УДК 303.732.4
DOI: 10.26102/2310-6018/2026.56.5.001

Облачная платформа масштабируемого моделирования магнитостатических полей методом конечных элементов для формирования наборов данных реконструкции магнитных характеристик

В.А. Сурняев✉, В.В. Гречихин

Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова, Новочеркасск, Российская Федерация

Резюме. В статье рассматривается задача ускоренного формирования репрезентативных наборов данных численного моделирования, используемых при реконструкции магнитных характеристик материалов в информационно-измерительных системах для коротких образцов. В ранее разработанных измерительных решениях повышение чувствительности достигается, в том числе, введением параллельного магнитного шунта, однако интерпретация измерительной информации требует решения обратной задачи и построения моделей, привязанных к конкретной геометрии установки. При изменении геометрии или параметров магнитной системы возникает необходимость повторного расчета большого числа сценариев методом конечных элементов. Прямое распараллеливание решения одной конечно-элементной задачи затруднено вследствие глобальной связности разреженной системы линейных алгебраических уравнений и высокой стоимости межпроцессорных обменов, поэтому предлагается параллелизация на уровне независимых расчетных задач параметрического перебора. Предложена облачная микросервисная архитектура, реализующая полный автоматизированный цикл: генерация сетки, формирование постановки магнитостатической задачи, численное решение, централизованное хранение результатов и формирование обучающей выборки. Реализация выполнена в инфраструктуре Yandex Cloud. Экспериментально показано, что среднее время расчета одной точки составляет 22,78 секунды, из которых 3,64 секунды занимает генерация сетки, а 19,13 секунды – решение задачи. Время формирования выборки из 900 характеристик сокращается со 105 часов до 9 минут при увеличении числа параллельных контейнеров до 900, что подтверждает близкую к линейной масштабируемость предложенного подхода.

Ключевые слова: облачные вычисления, распределенные вычисления, метод конечных элементов, магнитостатика, реконструкция магнитных характеристик, микросервисная архитектура, автоматизация расчетов, масштабируемость.

Для цитирования: Сурняев В.А., Гречихин В.В. Облачная платформа масштабируемого моделирования магнитостатических полей методом конечных элементов для формирования наборов данных реконструкции магнитных характеристик. *Моделирование, оптимизация и информационные технологии.* 2026;14(5). URL: https://moitvivt.ru/ru/journal/article?id=2262 DOI: 10.26102/2310-6018/2026.56.5.001

Cloud platform for scalable finite element modeling of magnetostatic fields to generate datasets for magnetic characteristics reconstruction

V.A. Surnyaev✉, V.V. Grechikhin

Platov South-Russian State Polytechnic University (NPI), Novocherkassk, the Russian Federation

Abstract. The problem of accelerated generation of representative numerical simulation datasets, used in the reconstruction of magnetic characteristics of materials in information-measuring systems for short

samples, is considered. In previously developed measuring solutions, increased sensitivity is achieved, among other things, by introducing a parallel magnetic shunt. However, the interpretation of measurement data requires solving an inverse problem and constructing models tailored to the specific geometry of the setup. When the geometry or parameters of the magnetic system change, a large number of scenarios must be recalculated using the finite element method. Direct parallelization of solving a single finite element problem is challenging due to the global connectivity of the sparse system of linear algebraic equations and the high cost of interprocessor communications. Therefore, parallelization at the level of independent computational tasks during a parametric sweep is proposed. A cloud-based microservice architecture is proposed that implements a fully automated cycle: mesh generation, formulation of the magnetostatic problem, numerical solution, centralized storage of results, and generation of a training dataset. The implementation was carried out within the Yandex Cloud infrastructure. It is experimentally demonstrated that the average calculation time for a single point is 22.78 seconds, of which mesh generation takes 3.64 seconds and the problem solving takes 19.13 seconds. The time required to generate a dataset of 900 characteristics is reduced from 105 hours to 9 minutes when the number of parallel containers is increased to 900, confirming the near-linear scalability of the proposed approach.

Keywords: cloud computing, distributed computing, finite element method, magnetostatics, magnetic characteristic reconstruction, microservice architecture, automation of calculations, scalability.

For citation: Surnyaev V.A., Grechikhin V.V. Cloud platform for scalable finite element modeling of magnetostatic fields to generate datasets for magnetic characteristics reconstruction. *Modeling, Optimization and Information Technology.* 2026;14(5). (In Russ.). URL: https://moitvivt.ru/ru/journal/article?id=2262 DOI: 10.26102/2310-6018/2026.56.5.001

Введение

Развитие методов неразрушающего контроля ферромагнитных изделий требует повышения точности определения магнитных характеристик материалов (кривых намагничивания B(H), петель гистерезиса) на образцах произвольной формы. В задачах контроля «коротких» образцов (пластины, стержни), где невозможно создать замкнутую магнитную цепь, возникает проблема размагничивающего фактора [1]. Для решения этой проблемы и повышения чувствительности применяются измерительные системы с параллельным магнитным шунтом [2, 3]. Однако введение шунта усложняет топологию поля: зависимость выходного сигнала измерительной обмотки от свойств материала становится существенно нелинейной и определяется геометрическими параметрами системы.

Традиционные методы пересчета магнитных величин (метод размагничивающего фактора) в таких условиях дают высокую погрешность [4, 5]. Интерпретация измерительной информации при этом сводится к обратной задаче магнитометрии; для задач магнитостатики существенны вопросы корректности постановки и однозначности решения [6, 7]. Перспективным решением является применение методов машинного обучения (Machine Learning, ML) и искусственных нейронных сетей для решения обратной задачи магнитометрии [8, 9].

Ключевым ограничением ML-подхода является потребность в обучающих выборках большого объема (тысячи примеров). Получение таких данных экспериментально невозможно из-за временных и материальных затрат. Одним из способов получения данных является генерация синтетических данных с помощью численного моделирования.

Однако стандартные пакеты конечно-элементного анализа (FEM), такие как ANSYS или COMSOL, ориентированы на высокопроизводительные вычисления (High-Performance Computing (HPC)) в рамках одной сложной задачи. При массовом расчете тысяч однотипных задач с разными параметрами исследователи сталкиваются с проблемой: последовательный расчет на локальных вычислительных станциях занимает

недели и требует вынесения параллелизма на уровень независимых расчетных задач и автоматизации вычислительного конвейера [10].

Целью данной работы является разработка методики и программной архитектуры для высокопроизводительной генерации синтетических данных, позволяющей сократить время моделирования на порядки за счет использования облачных технологий.

Научная новизна работы заключается в следующем:

1) предложен подход формирования обучающих выборок для реконструкции магнитных характеристик, основанный на параллельном выполнении независимых конечно-элементных расчетов и централизованном хранении результатов;

2) разработана микросервисная архитектура вычислительного конвейера с контейнеризацией, обеспечивающая воспроизводимость вычислительных экспериментов;

3) реализована переносимая конфигурация на инфраструктуре Yandex Cloud;

4) разработаны и зарегистрированы программные средства автоматизации подготовки данных для численного моделирования[1,2].

Материалы и методы

Математическая модель. Задача генерации элемента обучающей выборки сводится к расчету магнитостатического поля в 2D-постановке (плоскопараллельное поле). Магнитостатическое поле, созданное постоянными токами, описывается системой уравнений Максвелла:

$$\begin{cases} rot\vec{H} = \vec{0}, \\ \mathrm{div}\vec{B} = 0, \\ \vec{B} = \mu\vec{H}, \end{cases} \quad (1)$$

где H – напряженность магнитного поля, B – магнитная индукция, μ – абсолютная магнитная проницаемость среды.

Систему (1) для плоскопараллельных полей можно свести к одному дифференциальному уравнению путем введения векторного магнитного потенциала A (учитывая, что $B = rotA$):

$$\mathrm{div}(\mu^{-1} gradA) + J = 0, \quad (2)$$

где J – плотность стороннего тока намагничивающих обмоток.

На границах раздела сред с различными магнитными проницаемостями (μ^+ и μ^-) выполняются условия непрерывности для касательных и нормальных составляющих полей, что для векторного потенциала принимает вид:

$$A^+ = A^-; \quad \frac{1}{\mu^-}\frac{\partial A^-}{\partial n} = \frac{1}{\mu^+}\frac{\partial A^+}{\partial n}. \quad (3)$$

Сложность решаемой задачи обусловлена нелинейностью характеристики ферромагнитного материала: магнитная проницаемость зависит от индукции поля, $\mu = \mu(B)$. Для численного решения данной нелинейной краевой задачи используется итерационный метод Ньютона-Рафсона, требующий пересчета глобальной матрицы жесткости на каждой итерации. Это делает процесс вычисления ресурсоемким и плохо поддающимся внутреннему распараллеливанию.

[1] Сурняев В.А.; правообладатель Сурняев В.А. GmshModelBuilder – программа автоматического построения 2-D конечно-элементных сеток и описания материалов в Gmsh: заявл. 28.05.2025, № 2025663614; опубл. 17.06.2025. Свидетельство о государственной регистрации программы для ЭВМ № 2025665383 Российская Федерация.

[2] Сурняев В.А.; правообладатель Сурняев В.А. GetDPRunner – программа автоматической генерации .pro-файлов и запуска статического магнитного расчёта в GetDP: заявл. 28.05.2025, № 2025663626; опубл. 05.06.2025. Свидетельство о государственной регистрации программы для ЭВМ № 2025664673 Российская Федерация.

Для численного моделирования использовались открытые программные комплексы Gmsh и GetDP. Gmsh применяется для построения геометрии и генерации расчетной сетки конечных элементов [11]. GetDP используется для решения магнитостатической задачи методом конечных элементов на построенной сетке [12]. Связка «генерация сетки – решение – постобработка» пригодна для автоматизации и контейнерного исполнения в среде Linux.

Каждый расчетный сценарий формируется как набор входных артефактов: файл сетки (формат msh), файл постановки задачи (формат pro), включающий описание материалов, источников поля и граничных условий, спецификация контрольных точек или областей, в которых извлекается искомая величина. Результаты расчетов сохраняются вместе с параметрами сценария и идентификаторами версий программного окружения, что обеспечивает воспроизводимость вычислительного эксперимента [13, 14].

Формализация задачи распределенного вычислительного процесса. Пусть параметры магнитной системы (геометрия, свойства материалов, намагничивающий ток и др.) описываются вектором p. Для каждого набора параметров p_k требуется выполнить численный расчет магнитостатического поля и получить целевую величину y_k (например, значения магнитной индукции в контрольных точках или интегральные характеристики магнитной цепи). Требуемый набор данных имеет вид $D = \{(p_k, y_k)_{k=1..N}$.

Параметрический расчет рассматривается как отображение F:

$$P \rightarrow \mathbb{R}^d, \tag{4}$$

где P – пространство параметров, d – размерность вектора результатов (например, значения B в контрольных точках). Для сценария k вычисляется:

$$y_k = F(p_k),\ D = \{(p_k, y_k)\}_{k=1}^{N}. \tag{5}$$

Построение обучающей выборки D рассматривается как множество независимых задач $T = \{t_k\}_{k=1..N}$, где каждая задача t_k включает следующие этапы:

1) генерация конечно-элементной сетки для заданной геометрии (t_k^{mesh});
2) численное решение методом конечных элементов (t_k^{solve});
3) экспорт результатов и сохранение метаданных (t_k^{io}).

Чистое время выполнения одной вычислительной задачи представляется суммой соответствующих этапов конвейера:

$$t_k = t_k^{mesh} + t_k^{solve} + t_k^{i0}. \tag{6}$$

Для N независимых задач целесообразно использовать распределенное выполнение на M вычислительных контейнерах. С учетом балансировки нагрузки, общее время формирования всего набора данных оценивается выражением:

$$T(M) \approx \frac{1}{M}\sum_{k=1}^{N} t_k + T_{ovh}(M), \tag{7}$$

где в $T_{ovh}(M)$ объединены все невычислительные затраты времени: сетевые задержки при программном обращении к API облачного провайдера для инициации задач, инициализация вычислительной среды («холодный старт» контейнера T_{init}), операции независимого сохранения результатов (например, в объектное хранилище) и время финальной агрегации данных.

Представленная модель наглядно отражает специфику облачных вычислений при изменении степени параллелизма M. Если весь набор характеристик рассчитывается последовательно в одном контейнере ($M = 1$), время запуска T_{init} прибавляется к общему времени лишь единожды, и его доля пренебрежимо мала. Однако при стратегии максимального распараллеливания, когда под каждую расчетную точку стартует отдельный эфемерный контейнер ($M = N$), время выполнения всего процесса будет

определяться самым длительным узлом: $T(N) \sim max(t_k) + T_{ovh}(N)$. В таком сценарии накладные расходы на старт среды T_{init} могут быть сопоставимы со временем полезного расчета t_k и оказывают определяющее влияние на общую длительность сбора данных.

Масштабирование системы оценивалось с помощью классических метрик ускорения $S(M)$ и эффективности $E(M)$:

$$S(M) = T(1)/T(M), \tag{8}$$

$$E(M) = S(M)/M. \tag{9}$$

Архитектура облачной платформы и программная реализация. Платформа реализует микросервисный вычислительный конвейер и включает пользовательский интерфейс и программный интерфейс вызова сервисов, сервис постановки и оркестрации задач, пул контейнеров для численного моделирования, объектное хранилище файлов и управляемую базу данных PostgreSQL для хранения метаданных (Таблица 1). Вычислительные задачи запускаются в виде контейнеров, а логика запуска и контроля выполнения реализуется серверными функциями. Выбор облачной модели опирается на ключевое свойство эластичности облака и модель предоставления ресурсов «по требованию».

Таблица 1 – Сервисы облачного провайдера, используемые в платформе
Table 1 – Mapping of cloud provider services used in the platform

Функциональный компонент системы	Yandex Cloud
Интерфейс пользователя	Compute Cloud
API для вызова сервисов	API Gateway
Серверные функции	Cloud Functions
Управление контейнерами	Serverless Containers
Реестр контейнеров	Container Registry
Объектное хранилище	Object Storage
База данных	Managed PostgreSQL

Общая схема взаимодействия компонентов платформы представлена на архитектурной диаграмме Yandex Cloud (Рисунок 1), а также на диаграмме последовательности (Рисунок 2), описывающей полный цикл выполнения задачи: постановка, оркестрация, вычисление и сохранение результата.

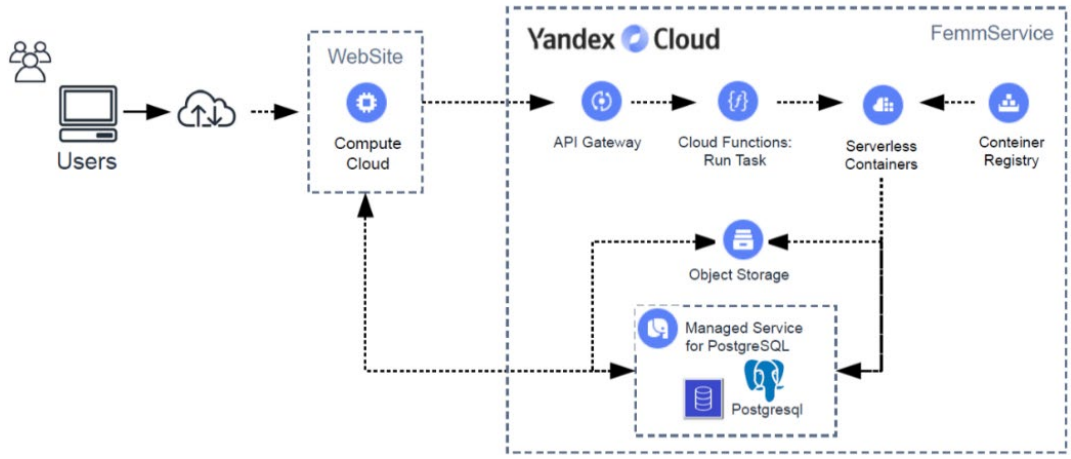

Рисунок 1 – Архитектура облачной информационно-измерительной системы на основе инфраструктуры Yandex Cloud
Figure 1 – Architecture of the cloud information and measurement system based on Yandex Cloud infrastructure

Функционально система разделена на следующие компоненты:

1) Узел взаимодействия и управления (Web-Orchestrator). Реализован на базе фреймворка Django. Принимает запросы пользователя, загружает базовые файлы геометрии (.gmsh, .pro) и конфигурацию параметров (объем выборки, диапазоны токов, степень параллелизма). Инициализирует задачу, сохраняя исходные файлы в объектное хранилище (Yandex Object Storage), а метаданные – в базу данных PostgreSQL.

2) Узел бессерверной маршрутизации (Serverless Routing). Связующее звено на базе API Gateway и облачных функций (Yandex Cloud Functions). Функция формирует конфигурацию кластера и программно инициирует запуск требуемого количества контейнеров.

3) Вычислительный кластер генерации данных (Workers Pool). Группа Docker-контейнеров (Serverless Containers). Каждый контейнер:

– Загружает исходные файлы из объектного хранилища;

– Использует авторские программные модули GmshModelBuilder[1] и GetDPRunner[2] для автоматической генерации уникальной сетки и проведения нелинейного расчета;

– Сохраняет результаты виртуальных измерений (векторы магнитной индукции) и метрики времени выполнения в PostgreSQL.

4) Система хранения (Storage Layer). Гибридная система, включающая объектное хранилище (для файлов геометрии) и транзакционную БД PostgreSQL (для агрегации структурированных результатов FEM-расчетов сотен контейнеров в единый обучающий датасет).

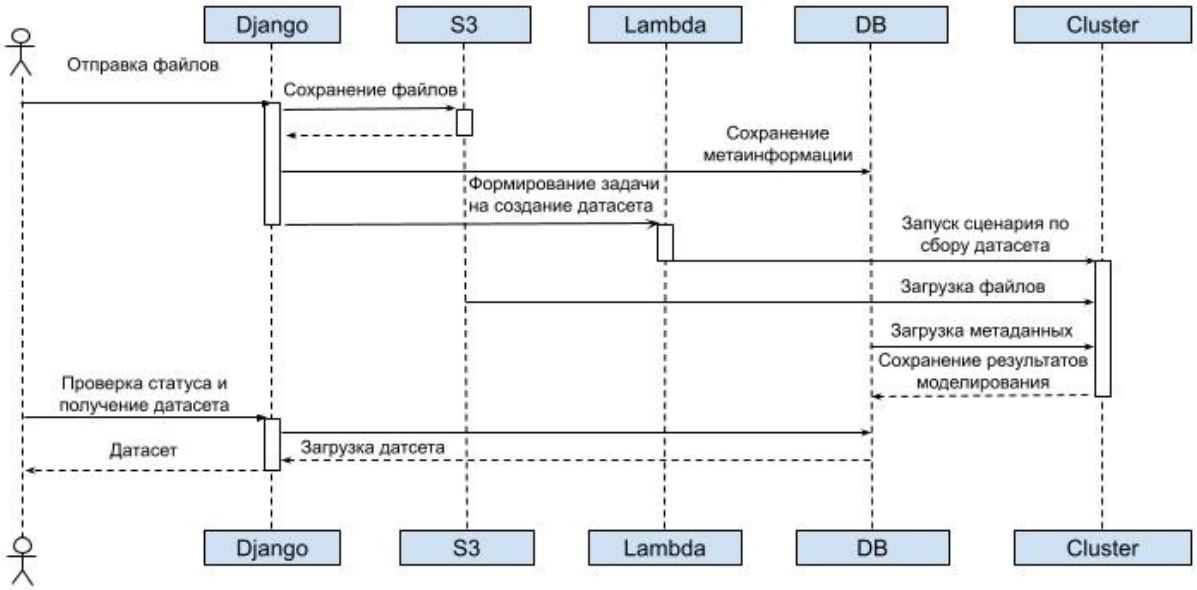

Рисунок 2 – Диаграмма последовательности взаимодействия компонентов облачной платформы
Figure 2 – Sequence diagram of interactions between cloud platform components

Контейнеризация вычислительных компонентов выполняется с использованием технологии Docker, что обеспечивает переносимость приложения и единообразие зависимостей при запуске в различных вычислительных средах[3].

[3] Merkel D. *Docker: Lightweight Linux Containers for Consistent Development and Deployment.* Linux Journal. URL: https://www.linuxjournal.com/content/docker-lightweight-linux-containers-consistent-development-and-deployment (дата обращения: 10.03.2026).

Результаты

Экспериментальная оценка выполнена для процесса формирования обучающего набора данных, включающего 900 вариантов магнитных характеристик материала. В качестве базовой геометрической модели использовалась конструкция измерительной системы с параллельным магнитным шунтом, предложенная и верифицированная авторами в предыдущих работах [2, 3]. Каждая магнитная характеристика рассчитывалась для 15 различных точек кривой намагничивания (варьирование тока возбуждения), что в совокупности потребовало решения $N = 13500$ независимых задач.

В ходе эксперимента оценивались статистические временные характеристики отдельной задачи и эффективность масштабирования системы при увеличении числа параллельных контейнеров M.

Полученные статистические показатели (Таблица 2) демонстрируют, что среднее время чистой калькуляции одной задачи составляет $t \approx 22{,}78$ с. При этом генерация сетки в Gmsh занимает стабильное время (в среднем 3,64 с с минимальным отклонением), а основная доля времени и весь наблюдаемый разброс (от 5,76 до 35,32 с) приходятся на решение задачи в GetDP.

Данная вариативность имеет строгую физико-математическую природу: при переборе параметров изменяется ток намагничивающей обмотки. При высоких значениях тока ферромагнитные элементы системы (шунт и образец) переходят в область глубокого насыщения. Работа на высоконелинейных участках кривой $\mu(B)$ существенно усложняет сходимость итерационного алгоритма Ньютона-Рафсона, требуя значительно большего числа пересчетов матрицы жесткости по сравнению с расчетами в слабых полях.

Таблица 2 – Статистические показатели времени выполнения расчета одной точки
Table 2 – Statistical metrics of the runtime for a single calculation point

Показатель	Полное время, с	Генерация сетки в Gmsh, с	Расчет в GetDP, с
Среднее	22,78	3,64	19,13
Стандартное отклонение	7,23	0,43	7,20
Минимум	9,22	2,86	5,76
Медиана	24,65	3,59	20,98
Максимум	39,25	7,40	35,32

Масштабирование системы исследовалось путем изменения количества одновременно работающих контейнеров от 1 до 900 (Таблица 3).

Таблица 3 – Масштабирование времени формирования обучающей выборки при увеличении числа контейнеров
Table 3 – Scaling of dataset generation time with increasing number of containers

Количество параллельных контейнеров (M)	Время формирования набора данных (T)	Ускорение S(M)	Эффективность E(M)
1	105 часов	1,00	1,00
10	11 часов	9,55	0,95
100	1,5 часа	70	0,70
900	9 минут	700	0,78

Анализ показал, что чистое машинное время для расчета 13500 точек составляет ~ 85,4 часа. При последовательном выполнении ($M = 1$) общее время составило 105 часов, откуда оценка накладных расходов T_{ovh} ~ 19,6 ч. В данном режиме издержки имеют ярко выраженный кумулятивный характер: в пересчете на одну задачу сетевой HTTP/API-запрос, установки соединения и записи результата в БД занимают около 5,2 секунд, которые последовательно суммируются 13500 раз.

При максимальном распараллеливании ($M = 900$) каждый контейнер обрабатывает собственный пакет из 15 задач, а общее время генерации сокращается до 9 минут. Оценка накладных расходов составляет $T_{ovh}(900)$ ~ 3,3 мин. На данном уровне параллелизма природа накладных расходов меняется: кумулятивные сетевые задержки нивелируются асинхронным запуском. Значение 3,3 мин отражает время параллельной инициализации 900 контейнеров («холодный старт»), задержки, вызванные квотированием API провайдера при массовом старте и время ожидания завершения самых медленных контейнеров.

В проведенном эксперименте зависимость эффективности вычислений $E(M)$ от числа контейнеров носит немонотонный характер. Как видно из Таблицы 3, при $M = 100$ эффективность локально снижается до 0,70. В этом режиме каждый контейнер последовательно обрабатывает по 135 задач, накапливая кумулятивные сетевые задержки при установке соединения и записи результатов в БД (Tovh(100) ~ 39 мин). Однако при переходе к $M = 900$ размер пакета снижается до 15 задач на узел. Кумулятивные задержки оркестратора эффективно нивелируются высокой степенью параллелизма, а основной вклад в накладные расходы начинает вносить разовая первичная инициализация вычислительной среды («холодный старт» пула контейнеров).

Результаты подтверждают, что при переходе от одного контейнера к параллельному выполнению задач достигается существенное сокращение общего времени формирования обучающей выборки. Наблюдаемая эффективность остается высокой даже при большом числе контейнеров, что указывает на доминирование вычислительной составляющей над накладными расходами оркестрации для рассматриваемого типа нагрузки. Тем самым обеспечивается практическая возможность оперативного формирования наборов данных при изменении геометрии измерительной системы или параметров исследуемого материала.

Обсуждение

В бессерверной облачной среде выбор числа параллельных контейнеров M определяется компромиссом между временем получения результата и стоимостью вычислений. В простейшей модели стоимость генерации выборки масштабируется пропорционально числу исполнителей и времени их работы:

$$C(M) \approx c_{cpu} \cdot M \cdot T(M) + C_{stor} + C_{net}, \quad (10)$$

где c_{cpu} – тарификация вычислительного узла в единицу времени, C_{stor} – стоимость хранения артефактов в объектном хранилище, C_{net} – стоимость сетевого трафика.

Учитывая модель времени

$$T(M) = N\bar{t}/M + T_{\text{ovh}}(M), \quad (11)$$

подстановка показывает, что стоимость чистых вычислений

$$c_{cpu} \cdot M \cdot (N\bar{t}/M) = c_{cpu}N\bar{t} \quad (12)$$

остается константой (оплачивается только полезная работа процессора). Дополнительные затраты при росте M возникают исключительно из-за накладных

расходов $T_{ovh}(M)$. Типовая постановка выбора M в таких условиях формулируется как задача минимизации стоимости при ограничении на требуемое время завершения T_{req}, что соответствует классической задаче бюджетно-ограниченного планирования пула задач (Bag-of-Tasks) [15]:

$$\min_{M} C(M), \quad T(M) \leq T_{req}. \tag{13}$$

Полученные нами результаты согласуются с общими тенденциями применения облачных технологий для вычислительно-интенсивных инженерных расчетов, где ключевыми факторами эффективности выступают стратегия планирования и независимость нагрузок [16]. Предложенный подход позволяет достигать линейного масштабирования и избегать усложнения вычислительного ядра, что выгодно отличает его от попыток применения сложных параллельных решателей внутри одной конечно-элементной задачи [17].

Архитектурно предложенная платформа реализует современную «облачно-нативную» (Cloud-Native) модель. Как отмечается в обзорной работе [18], оркестрация через бессерверные функции и обмен данными через объектное хранилище обеспечивают идеальный баланс между автоматическим масштабированием и моделью оплаты по факту использования (Pay-as-you-go).

Практическая значимость платформы заключается в радикальном сокращении времени подготовки данных при адаптации измерительной системы к новой геометрии или материалам.

Заключение

Разработана и экспериментально исследована облачная платформа для масштабируемого формирования наборов данных численного моделирования магнитостатических полей методом конечных элементов. Доказано, что переход к бессерверному распределенному выполнению независимых задач (Bag-of-Tasks) позволяет сократить время подготовки обучающих выборок для ML-моделей на несколько порядков, избегая неэффективных внутренних распараллеливаний итерационных решателей.

Ключевые научные и практические результаты исследования:

1. Выполнена формализация процесса синтеза данных как распределенного вычисления. Предложена математическая оценка накладных инфраструктурных расходов (T_{ovh}), учитывающая API-квоты облачного провайдера и эффекты «холодного старта».

2. Разработан и защищен свидетельствами Роспатента вычислительный конвейер на базе модулей GmshModelBuilder и GetDPRunner, автоматизирующий генерацию сетки и управление решателем без интерактивного участия оператора.

3. Реализована кроссплатформенная микросервисная архитектура, доказавшая свою работоспособность и переносимость в гетерогенных инфраструктурах (Yandex Cloud).

4. Экспериментально подтверждена высокая эффективность масштабирования ($E(M) = 0{,}78$): при увеличении числа параллельных контейнеров до 900, время формирования нелинейной выборки из 900 характеристик снижено со 105 часов до 9 минут.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Сандомирский С.Г. *Расчет и анализ размагничивающего фактора ферромагнитных тел.* Минск: Беларуская навука; 2015. 244 с.

2. Сурняев В.А., Блажко И.О., Блажкова Е.Н. и др. Применение магнитного шунта для повышения чувствительности устройства испытания образцов магнитострикционных материалов. *Инженерный вестник Дона.* 2021;(9). URL: https://ivdon.ru/ru/magazine/archive/n9y2021/7200
Surnyaev V.A., Blazhko I.O., Blazhkova E.N., et al. Application of a magnetic shunt to increase the sensitivity of a device for testing samples of magnetostrictive materials. *Engineering Journal of Don.* 2021;(9). (In Russ.). URL: https://ivdon.ru/en/magazine/archive/n9y2021/7200
3. Сурняев В.А., Сурняев Д.А. Устройство измерения магнитных характеристик магнитострикционных материалов. *Инженерный вестник Дона.* 2016;(4). URL: https://ivdon.ru/ru/magazine/archive/n4y2016/3956
Surnyaev V.A., Surnyaev D.A. Measuring device of the magnetic characteristics of the magnetostrictive materials. *Engineering Journal of Don.* 2016;(4). (In Russ.). URL: https://ivdon.ru/en/magazine/archive/n4y2016/3956
4. Бахвалов Ю.А., Горбатенко Н.И., Гречихин В.В. Метод решения обратных задач магнитных измерений. *Измерительная техника.* 2015;(3):58–60.
Bakhvalov Y.A., Gorbatenko N.I., Grechikhin V.V. A Method of Solving Inverse Problems of Magnetic Measurements. *Measurement Techniques.* 2015;58(3):336–340. https://doi.org/10.1007/s11018-015-0711-5
5. Aharoni A. Demagnetizing factors for rectangular ferromagnetic prisms. *Journal of Applied Physics.* 1998;83(6):3432–3434. https://doi.org/10.1063/1.367113
6. Beilina L., Smirnov Yu.G. *Nonlinear and Inverse Problems in Electromagnetics: PIERS 2017, 22–25 May 2017, Saint Petersburg, Russia.* Cham: Springer; 2018. 145 p. https://doi.org/10.1007/978-3-319-94060-1
7. Дякин В.В., Кудряшова О.В., Раевский В.Я. К вопросу о корректности прямой и обратной задачи магнитостатики. Часть 2. *Дефектоскопия.* 2018;(10):15–24.
Dyakin V.V., Kudryashova O.V., Rayevskii V.Y. On the Well-Posedness of the Direct and Inverse Problem of Magnetostatics. Part 2. *Russian Journal of Nondestructive Testing.* 2018;54(10):687–697. https://doi.org/10.1134/S1061830918100030
8. Оборнев Е.А., Оборнев И.Е., Родионов Е.А., Шимелевич М.И. Применение нейронных сетей в нелинейных обратных задачах геофизики. *Журнал вычислительной математики и математической физики.* 2020;60(6):1053–1065. https://doi.org/10.31857/S0044466920060071
Obornev E.A., Obornev I.E., Rodionov E.A., Shimelevich M.I. Application of Neural Networks in Nonlinear Inverse Problems of Geophysics. *Computational Mathematics and Mathematical Physics.* 2020;60(6):1025–1036. https://doi.org/10.1134/S0965542520060071X
9. Elgy J., Ledger P.D. Reduced order model approaches for predicting the magnetic polarizability tensor for multiple parameters of interest. *Engineering with Computers.* 2023;39(6):4061–4076. https://doi.org/10.1007/s00366-023-01868-x
10. Воскобойников М.Л., Феоктистов А.Г. Сравнительный анализ систем управления научными рабочими процессами. *Информационные и математические технологии в науке и управлении.* 2024;(3):102–111. https://doi.org/10.25729/ESI.2024.35.3.009
Voskoboinikov M.L., Feoktistov A.G. Comparative analysis of scientific workflow management systems. *Informational and mathematical technologies in science and management.* 2024;(3):102–111. (In Russ.). https://doi.org/10.25729/ESI.2024.35.3.009
11. Geuzaine Ch., Remacle J.-F. Gmsh: A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering.* 2009;79(11):1309–1331. https://doi.org/10.1002/nme.2579

12. Geuzaine Ch. GetDP: A general finite-element solver for the de Rham complex. *PAMM.* 2007;7(1):1010603–1010604. https://doi.org/10.1002/pamm.200700750
13. Киселёв Е.А., Яровой А.В. Применение контейнерной виртуализации для реализации параллельных вычислений на суперкомпьютерных кластерах. *Программные продукты и системы.* 2026;(1):058–069. https://doi.org/10.15827/0236-235X.153.058-069
Kiselev E.A., Yarovoy A.V. Application of container virtualization for implementing parallel computing on supercomputer clusters. *Software & Systems.* 2026;(1):058–069. (In Russ.). https://doi.org/10.15827/0236-235X.153.058-069
14. Boettiger C.D. An introduction to Docker for reproducible research. *ACM SIGOPS Operating Systems Review.* 2014;49(1):71–79. https://doi.org/10.1145/2723872.2723882
15. Oprescu A.-M., Kielmann Th. Bag-of-Tasks Scheduling under Budget Constraints. In: *2010 IEEE Second International Conference on Cloud Computing Technology and Science, 30 November – 03 December 2010, Indianapolis, IN, USA.* IEEE; 2010. P. 351–359. https://doi.org/10.1109/CloudCom.2010.32
16. Ari I., Muhtaroglu N. Design and implementation of a cloud computing service for finite element analysis. *Advances in Engineering Software.* 2013;60-61(3-4):122–135. https://doi.org/10.1016/j.advengsoft.2012.10.003
17. Butrylo B., Musy F., Nicolas L., et al. A survey of parallel solvers for the finite element method in computational electromagnetics. *COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering.* 2004;23(2):531–546. https://doi.org/10.1108/03321640410510721
18. Shafiei H., Khonsari A., Mousavi P. Serverless Computing: A Survey of Opportunities, Challenges, and Applications. *ACM Computing Surveys.* 2022;54(11s). https://doi.org/10.1145/3510611

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Сурняев Виталий Андреевич, аспирант, Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова, Новочеркасск, Российская Федерация.
e-mail: vitaly.surnyaev@gmail.com

Vitalii A. Surnyaev, Postgraduate, Platov South-Russian State Polytechnic University (NPI), Novocherkassk, the Russian Federation.

Гречихин Валерий Викторович, доктор технических наук, доцент, профессор кафедры «Информационные и измерительные системы и технологии» Южно-Российского государственного политехнического университета (НПИ) имени М.И. Платова, Новочеркасск, Российская Федерация.
e-mail: vgrech@mail.ru

Valerii V. Grechikhin, Doctor of Engineering Sciences, Docent, Professor at the Department of Information and Measuring Systems and Technologies, Platov South-Russian State Polytechnic University (NPI), Novocherkassk, the Russian Federation.

Статья поступила в редакцию 01.03.2026; одобрена после рецензирования 24.04.2026; принята к публикации 04.05.2026.

The article was submitted 01.03.2026; approved after reviewing 24.04.2026; accepted for publication 04.05.2026.