin D$ [п.1.6], $F(d) = D$ [п.1.4], то $d \notin F(d)$.

1.8. Так как $d \notin F(d)$ [п.1.7], $D = \{x \in X \mid x \notin F(x)\}$ [п.1.2], то $d \in D$.

1.9. Так как $d \in D$ [п.1.8], $d \notin D$ [п.1.6], то получено противоречие. Пункт 1 доказан.

2. Из п.1 следует, что X не равномощно Y. Теорема доказана.

Стандарт как промпт и спецификация для обучения LLM. Описание правил стандарта — шаблонные обороты, типы шагов, требования к ссылкам и области видимости переменных — образует структурированную спецификацию, пригодную как промпт для LLM, специализированной на составлении доказательств. Такая LLM генерирует не свободный текст, а доказательные наброски в терминах разрешённых шагов стандарта; корректность проверяется внешним символьным верификатором, а его замечания возвращаются модели как обратная связь. Корпус доказательств, оформленных по стандарту, может служить обучающим материалом для дообучения LLM и для генерации с ограничениями грамматики (grammar-constrained decoding). Это совмещает гибкость языковых моделей со строгостью символьной проверки.

Обсуждение и перспективы. Стандарт формулируется в общем виде и допускает применение к классическим примерам из теории множеств, математического анализа и теории чисел: он минимизирует неоднозначность интерпретации, приближает доказательство к формальному коду и облегчает автоматическую верификацию, а стандартизированный текст естественно рассматривается как объект символьных вычислений, синтезируемый решателем и проверяемый верификатором. Дальнейшее развитие связано с созданием инструментов на базе LaTeX для проверки синтаксиса стандарта, реализацией парсера и верификатора доказательных текстов, разработкой решателя, синтезирующего доказательства в терминах стандарта, и обучением LLM, генерирующих доказательные наброски под контролем символьного верификатора. Эти направления соотносятся с существующими пруф-ассистентами, прежде всего Coq [3], и дополняют их промежуточным слоем, в котором доказательство остаётся человекочитаемым, но пригодным для машинной обработки. От универсальных декларативных языков Mizar и Isabelle/Isar стандарт отличается не только тем, что не является входным языком пруф-ассистента и не требует полной формализации в фиксированной логике, но и расстановкой акцентов: в Mizar и Isar управление кванторами — одна из многих конструкций, тогда как стандарт выдвигает его на первый план, ориентируясь на определения и утверждения с чередованием кванторов, каноническим примером которых служат понятие предела и связанные с ним теоремы. Именно здесь явное управление областями видимости переменных даёт наибольший выигрыш — и для педагогических целей, и для машинной проверки.

Список литературы

[1] В. А. Успенский, Н. К. Верещагин, В. Е. Плиско, *Вводный курс математической логики*, Физматлит, Москва, 2004, 128 с.

[2] Р. Мартин, *Чистый код: создание, анализ и рефакторинг*, Питер, Санкт-Петербург, 2019, 464 с.

[3] C. Paulin-Mohring, "Introduction to the Coq proof-assistant for practical software verification", *Tools for Practical Software Verification*, Lecture Notes in Computer Science, **7682**, Springer, Berlin, 2012, 45–95. DOI: 10.1007/978-3-642-35746-6_3.

Статья поступила 29 октября 2026 г.

Standardized Proof Text as a Symbolic Computation Object and a Specification Source for LLM Training

I. B. Kazakov

The paper addresses the lack of unified standards for the logical presentation of mathematical proofs. Drawing an analogy between mathematical text and program code, and building on the principles of Clean Code and formal logic, we propose a unified standard for writing definitions, theorem statements, and proofs. We discuss the possibility of representing a proof text constructed according to the proposed rules as a formal object — a derivation in the grammar of steps — and outline how such an object could be synthesized by a symbolic solver and checked by a verifier. The description of the standard's rules can serve as a structured specification (a prompt) for training a specialized LLM that generates proof sketches under the control of symbolic verification.

Keywords: symbolic computation, mathematical logic, proof structure, standardization, proof synthesis, verification, large language models, formalization.

References

[1] V. A. Uspenski, N. K. Vereshchagin & V. E. Plisko, *Vvodnyj kurs matematicheskoj logiki [An introductory course in mathematical logic]*, FIZMATLIT, Moscow, Russia, 2004 (in Russian), 128 pp.

[2] R. Martin, *Chistyj kod: sozdanie, analiz i refaktoring [Clean Code: A Handbook of Agile Software Craftsmanship]*, Piter, Saint Petersburg, Russia, 2019 (in Russian), 464 pp.

[3] C. Paulin-Mohring, "Introduction to the Coq proof-assistant for practical software verification", *Tools for Practical Software Verification*, Lecture Notes in Computer Science, **7682**, Springer, Berlin, 2012, 45–95. DOI: 10.1007/978-3-642-35746-6_3.

Received on October 29, 2026