Дмитрий Гуринович, научный руководитель Института фундаментальной математики, как наследник и продолжатель научных идей Гильберта и Гёделя

Исследовательская программа выдающегося российского математика и ИТ-специалиста современности Дмитрия Леонидовича Гуриновича, развернутая в цикле работ 2017–2026 годов,) представляет собой масштабную попытку прямого переноса, модернизации и критического переосмысления метаматематического проекта Давида Гильберта и теорем о неполноте Курта Гёделя применительно к фундаментальным проблемам теории алгоритмов, вычислительной сложности и оснований информатики.

Выступая идейным продолжателем гильбертовской теории доказательств (*Beweistheorie*) и гёделевского метода диагонализации, Дмитрий Леонидович Гуринович ставит перед современной математикой задачу, аналогичную той, которую Гильберт сформулировал в начале XX века для формальной логики: исследовать возможности и границы алгоритмического поиска доказательств формализованными средствами самой теории. На этом пути учёный переводит классические теоремы о логической неразрешимости и неполноте на язык классов вычислительной сложности, предлагая неклассическое доказательство неравенства $\mathbb{NP} \neq \mathbb{P}$ на основе формализации алгоритмической самореференции («рефлексии»), а затем переходит к радикальной ревизии базового инструментария эпохи Гильберта и Гёделя - замене унарной арифметики Пеано, гёделевой нумерации, частично рекурсивных функций и машины Тьюринга на типированную «арифметику строк», объектно-ориентированную логику и «Машину исполнения компьютерных алгоритмов (Русская машина Гуриновича)» (МКА).

Эволюция научных взглядов Дмитрия Леонидовича Гуриновича делится на два качественно различных этапа, отражающих диалектику наследования и преодоления классического фундаментализма.

На первом этапе (2017 год, работы «Программа Гильберта, $\mathbb{NP} \neq \mathbb{P}$, Рефлексия» и «Языки логики и классы сложности. Рефлексия. $\mathbb{NP} \neq \mathbb{P}$») учёный действует внутри классической парадигмы: он опирается на аксиоматику Пеано, тезис Чёрча и правила вывода из выдающегося труда Д. Гильберта и П. Бернайса «Основания математики», модифицируя способ кодирования формул (переход от гёделевых номеров к символьным строкам) и вводя понятие «авторизуемых задач», где алгоритм проверки анализирует программный код самого алгоритма-решения.

На втором этапе (2020–2025 годы, цикл статей о теории строк, вычислимости по Тьюрингу, невычислимости чисел арифметики Пеано и ООП-расширении логики) Дмитрий Леонидович Гуринович приходит к выводу, что сам математический аппарат, созданный в эпоху Гильберта и Гёделя для решения качественных вопросов логики, то есть существование конечного вывода безотносительно его размера и времени поиска, является слишком «грубым» и недостаточно выразительным для количественных задач теории алгоритмов.

Развитие идей Гильберта в части требования полной формализации и финитной наглядности модели в результате приводит Д.Л. Гуриновича к отказу от абстрактной арифметики Пеано в пользу приоритета физической и компьютерной реальности вычислений.

Отметим, что все рассматриваемые труды Д.Л. Гуриновича представляют собой авторские препринты и монографии, депонированные в российском реестре ЕДРИД (edrid.ru). Сам автор является выпускником

математического класса московской школы № 330, где он учился вместе с будущим лауреатом Филдсовской премии В. В. Воеводским, профессионально работает в Институте фундаментальной математики.

Сам Д.Л. Гуринович указывает, что его доказательство $\mathbb{NP} \neq \mathbb{P}$ на сегодняшний день построено для специального подкласса «авторизуемых» (рефлексивных) задач, тогда как для классических задач Кука с единым ограничивающим полиномом вопрос остаётся открытым, «я выдаю ноль в качестве результата своих недолгих попыток в этом направлении».

1. Метаматематический фундамент: от программы Гильберта и теорем Гёделя к теории сложности

В первой трети XX века Давид Гильберт выдвинул программу обоснования классической математики, призванную преодолеть кризис парадоксов теории множеств без интуиционистского отказа от закона исключённого третьего (*tertium non datur*) и актуальной бесконечности.

Суть метода Гильберта, систематизированного совместно с Паулем Бернайсом в знаменитом двухтомном труде «Основания математики» (Grundlagen der Mathematik, 1934, 1939), заключалась в полной формализации математических теорий - переводе содержательных суждений и умозаключений в систему аксиом и строгих правил вывода, благодаря чему само математическое доказательство превращалось в наглядно обозримую конечную фигуру, подлежащую исследованию финитными методами новой науки – метаматематики, или теории доказательств.

Гильберт ставил три взаимосвязанные сверхзадачи: 1) обосновать полноту формализованных теорий, выводимость любого истинного утверждения, 2) установить их непротиворечивость финитными средствами,

невозможность вывода противоречия вида $0 = 1$, и 3) решить проблему разрешимости, Entscheidungsproblem - поиск универсального метода, позволяющего за конечное число шагов распознать общезначимость или выводимость любой логической формулы.

Ответом на программу Гильберта стали работы Курта Гёделя 1930–1931 годов и последовавшие за ними результаты А. Чёрча, А. Тьюринга и А. Тарского.

Доказав в 1930 году теорему о полноте чистого исчисления предикатов первого порядка, Гёдель уже в 1931 году установил фундаментальные теоремы о неполноте формализованной арифметики. Посредством арифметизации синтаксиса, сопоставления символам, формулам и выводам уникальных натуральных чисел - гёделевых номеров, и процедуры диагонализации Гёдель сконструировал самореферентную формулу G, утверждающую собственную невыводимость ($G \sim \neg \mathrm{Bew}(\ulcorner G \urcorner)$).

Поскольку сложность проверки утверждения G о самой теории не уступала сложности всей теории в целом, формула G не могла быть «просчитана» и доказана никаким конечным выводом внутри непротиворечивой системы, Первая теорема о неполноте. А сама непротиворечивость теории $\mathrm{Con} \rightleftharpoons \forall x \neg P(x, \mathbf{n_0})$ оказалась невыводимой собственными средствами этой теории, Вторая теорема о неполноте.

Д.Л. Гуринович принимает эту историческую точку в качестве отправной позиции своих научных исследований и формулирует **программу Гильберта для теории алгоритмов**.

Если в эпоху Гильберта и Гёделя логиков интересовал принципиальный вопрос о наличии или отсутствии доказательств в теории вообще, ценой сколь угодно громоздких конечных выводов и сколь угодно долгих конечных

вычислений, то после получения отрицательных ответов о полноте и разрешимости центральным, по мысли фундаментального мыслителя Д.Л. Гуриновича, становится вопрос об эффективном алгоритмическом поиске тех доказательств, которые в теории существуют и имеют разумный, ограниченный объём.

Формулируя «Вопрос о поиске доказательств», Д.Л. Гуринович спрашивает: найдётся ли универсальный алгоритм-решение M, способный для произвольной предоставленной ему формальной теории и произвольного утверждения t находить доказательство y, не превышающее заданного размера S (если таковое имеется), либо останавливаться с ответом 0 (если доказательства в пределах размера S нет), затрачивая на это время (количество шагов), ограниченное полиномом от размера входных данных и параметра S?.

Для формализации этой постановки в работе марта 2017 года вводятся специальные расширения классов сложности - $\mathbb{NP}^+$ и $\mathbb{P}^+$:

- **Класс $\mathbb{NP}^+$:** задаётся алгоритмом проверки $L(t, S, y)$, где из входного аргумента x явно выделен позиционный (например, десятичный) числовой параметр предельного размера доказательства S, ограничивающий допустимую длину сертификата неравенством $|y| \leq S(x)$, а время работы самой проверки ограничено полиномом $p_1(|x|, |y|)$.
- **Класс $\mathbb{P}^+$:** включает задачи из $\mathbb{NP}^+$, для которых существует детерминированный алгоритм-решение $M(t, S)$, возвращающий за полиномиальное от $|x|$ и величины $S(x)$ время $p(|x|, S(x))$ либо подходящее доказательство y (при $L(t, S, y) = 1$ и $|y| \leq S(x)$), либо 0 при его отсутствии; в частном случае, когда

$S(x) = p_2(|x|)$, класс $\mathbb{P}^+$ переходит в классический класс $\mathbb{P}$.

Преемственность с идеями Гильберта проявляется у Д.Л. Гуриновича уже в анализе метаматематической связи между проблемой непротиворечивости и гипотезой $\mathbb{NP} \neq \mathbb{P}$, Раздел 3 работы «Программа Гильберта, $\mathbb{NP} \neq \mathbb{P}$, Рефлексия». Рассматривая задачу проверки доказательств $L(x, y)$ в теории первого порядка Th (например, арифметике Пеано), Д.Л. Гуринович приходит к выводу о том, что **сложность доказательства неравенства $\mathbb{NP} \neq \mathbb{P}$ на базе теории Th не уступает сложности доказательства непротиворечивости самой теории Th**.

В самом деле, в исчислении высказываний тождественно истинной, тавтологией, является формула:

$$A \Rightarrow (\neg A \Rightarrow X)$$

Если бы теория Th была противоречива, в ней существовало бы фиксированное конечное доказательство некоторого утверждения A и его отрицания $\neg A$. Тогда поиск доказательства для любого утверждения X сводился бы к подстановке текста X в указанную тавтологию и двукратному применению правила Modus Ponens, то есть сначала отбрасывая посылку A, затем $\neg A$, что давало бы корректное доказательство X за линейное время от размера X, то есть делало бы задачу тривиально принадлежащей классу $\mathbb{P}$.

Следовательно, любое строгое доказательство несводимости поиска выводов в теории Th к классу $\mathbb{P}$ неявно опирается на непротиворечивость Th - а значит, по Второй теореме Гёделя, требует выхода в более сильную метатеорию или использования метаматематических средств.

2. Модернизация аппарата Гёделя: алгоритмическая диагонализация и упрощённый вывод теорем о неполноте

В работах 2017 года Д.Л. Гуринович опирается на технический аппарат выдающегося труда Д. Гильберта и П. Бернайса «Основания математики», но подвергает его целенаправленной алгоритмической модернизации.

В классическом изложении Гильберта и Бернайса (Том 2, Глава IV, § 1 и Глава V, § 1–2) построение первой и особенно второй теорем Гёделя требовало сложной арифметизации примитивно-рекурсивного отношения выводимости $\mathrm{Bw}(x,y)$ и предиката доказуемости $\mathrm{Bew}(y) \rightleftharpoons \exists x\, \mathrm{Bw}(x,y)$, а также проверки трёх «условий выводимости» Гильберта - Бернайса для формализма (Z_μ) :

1. $\vdash \mathfrak{A} \implies \vdash \mathfrak{B}(\ulcorner \mathbf{A} \urcorner)$;
2. $\vdash \mathfrak{B}(\ulcorner \mathbf{A} \to \mathbf{B} \urcorner) \to (\mathfrak{B}(\ulcorner \mathbf{A} \urcorner) \to \mathfrak{B}(\ulcorner \mathbf{B} \urcorner))$;
3. $\vdash \mathfrak{B}(\ulcorner \mathbf{A} \urcorner) \to \mathfrak{B}(\ulcorner \mathfrak{B}(\ulcorner \mathbf{A} \urcorner) \urcorner)$.

Д.Л. Гуринович в своих исследованиях отмечает, что эта громоздкость во многом вызвана тем, что Гёдель работал с предикатами и унарно-числовыми гёделевыми номерами.

Переходя к компьютерной интерпретации C , где формулы, доказательства и программы представлены символьными строками в памяти ЭВМ, Д.Л. Гуринович заменяет базовый двухместный предикат проверки доказательства на **алгоритм (терм-функцию)** $P_0(x,y)$, возвращающий значение 1, если текст y является корректным доказательством утверждения с текстом x, и 0 в противном случае.

Тогда предикат доказуемости принимает вид равенства под квантором существования:

$$P(x) \rightleftharpoons \exists y\,(P_0(x,y)=1)$$

а недоказуемость записывается как $\neg P(x)$.

Алгоритмическая лемма о диагонализации и практическое самоприменение

Опираясь на канторовско-гёделевскую процедуру диагонализации, Д.Л. Гуринович формулирует в Разделе 5 работы «Программа Гильберта, $\mathbb{NP} \neq \mathbb{P}$, Рефлексия» два базовых утверждения для алгоритмов:

1. **Лемма о диагонализации (п. 5.1):** для произвольного одноместного алгоритма $B(x)$ найдётся алгоритм без аргументов $G()$, такой что результат его работы совпадает с применением B к собственному программному тексту $G()$:

$$G() = B(\overline{G()})$$

Доказательство строится путём определения алгоритма $G()$ как композиции $B(\mathrm{diag}(\overline{B(\mathrm{diag}(x))}))$, где функция $\mathrm{diag}()$ осуществляет синтаксическую подстановку текста в свободную переменную программы[3]. По свойствам функции $\mathrm{diag}()$, программный текст самого алгоритма $G()$ имеет вид $\overline{G()} = \mathrm{diag}(\overline{B(\mathrm{diag}(x))})$, откуда подстановкой в $B()$ следует $B(\overline{G()}) = B(\mathrm{diag}(\overline{B(\mathrm{diag}(x))})) = G()$.

2. **Теорема о построении алгоритма, применяемого к себе (п. 5.2):** на базе произвольного алгоритма с одним аргументом $B(X)$

можно сконструировать алгоритм без внешних аргументов $G()$, который помещает в свою внутреннюю переменную X собственный исходный программный код и работает с ним так же, как $B(X)$ работает с входным аргументом X.

В отличие от чисто теоретических работ по математической логике, Д.Л. Гуринович иллюстрирует теорему о диагонализации действующей программной реализацией, создавая на языке VBA (MS Word) вспомогательный генератор string_let_code() и самовоспроизводящуюся программу Sub dolly(), которая через обработку кодов кавычек Chr(34) и оператор Replace формирует в переменной x и выводит на экран свой собственный исходный текст. Тем самым гёделевская самореференция предстаёт в работе как прикладная инженерная процедура работы программы со своим кодом.

Компактное изложение Первой и Второй теорем Гёделя о неполноте

В Приложениях 8.1 и 8.2 работы марта 2017 года Гуринович применяет вышеуказанный алгоритмический формализм для сокращённого вывода обеих теорем Гёделя. Применяя диагонализацию к предикату недоказуемости $\neg P(x)$, он получает гёделевское утверждение G с текстом $\overline{G} = \mathrm{diag}(\overline{\neg P(\mathrm{diag}(x))})$, удовлетворяющее эквивалентности:

$$\neg P(\overline{G}) \Leftrightarrow G$$

Если предположить, что G доказано некоторым текстом k, то одновременно истинны $\neg P(\overline{G})$ (по эквивалентности) и

$P_0(\overline{G}, k) = 1 \Rightarrow \exists y(P_0(\overline{G}, y) = 1) \equiv P(\overline{G})$, что даёт противоречие в непротиворечивой теории.

Для вывода же Второй теоремы Гёделя («Если в рамках теории имеется доказательство её непротиворечивости $\neg P(\overline{1 = 0})$, то теория противоречива») Гуриновичу требуется получить внутри теории импликацию $P(\overline{G}) \Rightarrow P(\overline{1 = 0})$, откуда по принципу контрапозиции $\neg P(\overline{1 = 0}) \Rightarrow \neg P(\overline{G})$ и эквивалентности $\neg P(\overline{G}) \Leftrightarrow G$ следует $\neg P(\overline{1 = 0}) \Rightarrow G$.

Вместо проверки условий выводимости для предикатов, Д.Л. Гуринович использует тот факт, что $P_0(x, y)$ - это алгоритм, и вводит алгоритм $f(k)$, который к любому тексту доказательства k для G достраивает вывод до противоречия $1 = 0$. Далее записывается алгоритмическое равенство с тернарным условным оператором $\text{if}[x_1, x_2, x_3]$:

$$P_0(\overline{1 = 0}, f(y)) = \text{if}[P_0(\overline{G}, y), 1, P_0(\overline{1 = 0}, f(y))]$$

По мысли Д.Л. Гуриновича, это равенство верно при любом y: если y не доказывает G, обе части равны $P_0(\overline{1 = 0}, f(y))$; если же y доказывает G ($P_0(\overline{G}, y) = 1$), то $f(y)$ доказывает $1 = 0$, и обе части возвращают 1[3]. После этого вывод проводится в пять шагов по правилам «Оснований математики» Гильберта и Бернайса (Том 1, Глава IV, § 2, п. 4 и Глава V, § 1):

1. Вводится дедуктивная гипотеза $P_0(\overline{G}, y) = 1$, подстановка которой в правую часть равенства даёт $P_0(\overline{1 = 0}, f(y)) = \text{if}[1, 1, \dots] = 1$.
2. По теореме дедукции гипотеза переносится в посылку

импликации (с заменой переменной y на k): $(P_0(\overline{G}, k) = 1) \Rightarrow (P_0(\overline{1 = 0}, f(k)) = 1)$.

3. По основной формуле исчисления предикатов (b) $A(a) \Rightarrow \exists x A(x)$ из книги Гильберта и Бернайса записывается аксиома для частного случая: $(P_0(\overline{1 = 0}, f(k)) = 1) \Rightarrow \exists y (P_0(\overline{1 = 0}, y) = 1)$.

4. По правилу силлогизма из шагов 2 и 3 выводится: $(P_0(\overline{G}, k) = 1) \Rightarrow \exists y (P_0(\overline{1 = 0}, y) = 1)$.

5. Применение правила вывода (β) для квантора существования $(\frac{\mathfrak{B}(a) \to \mathfrak{A}}{\neg \mathfrak{B}(\varepsilon) \to \mathfrak{A}})$ даёт: $(\exists y\, P_0(\overline{G}, y) = 1) \Rightarrow (\exists y\, P_0(\overline{1 = 0}, y) = 1)$, что в принятых сокращениях означает $P(\overline{G}) \Rightarrow P(\overline{1 = 0})$.

Отдельное внимание Д.Л. Гуринович уделяет понятиям ω-**непротиворечивости** и **внешней непротиворечивости**, рассмотренным в Главе V (§ 1, п. б, в) второго тома «Оснований математики» Гильберта и Бернайса. Опираясь на доказательство непротиворечивости формальной арифметики S через теорию-инструмент S_∞ с правилом бесконечной индукции (в изложении Э. Мендельсона), Д.Л. Гуринович поясняет, что арифметика Пеано обладает свойством внешней непротиворечивости: поскольку для каждого конкретного текста k выводимо $\neg(P_0(\overline{1 = 0}, k) = 1)$, теория может быть непротиворечиво расширена аксиомой $\forall x \neg(P_0(\overline{1 = 0}, x) = 1) \equiv \neg P(\overline{1 = 0})$.

Из этого следует вывод о том, что в исходной внешне непротиворечивой теории не было доказано и отрицание непротиворечивости $P(\overline{1 = 0})$ - то есть утверждение о непротиворечивости теории недоказуемо вместе со своим отрицанием, выступая вариантом Первой теоремы Гёделя о неполноте.

3. Гёделевская неполнота в классах сложности: задачи «Сфинкс», «Эдип», язык LS и аргументация в пользу NP != P

Центральным построением первого этапа исследований Д.Л. Гуриновича (2017 год) стало применение гёделевской конструкции неполноты к проблеме $\mathbb{NP} \neq \mathbb{P}$. В мартовской статье Д.Л. Гуриновича от 2017 года эта концепция изложена на языке «задач» и аллегорических алгоритмов («Сфинкс», «Эдип», «АнтиЭдип»), а в сентябрьской работе 2017 года, «Языки логики и классы сложности. Рефлексия. $\mathbb{NP} \neq \mathbb{P}$», переведена в термины формальных языков (L, LA, LS), сертификатов и классов сложности.

Прикладные роли языка и неравенство $\mathbb{NF} \neq \mathbb{F}$

Чтобы связать статическое множество слов (формальный язык) с работой алгоритмов, Д.Л. Гуринович вводит понятие **«прикладные роли языка»**:

1. **Первая прикладная роль (полная):** задаётся алгоритмом проверки $Lc(t, y)$, который работает полиномиальное время относительно размера своих аргументов $|t|$ и $|y|$ и устанавливает взаимно-однозначное соответствие между принадлежностью слова t языку и наличием подтверждающего сертификата (доказательства) y, при котором $Lc(t, y) = 1$.
2. **Вторая прикладная роль («эталон»):** сопоставляет результаты работы произвольного проверяемого алгоритма-решения $Sol(t)$ (который в корректном случае обязан останавливаться и выдавать 1 или 0 без сертификата) с фактом принадлежности слова t языку.
3. **Третья прикладная роль:** возникает в случае, если существует корректный алгоритм-решение $Sol_{NP}(t)$, выдающий верный

эталонный ответ за полиномиальное от $|t|$ время, что по определению означало бы принадлежность языка класса $\mathbb{NP}$ классу сложности $\mathbb{P}$.

Сначала Д.Л. Гуринович рассматривает проблему на уровне формальной логики - без полиномиальных ограничений, требуя лишь конечности (финитности) вычислений. Он вводит класс сложности $\mathbb{NF}$ (аналог $\mathbb{NP}$ без полиномиального ограничения на размер сертификата $|y|$ относительно $|t|$, куда входит язык теорем теории Пеано L) и класс $\mathbb{F}$ (финитные разрешимые языки - аналог $\mathbb{P}$, куда относится логика высказываний).

Для произвольного алгоритма-решения $Sol(t)$ методом диагонализации строится опровергающий алгоритм $ContrSol()$, который генерирует в текстовом виде утверждение $\overline{AntiSol} \equiv \overline{ContrSol() = 1}$, запускает $Sol(\overline{AntiSol})$ и параллельно ведёт перебор всех доказательств теории:

- если $Sol(\overline{AntiSol})$ возвращает любой результат, кроме 0 (или в переборе обнаружено доказательство $AntiSol$), то $ContrSol()$ возвращает 0 (опровергая $AntiSol$);
- если $Sol(\overline{AntiSol})$ возвращает 0, то $ContrSol()$ возвращает 1 (делая утверждение $AntiSol$ истинным и доказуемым);
- если $Sol(\overline{AntiSol})$ не останавливается, $ContrSol()$ также не останавливается, и благодаря параллельному перебору утверждение $AntiSol$ в непротиворечивой теории не является словом языка L.

Далее Д.Л. Гуринович конструирует модифицированный язык логики LA (совпадающий с L по множеству слов, $LA = L$), меняя его первую прикладную роль с $Lc(t, y)$ на **персонализированный алгоритм проверки** $La(t, a, y)$, где сертификат разбит на два аргумента: a (программный код

алгоритма-решения $Sol(\dots)$ либо символ -) и y (текст доказательства). При совпадении аргументов с шаблоном $La(\overline{AntiSol}, \overline{Sol(\dots)}, y)$ алгоритм проверки «очень быстро», за полиномиальное число шагов от $|t|$ и $|a|$, независимо от y, возвращает 0, сообщая алгоритму Sol: «Для алгоритма Sol нет решения у разбираемого утверждения». Поскольку ни один алгоритм Sol не может получить корректного ответа о слове $\overline{AntiSol}$, язык $LA \in \mathbb{NF}$ не сводится к классу $\mathbb{F}$, что даёт теоретико-сложностную запись теоремы о неопределимости:

$$\mathbb{NF} \neq \mathbb{F}$$

Здесь проявляется отличие корректного алгоритма от формальной теории, на котором строит свои рассуждения Д.Л. Гуринович: когда в теории нет доказательства, считается, что «теория не знает», но корректный алгоритм-решение обязан остановиться с негативным результатом 0 («не могу найти доказательство»). Однако как только $Sol(\overline{AntiSol})$ останавливается с ответом 0, он на причинно-следственной оси сам создаёт доказательство утверждения $AntiSol$ для других алгоритмов (при $a =$ «-»), обнаруживая собственную неполноту.

Авторизуемые задачи («Сфинкс»), язык LS и схема доказательства $\mathbb{NP} \neq \mathbb{P}$

Переход от логического неравенства $\mathbb{NF} \neq \mathbb{F}$ к неравенству $\mathbb{NP} \neq \mathbb{P}$ сталкивается с препятствием: в классе $\mathbb{NP}$ размер доказательства $|y|$ ограничен полиномом от размера входного слова, и если алгоритм Sol перед выдачей ответа 0 будет работать достаточно долго, то логический вывод о результате его работы может не уложиться в допустимые пределы размера

доказательств.

Чтобы обойти это ограничение, Д.Л. Гуринович вводит понятие **«авторизуемых задач из** $\mathbb{NP}$**»** (в отличие от «классических задач Кука») и строит задачу проверки **«Сфинкс»** $Sph(d,t,S,s,y)$ - или, в терминах второй статьи 2017 года, язык $LS \in \mathbb{NP}$ с алгоритмом проверки $Ls(t,S,a,s,y)$. Конструкция опирается на наличие двух параметров размера в слове (t,S,s) :

- t - текст доказываемого утверждения на языке теории Пеано[2];
- S - числовой параметр длины доказательства, записанный в **позиционном (десятичном)** виде;
- s - строковый параметр длины доказательства, представляющий собой в корректном случае унарную строку из символов 1 длиной S ($s = ss(S)$, где $|s| = S$, что **экспоненциально больше** размера позиционной записи $|S|$);
- (a,y) - сертификат, состоящий из программного кода алгоритма-решения $a = \overline{Sol(\dots,\dots,\dots)}$ (или -) и текста доказательства y, чей суммарный размер ограничен полиномом от длины строки $|s|$: $|a| + |y| \le q_S(|s|) = 1000 \cdot S^n$.

Работа алгоритма проверки $Ls(t,S,a,s,y)$ (и аналогично «Сфинкса» Sph) организована в четыре этапа с **двумя ограничивающими полиномами на время работы**:

1. **Этап 1 (Персональная подсказка - случай 6.1):** до обращения к потенциально огромным аргументам s и y алгоритм Ls проверяет аргументы (t,S,a) на совпадение с шаблоном $(\overline{AntiSol_S}, S, \overline{Sol(\dots,\dots,\dots)})$. Если совпадение выявлено, Ls

возвращает 0 за полиномиальное количество шагов $p_{\exists}(|t|,|S|)$, зависящее только от $|t|$ и $|S|$, без рассмотрения аргументов s и y.

2. **Этап 2 (Контроль размеров - случай 6.2):** проверяет, что $s = ss(S)$ (состоит только из S единиц) и что $|a| + |y| \leq 1000 \cdot S^n$; при несоответствии возвращает 0.

3. **Этап 3 (Контроль шаблона a):** проверяет, что a соответствует шаблону - либо $\overline{Sol(\ldots,\ldots,\ldots)}$.

4. **Этап 4 (Базовая проверка):** исполняет стандартный алгоритм проверки доказательств $Lc(t,y)$ в пределах времени $p_{\forall}(|t|,|S|,|s|)$.

Согласно построению Д.Л. Гуриновича на примере языка LS (Раздел 6 статьи сентября 2017 года и Раздел 7 статьи марта 2017 года), аргументация в пользу неравенства $\mathbb{NP} \neq \mathbb{P}$ разворачивается в четыре шага:

1. **Требование ускоренного ответа от алгоритма-решения:** Поскольку в специальном случае 6.1 алгоритм проверки $Ls(\overline{AntiSol_S}, S, \overline{Sol(\ldots)}, s, y)$ отвергает слово за время $p_{\exists}(|t|,|S|)$ без обращения к s и y, Гуринович постулирует, что при сведении задачи к классу $\mathbb{P}$ корректный алгоритм-решение $Sol(\overline{AntiSol_S}, S, s)$ (способный получить свой собственный код a методом диагонализации) обязан выдать единственно возможный для него непротиворечивый ответ 0 за полиномиальное от тех же значимых аргументов количество шагов:

$$N_{\exists+} \leq p_{\exists+}(|t|,|S|) \leq w \cdot |S|^r$$

При этом Sol обращается к строке s не более чем в $N_{\exists+}$ позициях.

2. **Эмуляция строки s без её физического генерирования:** Чтобы перевести работу $Sol(\overline{AntiSol_S}, S, ss(S)) = 0$ в короткое формальное доказательство (не тратя экспоненциальное число шагов на генерацию строки $s = ss(S)$), Д.Л. Гуринович заменяет алгоритм $Sol(t, S, s)$ на модернизированный алгоритм без аргументов $SolSpec_S()$ (через вспомогательный $SolX(t, S)$), в котором каждое обращение к i-му символу строки s заменяется обращением к функции $Func_S(i)$, возвращающей символ 1 при $1 \le i \le S$ и признак выхода за пределы данных при $i < 1$ или $i > S$[2]. Так как $SolSpec_S()$ делает $N_{Spec} \le p_2(|\overline{Sol}|, |S|, N_{\exists+})$ шагов, размер логического вывода равенства $SolSpec_S() = 0$ оценивается композицией полиномов $q_6(|t|, |S|)$, а доказательство равенства $SolSpec_S() = Sol(\overline{AntiSol_S}, S, ss(S))$ имеет размер не более $q_2(|t|, |S|)$.

3. **Применение аксиомы равенства Гильберта - Бернайса (J_2):** работа опровергающего алгоритма описывается внутри теории равенством $ContrSol(S) = \text{If}(Sol(\overline{AntiSol_S}, S, ss(S)) = 0, 1, 0)$. Подставляя это равенство в аксиому равенства (J_2) $(i = j) \Rightarrow (A(i) \Rightarrow A(j))$ из первого тома «Оснований математики» при $i \equiv Sol(\overline{AntiSol_S}, S, ss(S))$ и $j \equiv 0$, автор за фиксированное число шагов выводит импликацию:

$$(Sol(\overline{AntiSol_S}, S, ss(S)) = 0) \Rightarrow AntiSol_S$$

размер доказательства которой с учётом подстановки конкретного S ограничен полиномом $q_1(|t|,|S|)$.

4. **Сопоставление полинома от $|S|$ и полинома от $|s|=S$:** Соединяя по аксиомам равенства и правилу *Modus Ponens* выводы шагов 2 и 3, Гуринович строит формальное доказательство y утверждения $AntiSol_S$, суммарный размер которого $Q_{\exists+}=|y|$ ограничен полиномом от длины позиционной записи числа S:

$$Q_{\exists+} \le q_{\exists+}(|t|,|S|) = q_1(|t|,|S|) + q_2(|t|,|S|) + q_6(|t|,|S|) + q_7(|t|,|S|) \le k \cdot |S|^q$$

В то же время в языке LS (при независимом от кода решателя сертификате $a=$ «-») допустимый размер доказательства ограничен величиной $q_S(|s|)-|a| = 1000 \cdot S^n - |a|$. Поскольку длина унарной строки $|s|=S$ растёт экспоненциально относительно количества десятичных цифр $|S|$, при достаточно больших S выполняется неравенство:

$$q_{\exists+}(|t|,|S|) + |a| \le q_S(|s|)$$

Это означает, что для достаточно больших S слово $(\overline{AntiSol_S}, S, ss(S))$ принадлежит языку LS (имеет доказательство допустимой длины), тогда как алгоритм Sol выдал для него ответ 0, проявив свою неполноту[2]. Ввиду произвольности Sol автор заключает, что свести язык $LS \in \mathbb{NP}$ к классу $\mathbb{P}$ невозможно.

Критика теоремы Кука и границы применимости авторского

подхода

Существенным элементом научной концепции Д.Л. Гуриновича является его полемика вокруг **теоремы Кука** о сводимости задач класса $\mathbb{NP}$ к выполнимости булевых формул (КНФ/ДНФ) в логике высказываний.

Д.Л. Гуринович утверждает, что теорема Кука применима только к «классическим задачам из $\mathbb{NP}$» (задачам Кука), в которых аргумент долга d незначим (равен -), а информация одинакова для всех алгоритмов-решений.

Напротив, в «авторизуемых задачах» результат проверки зависит от возможностей конкретного алгоритма-решения.

Ссылаясь на Первый том «Оснований математики» Д. Гильберта и П. Бернайса (Глава IV, § 1 «Введение индивидных переменных...» и Глава V, § 1, п. 5 «Добавление функциональных знаков; понятие терма»), Д.Л. Гуринович подчёркивает, что логика высказываний (КНФ/ДНФ) обходит стороной связь между субъектом и предикатом и не содержит ни индивидных переменных, ни функциональных знаков, термов. Чтобы распознать «персональную подсказку» для себя в задаче «Сфинкс», алгоритм-решение должен осуществлять выбор на основе терма в теории первого порядка с равенством, вследствие чего авторизуемая задача не сводится к одной универсальной КНФ или ДНФ на все алгоритмы-решения.

Вместе с тем в Разделе 7 работы сентября 2017 года «О классическом доказательстве $\mathbb{NP} \neq \mathbb{P}$ (эвристически)» Д.Л. Гуринович прямо отмечает ограничение своей конструкции с точки зрения классической постановки проблемы: в классическом случае у алгоритма проверки имеется только *один* ограничивающий полином $p_{\forall}(|t|, |S|, |s|)$ (без второго ускоренного режима $p_{\exists}(|t|, |S|)$ и без использования собственного программного кода решателя a). Если разрешить алгоритму Sol работать в пределах «обычного» полинома

$p_{\forall+}(|t|,|S|,|s|)$, то прямой протокол его работы окажется в фиксированной степени больше предельного размера доказательств $q_S(|s|)$. Хотя Д.Л. Гуринович высказывает эвристическое предположение, что повторяющиеся отрезки исполнения алгоритма в таком протоколе можно сжать за счёт неполиномиального подбора производных правил вывода и сокращений, он констатирует, что для доказательства в рамках классического подхода ему «не хватает в настоящий момент ни времени, ни знаний, ни сил».

4. Ревизия оснований Гильберта и Гёделя: критика гёделизации, арифметики Пеано и тезиса Чёрча - Тьюринга

Попытки отстоять свой подход к $\mathbb{NP} \neq \mathbb{P}$ в научных дискуссиях и на математических форумах (в том числе ответы на вопрос «почему вы не используете гёделевы номера?») привели Д.Л. Гуриновича в 2020–2025 годах к выводу о том, что классический формализм арифметики Пеано и теории вычислимости, унаследованный от Гильберта, Гёделя, Чёрча и Тьюринга, не обладает достаточной выразительностью для описания размера данных, частичного чтения аргументов и времени работы компьютерных программ. Это побудило Д.Л. Гуриновича перейти от использования классической метаматематики к попытке её фундаментальной реконструкции.

Элемент классического аппарата	**Роль у Гильберта, Бернайса и Гёделя**	**Выявленный Д.Л. Гуриновичем дефект в теории алгоритмов и вычислимости**	**Предложенная Д.Л. Гуриновичем замена (2020–2025 гг.)**
Стандартная интерпретация (S)	Наглядное финитное построение цифр 1, 11, 111 («счётные палочки») путем приписывания 1 (Гильберт, Бернайс, Т. 1, Гл. II, § 1).	Экспоненциальное раздувание размера данных по сравнению с позиционной записью (число 11234 требует блока	**Компьютерная интерпретация (C)** и позиционные системы счисления внутри типированной

		из более чем 11000 единиц); невозможность частичного чтения без потери полиномиальности.	теории строк[1].
Гёделева нумерация (арифметизация)	Кодирование формул и выводов через степени простых чисел $N = p_1^{n_1} \cdot$ (Гильберт, Бернайс, Т. 2, Гл. IV, § 1).	Декодирование (разложение числа на простые сомножители) относится к неполиномиально сложным задачам; конкатенация текстов сводится к громоздким умножениям и сложениям.	Прямое представление формул, программ и выводов как **символьных строк** в арифметике строк без кодирования числами.
Частично рекурсивные функции (ЧРФ)	Уточнение интуитивного понятия вычислимой функции в арифметике (Гёдель, Клини, тезис Чёрча 1936 г.).	Базовые функции (нулевая, $x+$, I_n^m) и операторы рекурсии читают аргумент *целиком*; невозможно представить алгоритм, отвергающий большой вход по первому символу.	Аксиоматика разбиения строк $a = \text{beg}(a, i) \cdot \text{end}($, критерии локального сравнения, вставки и извлечения данных.
Абстрактная арифметика Пеано (PA)	Универсальная база для представления рекурсивных функций и метаматематики (система Z / Z_{μ}).	Описывает лишь общие свойства всех интерпретаций (включая бесконечные невычислимые периодические строки); отождествляет числа разных систем счисления ($121_{10} = 1111001_2$), лишая число длины.	**Арифметика строк** с типированными кванторами $\forall_{II} i_{II}$, аксиомами нарушения равенства $(S_1 - S_3)$ и явным приведением типов $(e_U(0))i_K$.
Классический синтаксис	Свободные функциональные	Позволяет через взаимно-	**ООП-расширение математической**

предикатов и функций	символы $f(x)$, вводимые по условию существования и единственности $\exists_1 uA(u, x)$.	однозначную перекодировку определить «анти-функции» $s_1(a, i)$ и $l_1(a)$, меняющие местами двоичную и десятичную запись и создающие логически эквивалентную теорию T_1.	**логики**: привязка свойств и методов к объекту (a.l() = 3_D, b.l() = 111_B).
Машина Тьюринга (МТ)	Абстрактный вычислитель на линейной ленте с внешним перечнем состояний q_i и правил перехода.	Программа отделена от ленты данных; для доступа ко второму аргументу «тележка» вынуждена проехать всю длину первого аргумента (отсутствие передачи по ссылке); неудобство для оценки времени реального ПО.	**Машина исполнения компьютерных алгоритмов (МКА)** с центральным процессором, неограниченными лучами данных и программным текстом на луче Program.

Критика частично рекурсивных функций и построение Теории строк (2020)

В 84-страничной работе «Теория строк (слов). Недостаточная выразительность рекурсивных функций и арифметики для теории алгоритмов» (25 апреля 2020 года) Д.Л. Гуринович обосновывает тезис о том, что аппарат частично рекурсивных функций (ЧРФ), на котором со времён Гёделя, Клини и Чёрча строилась представимость алгоритмов в арифметике, не отражает современную сложность реальных вычислений.

Анализируя в подразделе VI базовые примитивно-рекурсивные функции (нулевую функцию, функцию следования $x + 1$, функции проекции I_n^m) и операторы суперпозиции, примитивной рекурсии и минимизации аргумента,

Гуринович указывает, что ни одна базовая функция и ни один оператор не могут работать с начальной частью своего входного аргумента, не читая его «хвоста» до конца (так как для увеличения или уменьшения числа на 1 необходимо прочитать входной аргумент целиком). Напротив, реальный алгоритм проверки сертификата в классе $\mathbb{NP}$ должен самостоятельно контролировать размер сертификата и отбраковывать чрезмерно длинный сертификат при чтении первого же символа за пределами допустимого порога, не тратя время на обработку всего «входа».

Для формального решения этой проблемы Д.Л. Гуринович строит аксиоматическую **теорию строк первого порядка с равенством**, опираясь на логическое исчисление Д. Гильберта и П. Бернайса, аксиомы импликации, конъюнкции, дизъюнкции, эквивалентности, отрицания, правила кванторов, аксиомы равенства J_1, J_2 и теорему дедукции. Ядро теории образуют четыре независимые группы аксиом, причём независимость каждой группы обосновывается построением интерпретаций в метатеоремах (м.5.1), (м.6.1), (м.7.1), (м.8.1):

1. **Разбиение строк (аксиомы 1–4):** любая строка a в произвольной позиции i делится на начальную и конечную части: $a = \mathrm{beg}(a, i) \cdot \mathrm{end}(a, i')$, причём после i-го символа в префиксе находится пустая строка $\ominus$ ($\mathrm{end}(\mathrm{beg}(a, i), i') = \ominus$, $i \neq \ominus$), а подстрока определяется как $\mathrm{str}(a, i, j) = \mathrm{beg}(\mathrm{end}(a, i), j)$.
2. **Конечный алфавит (аксиомы 5.1–5.2):** Всякий элемент $\mathrm{str}(a, i, 1)$ равен некоторому «атому» алфавита $\mathrm{Chr}(j)$, где непустые попарно различные символы нумеруются до фиксированного предела l_{ChrLim} , а при $i > l_{\mathrm{ChrLim}}$ функция $\mathrm{Chr}(i)$ возвращает $\ominus$.
3. **Концы и соединение строк (аксиомы 6.1, 7.1–7.2):**

Появление пустой строки $\ominus$ в позиции $i \neq 0$ служит критерием окончания строки ($\mathrm{str}(a,i,1) = \ominus \Rightarrow \mathrm{str}(a,i',1) = \ominus$), что позволяет определить длину конечной строки $\mathrm{len}(a) = i$ (а для бесконечной строки $\mathrm{len}(a) = \ominus$) и аксиоматизировать конкатенацию $a \cdot b$.

4. **Посимвольное равенство (аксиома 8.1):** $a = b \Leftrightarrow \forall i\, \mathrm{str}(a,i,1) = \mathrm{str}(b,i,1)$.

На этой основе Д.Л. Гуринович вводит определения посимвольного сравнения $\mathrm{Comp}(a,b)$ через первую позицию отличия $\mathrm{CompIn}(a,b)$, поиска подстроки $\mathrm{find}(\mathrm{in},\mathrm{what},i)$ и вставки $\mathrm{Ins}(a,i,b)$, выводя три ключевых, фундаментальных результата – 1) **Критерий локального сравнения (с.9.2)**, 2) **Критерий локальной вставки (т.9.4)** и 3) **Критерий локального извлечения данных (т.9.5)**:

$$\mathrm{len}(a) \neq \ominus \Rightarrow (\mathrm{IsStr}(\mathrm{len}(a),i,j) = 1 \Leftrightarrow \forall u\, \mathrm{str}(a \cdot u,i,j) = \mathrm{str}(a,i,j))$$

Эти теоремы формально выражают тот факт, что если запрашиваемая операция лежит в пределах длины начальной части $\mathrm{len}(a)$ ($i + j - 1 \leq \mathrm{len}(a)$), то результат функции на составной строке $a \cdot u$ не зависит от присоединённого справа «хвоста» u.

Далее в метатеоремах (м.9.1) и (м.9.2) Д.Л. Гуринович приводит доказательство того, что в теории строк с односимвольным алфавитом ($l_{\mathrm{ChrLim}} = 0$) и, как следствие, в арифметике невозможно в общем случае представить работу алгоритмов со строками многосимвольного алфавита так, чтобы размер логического вывода находился в полиномиальных (или любых иных) пределах от времени работы алгоритма. Если алгоритм проверки

русского правописания отвергает роман по первой же букве Ъ, то в односимвольной (унарной) записи 000...0 никакой вывод на основании фиксированной начальной части длины n невозможен: иначе все романы, представленные унарными строками длины $\geq n$, начинались бы на Ъ, а для бесконечного множества романов, не начинающихся на Ъ, осталось бы лишь конечное число вариантов строк длины меньше n.

Критика эквивалентности вычислимости по Тьюрингу и рекурсивных функций (2022–2023)

В работах «Вычислимость по Тьюрингу превосходит в принципиальном отношении вычислимость рекурсивных функций» (август 2022 года) и «Числа арифметики Пеано не обладают необходимыми для вычислимости свойствами...» (январь 2023 года) Д.Л. Гуринович подвергает ревизии классическое положение теории вычислимости о совпадении множества функций, реализуемых на машинах Тьюринга, и множества частично рекурсивных функций.

Аргументация Д.Л. Гуриновича опирается на понятие **«числа-строки»**, которое обладает одновременно свойствами натурального числа (подчиняясь аксиомам арифметики) и свойствами строки (имея вычислимую длину и посимвольный состав в конкретном представлении).

На ленте машины Тьюринга одно и то же натуральное число имеет разную длину при разных способах записи:

- число 5681245567 имеет в десятичной записи длину 10, а в шестнадцатеричной записи (152a0ed7f) - длину 9;
- число 10 имеет в десятичной записи длину 2, в двоичной (1010) - 4, а в стандартной унарной интерпретации Гильберта (00000000000) - длину 11;

- функция взятия первого символа $s(a,1)$ для числа 15 даёт в десятичной записи $s(15_9,1_9)=1_9$, а в шестнадцатеричной - $s(\mathrm{F}_{15},1_9)=\mathrm{F}_{15}$.

В рамках же абстрактной арифметики Пеано разные представления одного числа (например, десятичное 121_{10} и двоичное 1111001_2) равны между собой ($121_{10}=1111001_2$) и выступают одним и тем же объектом, поэтому никакая рекурсивная функция от этого аргумента не может выдать для него два разных значения длины или разные первые символы. Стандартное доказательство сводимости машины Тьюринга к рекурсивным функциям (например, в книге Дж. Булоса и Р. Джеффри «Вычислимость и логика», где левый и правый лучи ленты МТ рассматриваются как числа A и B) неявно использует внешнее по отношению к арифметике соглашение о смысле и способе записи чисел. Кроме того, Д.Л. Гуринович строит **невычислимые бесконечные интерпретации** арифметики Пеано, например, сопоставляя числу N бесконечную периодическую строку 1111..101111..10... с повторяющимся блоком из 2^N единиц и одного нуля, которая удовлетворяет аксиомам Пеано, но не может быть обработана за конечное время.

Полемизируя со статьёй Дж. Коркорана, У. Франка и М. Малони «String Theory» (The Journal of Symbolic Logic, 1974), где обосновывалась «синонимичность» теории строк Гермеса S_n и Тарского арифметике второго порядка, Д.Л. Гуринович утверждает, что при наличии в теории нескольких типов чисел-строк **обычная аксиома арифметической индукции** $A(0)\wedge\forall i(A(i)\Rightarrow A(i'))\Rightarrow\forall iA(i)$ **перестаёт быть истинной**. В качестве контрпримера он рассматривает формулу $A(i)\equiv l_9(i'')=(0_9)i'''$ («десятичная длина числа-строки $i+2$ равна числу $i+3$, приведённому к

десятичному типу»):

1. **База индукции** $A(0_0)$ для нуля стандартной унарной интерпретации 0_0 истинна, так как число 2 в унарной записи имеет вид 000 и его длина равна $l_9(0_0'') = 3_9 = (0_9)0_0'''$.

2. **Индуктивный переход** $\forall i(l_9(i'') = (0_9)i''' \Rightarrow l_9(i''') = (0_9)i'''')$ также оказывается истинным для всех типов чисел-строк: для унарного типа i_0 истинны и посылка, и заключение (длина унарной записи всегда на 1 больше величины числа), а для любой позиционной записи i_U ($U > 0$) посылка $l_9(i_U'') = (0_9)i_U'''$ всегда *ложна* (позиционная запись начиная с 1 короче унарной), что делает импликацию истинной.

3. **Заключение** $\forall i\, l_9(i'') = (0_9)i'''$ оказывается ложным, так как не выполняется для позиционных типов чисел-строк.

Взамен абстрактной арифметики Пеано Гуринович предлагает аксиоматику **«Арифметики строк»**:

- Предметные переменные и кванторы арифметики снабжаются **индексом типа** U (где $U = 0$ соответствует унарной записи, $U > 0$ - $(U + 1)$ -ичной позиционной системе счисления, а индексы вида SU - строкам без числовых свойств)[3]. Типированный квантор общности определяется через пустую строку на нулевом месте: $\forall_U i_U A(i_U) \Leftrightarrow \forall i(\ominus_U = s(i, 0) \Rightarrow A(i))$.

- Аксиомы теории Q (Q_1–Q_7) и схема индукции (Q_i) записываются с типированными кванторами отдельно для каждого типа U: $A(0_U) \wedge \forall_U i_U(A(i_U) \Rightarrow A(i_U')) \Rightarrow \forall_U i_U A(i_U)$.

- Формулируются **аксиомы нарушения равенства из-за**

строковых свойств (S_1, S_2, S_3 **)**: строки и числа-строки разных типов не равны между собой ($121_{10} \neq 1111001_2$); равенство хотя бы одной пары символов из двух алфавитов $e_U(i) = e_K(j)$ влечёт полное совпадение алфавитов и нулей (аксиома S_1), а для перевода числа из типа K в тип U с сохранением числовой величины вводится оператор приведения типов $(e_U(0))i_K$ по аналогии с языком Си с аксиомами $(S_2)\ 0_U = (e_U(0))0_K$ и $(S_3)\ i_U = (e_U(0))j_K \Rightarrow i'_U = (e_U(0))(j'_K)$.

Расширение математической логики средствами ООП (2025)

В статье от 22 июня 2025 года «Расширение математической логики средствами ООП для достижения необходимой выразительности при разборе вопросов вычислений и вычислимости» Д.Л. Гуринович указывает на ещё одну проблему выразительности классической математической логики.

Даже если в теории T заданы переменные $a = 121_D$ (с длиной $l(a) = 3_D$) и $b = 1111001_B$ (с длиной $l(b) = 111_B$), в рамках классической логики первого порядка (по Следствию 3.3 из книги Э. Мендельсона о введении новых функциональных букв при доказанном условии существования и единственности) благодаря взаимно-однозначной функции перекодировки между представлениями чисел можно определить «анти-функции» $s_1()$ и $l_1()$, которые «видят» вместо десятичного представления двоичное и наоборот: $l_1(a) = 111_B$ и $l_1(b) = 3_D$. Поскольку классическая логика не привязывает функции к объектам, теория T_1 с аксиоматизированными функциями $s_1(), l_1()$ оказывается логически эквивалентной исходной теории T с точностью до переименования функций.

Чтобы преодолеть эту неоднозначность «логически эквивалентных» наборов функций, Гуринович предлагает **расширить формальную логику**

синтаксисом и семантикой объектно-ориентированного программирования (ООП), восходящего к языку Smalltalk-80 Алана Кэя и Дэна Ингаллса. При записи длины строки как привязанного к объекту свойства с фиксированным именем l (a.l() = 3_D и b.l() = 111_B) альтернативное свойство с именем l_1 отсутствует, а попытка приписать тому же свойству .l() перевёрнутые значения (a.l() = 111_B и b.l() = 3_D) вступает в прямое противоречие с исходными равенствами.

5. Инженерно-математическое воплощение финитизма: ашина исполнения компьютерных алгоритмов, Русская машина Гуриновича (МКА)

Финитная установка Гильберта предполагала, что за формальной теорией стоит наглядная, конструктивно описываемая модель (интерпретация), обеспечивающая непротиворечивость аксиом и связь с реальностью. Считая машину Тьюринга неадекватной современным вычислительным системам и неудобной для оценки времени работы программ, Д.Л. Гуринович в 120-страничной работе от 16 марта 2021 года конструирует стандартную интерпретацию для своей теории компьютерных строк - **«Машину исполнения компьютерных алгоритмов» (МКА)**.

Ключевая архитектурная особенность МКА состоит в том, что **сам алгоритм, программный текст, является одной из входных символьных строк**, расположенной на выделенном луче данных Program, а номер позиции первого символа текущей команды хранится в десятичном представлении (разрядом единиц в первой ячейке) на луче i_Program. Благодаря этому программный текст, его использование в качестве алгоритма и время его исполнения становятся в МКА такими же объектами исследования теории компьютерных строк, как и обрабатываемые данные.

Масштабирование памяти без ограничения разрядности и языки SI / SMAU

В модели МКА Д.Л. Гуринович предлагает решение проблемы масштабирования реальных вычислительных систем, память которых на практике ограничена разрядностью адресации процессора (например, предел в 4 Гб для 32-разрядных процессоров или конечность адресного пространства в сетях).

Центральный процессор МКА имеет фиксированную разрядность и непосредственно обращается только к своим регистрам (Command, Source_1, Target_1, Continue_0 и др.) и начальным частям конечного числа лучей данных. Каждая ячейка луча данных устроена как автономный узел, способный обмениваться сообщениями лишь с двумя соседними ячейками (а первая ячейка - со второй ячейкой и процессором). Обращение к удалённой ячейке с произвольно большим номером осуществляется путём **передачи пакета сообщений по эстафете** от узла к узлу с поразрядным уменьшением передаваемого числа вплоть до установки указателя p1 = 1 в нужной ячейке.

Функционирование МКА описывается на трёх уровнях формальных языков:

1. **Стандартный язык SI (Язык стандартной интерпретации):** язык программирования среднего уровня для записи алгоритмов на луче Program. Содержит метки [next1];, переходы goto x1;, остановку .;, команды присваивания с дописыванием в конец x1() = ...;, полной заменой строки x1(0) = ...; или вставкой с позиции x1(x2) = ...;, а также базовые функции len(x1), Chr(x1), Value(x1), from(x1, x2), str(x1, x2, x3), Comp(x1, x2), Next(x1), Previous(x1).
2. **Аппаратный язык SI:** расширение языка SI, на котором написан «аппаратный интерпретатор» в ПЗУ процессора. Включает

свойства .Register и .SRegister, оператор косвенной адресации луча @ (@Source1), команды ожидания wait;, wait_i;, отправки сообщений SendTo_i и переключения режимов Setmode SMAU; / Setmode SI;.

3. **Язык SMAU (ОССУ - язык одновременных сообщений соседним узлам):** событийный язык без циклов и переходов для написания обработчиков сообщений в процессоре и ячейках. Каждая ячейка содержит регистры основного значения rv, теневого значения sv, указателей p1, p2, флагов увеличения/уменьшения ic, dc, флага MayBe1 и признака первой ячейки IsFirst.

Для бесконфликтной передачи пакетов и аксиоматизации времени в МКА применяются два механизма:

- **Сетки синхронизации и уход пакетов «в бесконечность»:** Передача сообщений синхронизирована по двоичным сеткам с интервалом 2^i моментов ($i = 0$ - 1 момент, $i = -1$ - $1/2$ момента, вплоть до $i = -4$ - $1/16$ момента, или «1 мгновение»)[3]. Эстафетные сообщения от процессора отправляются не чаще чем через 1 момент (скорость 1 ячейка за 1 момент), чтобы новые эстафеты не догоняли старые[3]. Когда пакет данных выполняет свою задачу (уменьшаемое число стало равно 1), тип сообщения переключается на «холостой» режим (/# Разряды для уменьшаемого числа исчерпаны с перебором (p1) #/), и пакет продолжает уходить по лучу в бесконечность, не производя изменений в остальных ячейках.

- **Теневой канал связи и оперативное завершение протокола:** чтобы избежать конфликтов, когда один луч данных одновременно выступает источником и получателем, каждая ячейка имеет два независимых канала - основной (rv) и «теневой» (sv). При этом

протокол передачи считается **оперативно завершённым** относительно процессора тогда, когда окончательный результат сформирован в 1-й ячейке, а в остальных ячейках формируется по мере удаления волны от процессора со скоростью не менее 1 ячейки за 1 момент.

Такая организация позволяет строго оценить время выполнения команд языка SI: при работе с неограниченными лучами данных время выполнения каждой команды зависит **линейно** от размера используемых в ней данных (плюс константа $T_{\mathrm{SiMaxCommand}}$ на работу в пределах регистров процессора). Вместе с тем учёт геометрии расположения данных на луче и конечной скорости света приводит Д.Л. Гуриновича к выводу о том, что для больших объёмов данных время сортировки N элементов на одномерном луче данных составляет не менее порядка N^2 единиц времени даже для быстрых алгоритмов сортировки (оценка $N \log N$ справедлива лишь для ограниченной памяти, где время доступа к любой ячейке считается константой).

6. Философское и социокультурное наследие: «рефлексивная логика», антииндивидуализм и ноология

Подобно тому как теоремы Гёделя о неполноте породили обширную философскую традицию, Д.Л. Гуринович выстраивает на основе своих математических работ собственную мировоззренческую и социологическую концепцию, которую он в 2017 году обозначает как **«рефлексивную логику»**, а в работе 2026 года развивает в учение о закономерностях функционирования разума - **«ноологию»**.

Формализация «Я», свободы воли и «коллективного алгоритма»

Д.Л. Гуринович неоднократно подчёркивает, что возможность формализации **рефлексии** - понимания системой того, «кто она такая» и какие данные задачи относятся непосредственно к ней, - представляется ему более

фундаментальным результатом, чем доказательство неравенств $\mathbb{NF} \neq \mathbb{F}$ или $\mathbb{NP} \neq \mathbb{P}$. Обращаясь к «дилемме заключённого» из теории игр и к примеру с подрывником, который должен снести направленным взрывом три дома, учитывая, не находится ли он сам внутри второго дома, автор указывает, что без математического выражения отличия «себя» от «не-себя» анализ жизнеспособных разумных систем невозможен.

В рамках «рефлексивной логики» Гуринович формулирует четыре философских обобщения:

1. **Определение субъектности через границу ограничения:** В языке LA конкретному алгоритму-решению Sol ставится в соответствие урезанный набор сертификатов, из которого отброшены недоступные ему подтверждающие доказательства. Отсюда автор выводит тезис: *«Быть кем-то - это не иметь чего-то такого, что есть в мире, но при этом отделено от тебя какой-то границей»*.

2. **Соотношение дескриптивных законов природы и нормативных требований «долга»:** если базовая проверка (при аргументе долга «объективно») подобна дескриптивным законам природы, которым подчиняются все объекты, то персональная подсказка в «аргументе долга» носит *нормативный* характер - ей подчиняется лишь выборка тех алгоритмов, которые способны распознать требование к себе и исполнить его.

3. **Критика «демона Лапласа» и алгоритмическая природа свободы воли:** ощущение свободы воли и непредсказуемости собственных мыслей возникает даже в детерминированном мире потому, что субъект «не может во всём опередить себя самого» на задачах самоприменения. Равным образом опровергается всеведение «демона

Лапласа»: чтобы знать систему X, наблюдатель должен отличаться от X, а при попытке демона Лапласа изучить собственный мозг материальные процессы запоминания будут изменять состояние мозга сильнее, чем оно было познано (по неформальной оценке автора, система не может знать больше половины информации о себе самой).

4. **Критика индивидуализма и концепция «коллективного алгоритма»:** Вторую теорему Гёделя о неполноте Д.Л. Гуринович интерпретирует как математическое опровержение самодостаточного индивидуализма: теория, доказавшая собственную правоту («непротиворечивость»), является противоречивой. В задаче «Сфинкс» отдельный алгоритм $Sol(\overline{AntiSol_S}, S, \dots)$ не способен сам подтвердить истинность $AntiSol_S$, но его оперативный ответ 0 («я не справлюсь с этим в такие сроки») служит необходимым промежуточным звеном («полуфабрикатом»), позволяющим другим алгоритмам в составе единого **«коллективного алгоритма»** быстро построить полное доказательство истинности $AntiSol_S$.

Историософия науки: судьба школы Гильберта и «общественный наследственный код»

В работах 2017–2026 годов Д.Л. Гуринович анализирует исторические и социальные причины того, почему за взлётом математической логики в эпоху Гильберта и Гёделя последовал длительный спад интереса к основаниям математики и разрыв между чистой математикой и сферой ИТ. В его текстах выделяются три ключевых фактора:

- **Разрушение немецкой математической школы и гибель Тьюринга:** в статье января 2023 года автор подробно останавливается на роли немецкого математического сообщества второй половины XIX -

первой трети XX века (от Гаусса и Георга Кантора до Феликса Клейна, Давида Гильберта, Эмми Нётер и главы университетского отдела министерства образования Пруссии Фридриха Альтхоффа). По оценке Д.Л. Гуриновича, насильственное уничтожение этого сообщества накануне и в ходе Второй мировой войны, а также последующее уголовное преследование и гибель Алана Тьюринга в Англии прервали «период логических открытий»: после смерти Гильберта и распада гёттингенской среды даже Гёдель перестал отвечать на самые принципиальные вопросы оснований математики, поскольку их больше никто не ставил на уровне Гильберта.

- **Профессиональная деформация «формализма»:** сопоставляя историю создания общей теории относительности, где физик А. Эйнштейн выбрал адекватный реальности формализм, а математик Г. Минковский в переписке с Гильбертом шутил о необходимости «очиститься» после общения с физиками и вернуться к математическому способу мышления, Д.Л. Гуринович отмечает склонность математиков работать исключительно внутри принятых в сообществе абстрактных конструкций, отбрасывая «лишние» свойства реального мира. В результате, когда в компьютерную эпоху потребовалось перенести из реальности в математику свойства вычислимых строк и времени работы алгоритмов, академическое сообщество продолжило опираться на невычислимые абстракции арифметики Пеано, предложив взамен реформировать подготовку кадров путём обучения практикующих программистов формальной логике.

- **Утрата «общественного наследственного кода» и идеологический барьер (работа 2026 года):** в работе марта 2017 года Д.Л. Гуринович высказывает мысль о том, что откат от достигнутого

уровня логики произошёл потому, что теоремы Гильберта, Гёделя, Тьюринга и Тарского о неполноте и неразрешимости вошли в мировоззренческий конфликт с доминирующим в XIX–XX веках индивидуалистическим трендом, отвергающим внешние ограничения над отдельным разумом. В работе «Причины научно-религиозного и демографического тупиков современного общества и вопрос об их преодолении» от 26 июля 2026 года) этот тезис обобщается Д.Л. Гуриновичем в рамках концепции **«общественного наследственного кода»** - небиологической системы передачи знаний и этики общественного долга, которую исторически поддерживали аристократическая и религиозная страты. Устранение этих страт и возобладание биологического индивидуализма, «извращенизации», привели, по мнению Д.Л. Гуриновича, сначала к прекращению века фундаментальных научных открытий (с переходом к механической эксплуатации прежних теорий), а затем и к демографическому кризису, преодоление которого требует возврата к приоритету теоретического разума и коллективного служения.