ПРОБЛЕМНО-ОРИЕНТИРОВАННОЕ РАСШИРЕНИЕ СУБД POSTGRESQL ДЛЯ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ВРЕМЕННЫХ РЯДОВ*

А.И. Гоглачев, М.С. Подседов, М.Л. Цымблер, А.А. Юртин
Южно-Уральский государственный университет
(454080 Челябинск, пр. им. В.И. Ленина, д. 76),
goglachevai@susu.ru, mpodsedov@yandex.ru, mzym@susu.ru, iurtinaa@susu.ru

В работе представлено проблемно-ориентированное расширение свободной СУБД PostgreSQL, поддерживающее хранение и интеллектуальный анализ временных рядов. Данное расширение основано на внедрении в СУБД концепции матричного профиля временного ряда. Матричный профиль (МП) представляет собой структуру данных, которая резюмирует временной ряд, сохраняя для каждой подпоследовательности ряда индекс и расстояние до ее ближайшего соседа (подпоследовательности ряда, наиболее похожей на данную). МП служит основой для поиска различных аналитических примитивов временного ряда (диссонансы, сниппеты и др.) и решения различных задач, специфичных для конкретной предметной области (восстановление, прогноз рядов и др.). Предусматриваются таблицы для хранения МП и аналитических примитивов, найденных на их основе. Расширение предоставляет пользователю функционал визуализации рядов, МП и найденных на их основе аналитических примитивов. Реализация расширения допускает его использование в виде микросервиса.

Ключевые слова: интеллектуальный анализ данных, временные ряды, матричный профиль, PostgreSQL.

Введение

Интеллектуальный анализ данных (Data Mining) представляет собой совокупность методов и алгоритмов для обнаружения в данных ранее неизвестных и практически полезных знаний, используемых для принятия стратегически важных решений в различных сферах человеческой деятельности [1.]. Временные ряды являются одним из наиболее важных классов данных, подвергаемых интеллектуальному анализу. Под временным рядом (time series) понимают последовательность хронологически упорядоченных вещественных значений. Развитие

* *Работа выполнена при финансовой поддержке Российского научного фонда (грант № 23-21-00465).*

современных технологий и рост объемов данных в последние десятилетия привели к возникновению новых требований к обработке и анализу информации. Особенно важным стал анализ временных рядов, который находит применение в различных областях, таких как финансовые рынки, метеорология, медицина и многие другие.

В данной работе предлагается проблемно-ориентированное расширение СУБД PostgreSQL для автоматизации выполнения интеллектуального анализа временных рядов, которое направлено на решение указанных проблем.

1. Обзор связанных работ

Для интеллектуального анализа временных рядов есть множество различных готовых решений. Они предоставляют различные возможности, такие как интеллектуальный анализ, хранение, визуализация и их комбинации.

Stumpy [2.] – это мощная и масштабируемая библиотека Python, она эффективно вычисляет МП, который может использоваться для различных задач интеллектуального анализа данных временных рядов. Данная библиотека имеет множество различных методов для поиска примитивов. С помощью Stumpy можно найти мотивы, диссонансы, цепочки и многое другое.

Matplotlib [3.] – это библиотека для создания графиков и визуализации данных на языке программирования Python, используемая для создания статических, анимированных и интерактивных визуализаций. Основная цель Matplotlib – предоставить пользователям разнообразные инструменты для создания качественных графиков различных типов, начиная от простых линейных до сложных трехмерных и функциональность для графического представления данных, упрощая их анализ и понимание. В настоящее время поддерживается большим сообществом разработчиков.

InfluxDB [4.] – это распределенная временная база данных, специализированная на хранении и обработке временных данных. Она широко используется в системах мониторинга, аналитике временных рядов и других приложениях, где требуется эффективное управление временными данными. InfluxDB доступна на всех популярных языках и фреймворках, что позволяет разработчикам приступить к работе за считанные минуты и располагает крупнейшим сообществом разработчиков облачных решений и решений с открытым исходным кодом для баз данных временных рядов. Позволяя сообществу вносить свой вклад в другие экосистемы и интегрироваться с ними, InfluxDB обладает большей

устойчивостью, качеством, возможностью использования и конфигурирования во всех вариантах использования.

ClickHouse [5.] – высокопроизводительная, ориентированная на столбцы база данных с открытым исходным кодом, разработанная для обработки больших объемов данных и для оперативной аналитической обработки, которая использует все доступные системные ресурсы в полной мере для максимально быстрой обработки каждого аналитического запроса. Она доступна как в виде программного обеспечения с открытым исходным кодом, так и в виде облачного приложения. TimescaleDB [24] – это расширение для базы данных PostgreSQL, специализированное на хранение временных данных.

TimescaleDB объединяет мощность реляционной базы данных PostgreSQL с возможностями хранения и анализа временных рядов. Благодаря своей специализации на временных данных, TimescaleDB обеспечивает высокую производительность и масштабируемость при работе с данными, обладая при этом всеми преимуществами PostgreSQL.

Graphite [6.] – это инструмент для сбора и визуализации метрик и временных рядов, который одинаково хорошо работает на дешевом оборудовании или облачной инфраструктуре. Он обладает мощными возможностями агрегации данных, построения графиков для мониторинга производительности системы.

Azure Time Series Insight (TSI) [7.] – это сервис временных рядов в облаке от Microsoft Azure, предназначенный для хранения, анализа и визуализации временных данных. Azure TSI обеспечивает возможности работы с большими объемами данных, мониторинга и аналитики. С помощью Azure TSI пользователи могут проводить глубокий анализ данных временных рядов, исследовать тренды, выявлять аномалии и принимать оперативные решения на основе этих данных.

2. Теоретический базис

Временной ряд представляет собой хронологически упорядоченную последовательность числовых значений, которую можно определить формально следующим образом:

$$T = (t_1, \dots, t_n), t_i \in R.$$

Подпоследовательность $T_{i,m}$ временного ряда T представляет собой непрерывное подмножество T, состоящее из m элементов и начинающееся с позиции i, формально определяется следующим образом:

$$T_{i,m} = (t_i, \dots, t_{i+m-1}),$$

где $1 \le i \le n-m+1, 1 \le m \ll n$.

Множество всех подпоследовательностей S_T^A временного ряда T является упорядоченным набором всех возможных подпоследовательностей длины m, содержащихся в T. Подпоследовательности сортируются в порядке возрастания индекса их первого элемента:

$$S_T^A = \left(T_{1,m}, T_{2,m}, \dots, T_{n-m+1,m} \right).$$

Профиль расстояния D_i^T представляет собой вектор расстояний между заданной подпоследовательностью $T_{i,m}$ и каждой подпоследовательностью T_j, m из множества всех подпоследовательностей:

$$D_i^T = \left(dist\left(T_{i,m}, T_{1,m} \right), \dots, dist\left(T_{i,m}, T_{n-m+1,m} \right) \right),$$

где $dist(,)$ означает евклидово расстояние между нормализованными подпоследовательностями.

С помощью профиля расстояния выполняется поиск ближайшего соседа для каждой подпоследовательности временного ряда, за исключением тривиальных соседей. Подпоследовательность $T_{j,m}$ называется ближайшим соседом подпоследовательности$T_{i,m}$:

$$dist\left(T_{i,m}, T_{j,m} \right) = min\left(D_i^T \right).$$

Подпоследовательность $T_{j,m}$ является тривиальным соседом для $T_{i,m}$, если $¿ i-j \vee \le \frac{m}{2}$.

Пусть неотрицательная симметричная функция используется в качестве функции расстояния:

$$dist : R^m \times R^m \to R.$$

Матричный профиль (МП) P^T временного ряда T представляет собой вектор расстояний между каждой подпоследовательностью $T_{i,m}$ и ее ближайшим нетривиальным соседом:

$$P^T = \left(nn_{nt}\left(D_1^T \right), \dots, nn_{nt}\left(D_{n-m+1}^T \right) \right),$$

где $nn_{nt}\left(D_i^T \right)$ означает минимальное расстояние между $T_{i,m}$ и ее нетривиальными соседями [8., 9.].

С МП ассоциируется дополнительная структура данных для определения местоположения ближайших нетривиальных соседей. Индекс МП профиля I^T временного ряда T представляет собой вектор, хранящий индекс ближайшего нетривиального соседа для каждой подпоследовательности:

$$I^T = \left(I_1^T, \dots, I_{n-m+1}^T \right),$$

где $I_j^T = j$, если $nn_{nt}\left(D_i^T \right) = dist\left(T_{i,m}, T_{j,m} \right)$.

МП представляет собой агностический (не зависящий от предметной области) инструмент интеллектуального анализа временного ряда, в котором исследователь должен указать лишь один параметр – длину подпоследовательности. Его можно рассматривать как метаданные для аннотирования соответствующего временного ряда.

МП позволяет выполнять поиск мотивов (шаблонов), который предполагает нахождение пар непересекающихся подпоследовательностей временного ряда, наиболее похожих друг на друга и формально определяется следующим образом. Пара подпоследовательностей $\{T_{i,m}, T_{j,m}\}$ ряда T называется мотивом, если они имеют наибольшую схожесть по сравнению с другими парами подпоследовательностей $(T_{a,m}, T_{b,m})$ этого временного ряда:

$$\forall a, b, |a-b| \geq \omega, i, j, |i-j| \geq \omega, \omega > 0,$$
$$dist(T_{i,m}, T_{j,m}) \leq dist(T_{a,m}, T_{b,m}),$$

где ω – параметр, определяющий минимальный промежуток, на который должны отстоять друг от друга подпоследовательности в мотиве. Данный параметр позволяет отбрасывать мотивы, которые состоят из взаимопересекающихся подпоследовательностей и потому не имеют практической ценности.

Поиск аномалий предполагает обнаружение подпоследовательностей временного ряда, которые наиболее непохожи на все остальные подпоследовательности ряда. Концепция диссонанса (discord), предложенная в работе [10.], уточняет и формализует понятие аномалии и в настоящее время признается научным сообществом как наиболее адекватный способ поиска аномалий во временном ряде. Диссонанс определяется следующим образом. Подпоследовательности $T_{i,m}$ и $T_{j,m}$ ряда T называются непересекающимися, если $¿i-j \vee \geq m$. Подпоследовательность, которая является непересекающейся к данной подпоследовательности $T_{i,m}$, обозначается как $M_{T_{i,m}}$. Подпоследовательность $T_{i,m}$ является диссонансом, если:

$$\forall T_{j,m}, M_{T_{i,m}} \in T, min\left(dist\left(T_{i,m}, M_{T_{i,m}}\right)\right) > min\left(dist\left(T_{j,m}, M_{T_{j,m}}\right)\right) \quad .$$

Иными словами, диссонанс представляет собой подпоследовательность ряда, имеющую максимальное расстояние до наиболее близкой к ней непересекающейся подпоследовательности.

В работах [11., 12.] предложено понятие сниппета (snippet). Поиск сниппетов предполагает нахождение значимых и информативных фрагментов данных, которые отображают основные характеристики временного ряда. Концепция сниппета уточняет понятие типичной

подпоследовательности временного ряда следующим образом. Временной ряд T может быть разбит на сегменты заданной длины m, где каждый сегмент S_i представляет собой подпоследовательность ряда:

$$S_i = T_{m\cdot(i-1)+1,\,m},$$

где $1 \le i \le \frac{n}{m}$.

Каждый сниппет представляет собой один из сегментов временного ряда. Со сниппетом ассоциируются его ближайшие соседи – подпоследовательности ряда, имеющие ту же длину, что и сниппет, которые более похожи на данный сниппет, чем на другие сегменты. Для вычисления схожести подпоследовательностей используется специализированная мера схожести, основанная на евклидовом расстоянии. Сниппеты упорядочиваются по убыванию мощности множества своих ближайших соседей.

Для нахождения сниппетов используется алгоритм Snippet-Finder предложенный в работе [13.], который включает в себя MPdist для поиска и сравнения сниппетов во временных рядах. Основой алгоритма являются методы анализа временных рядов, статистические меры и пороговые значения, которые позволяют определить, является ли сегмент сниппетом или нет.

Левый профиль расстояний DL_i временного ряда T представляет собой вектор евклидовых расстояний между заданной подпоследовательностью $T_{i,m}$ и каждой подпоследовательностью, которая появляется до нее во временном ряде:

$$DL_i = [d_{i,1}, d_{i,2}, \ldots, d_{i,l-m/4}],$$

где $d_{i,j} = ED_{norm}(T_{i,m}, T_{j,m}) = dist(T_{i,m}, T_{j,m})$.

Правый профиль расстояний DR_i временного ряда T представляет собой вектор евклидовых расстояний между заданной подпоследовательностью $T_{i,m}$ и каждой подпоследовательностью, которая появляется после нее во временном ряде:

$$DR_i = [d_{i,i+m/4}, d_{i,i+m/4+1}, \ldots, d_{i,n-m+1}].$$

Ближайший сосед слева $LNN(T_{i,m})$это подпоследовательность, которая появляется до $T_{i,m}$ во временном ряде и наиболее похожа на нее:

$$LNN(T_{i,m}) = T_{j,m},$$

если $d_{i,j} = min(DL_i)$.

Ближайший сосед слева $RNN(T_{i,m})$это подпоследовательность, которая появляется после $T_{i,m}$ во временном ряде и наиболее похожа на нее:

$$RNN(T_{i,m}) = T_{j,m},$$

если $d_{i,j} = min(DR_i)$.

Левый МП PL временного ряда T представляет собой вектор z-нормированного евклидова расстояния между каждой подпоследовательностью $T_{i,m}$ и ее левым ближайшим соседом во временном ряду:

$$PL = \left[min(DL_1), min(DL_2), \ldots, min(DL_{n-m+1}) \right],$$

где $DL_i (1 \le i \le n-m+1)$.

Индекс левого МП IL представляет собой вектор целых чисел и формально определяется следующим образом:

$$IL = \left[IL_1, IL_2, \ldots, IL_{n-m+1} \right],$$

где $IL_i = j$, если $LNN(T_{i,m}) = T_{j,m}$.

Правый МП PR временного ряда T представляет собой вектор z-нормированного евклидова расстояния между каждой подпоследовательностью $T_{i,m}$ и ее правым ближайшим соседом во временном ряду:

$$PR = \left[min(DR_1), min(DR_2), \ldots, min(DR_{n-m+1}) \right],$$

где $DR_i (1 \le i \le n-m+1)$.

Индекс левого МП IR представляет собой вектор целых чисел и формально определяется следующим образом:

$$IR = \left[IR_1, IR_2, \ldots, IR_{n-m+1} \right],$$

где $IR_i = j$, если $RNN(T_{i,m}) = T_{j,m}$.

В работе [28] представлено определение цепочки (chain) и алгоритмы ее поиска. Формальное определение цепочки представлено в формуле (20). Цепочка временного рядя T представляет собой упорядоченное множество подпоследовательностей:

$$TSC = \left\{ T_{C1,m}, T_{C2,m}, \ldots, T_{Ck,m} \right\},$$

где $\forall i \in 1 \ldots k (C1 \le C2 \le \ldots \le Ck)$, есть $RNN(T_{Ci,m}) = T_{C(i+1),m}$ и $LNN(T_{C(i+1),m}) = T_{Ci,m}$, k – длина цепочки.

Цепочки подразделяются на закрепленные и незакрепленные. Незакрепленная цепочка – это самая длинная цепочка во временном ряде, и она единственная, а закрепленных цепочек может быть много.

Множество всех цепочек (All-Chain Set) S – это множество всех закрепленных цепочек, каждая из которых не включена в какую-либо другую цепочку.

3. Реализация расширения СУБД PostgreSQL для интеллектуального анализа временных рядов

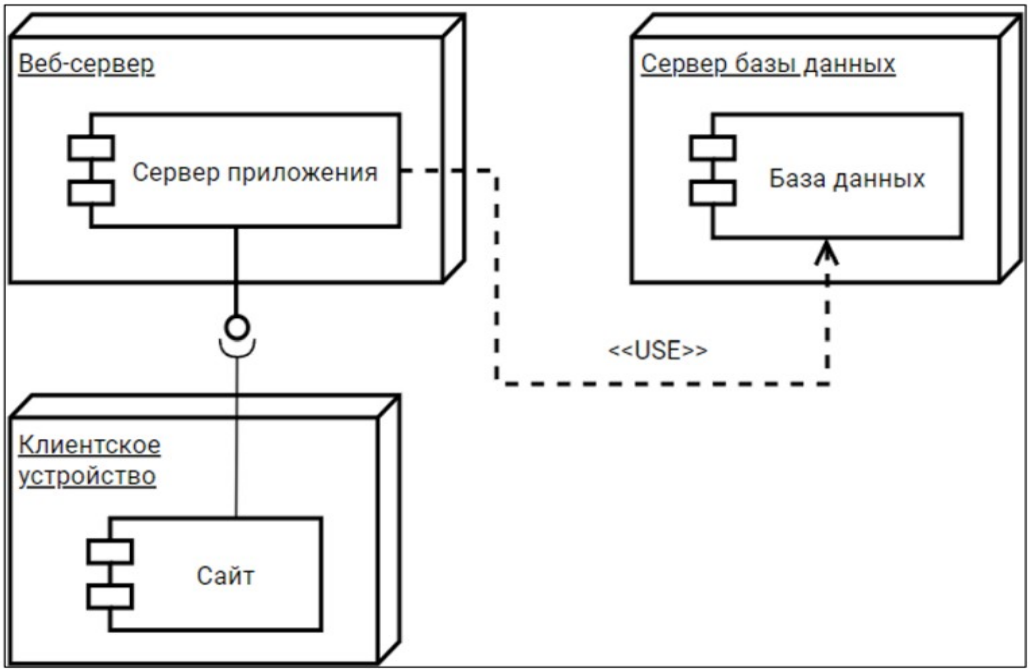

Рис. 1. – Архитектура системы

Взаимодействие с расширением СУБД осуществляется посредством разработанного веб-приложения, обеспечивающего пользовательский интерфейс. Система включает в себя следующие основные функции: загрузка временного ряда в проект из файла формата .txt, отображение временных рядов на графике, поиск примитивов временного ряда, отображение найденных примитивов на графиках.

3.1. Архитектура системы

На Error: Reference source not found представлена архитектура системы. Система состоит из двух частей: клиентской и серверной. *Серверная часть* отвечает за обработку запросов, приходящих с клиентских устройств. Обработка запросов в большинстве случаев включает в себя взаимодействие сервера с базой данных, хранящей информацию, доступную для просмотра на сайте. *Клиентская часть* отвечает за вывод информации, полученной от сервера, в виде веб-страниц в браузере пользователя, с интерфейсом которых он имеет возможность взаимодействовать

3.2. Структура базы данных

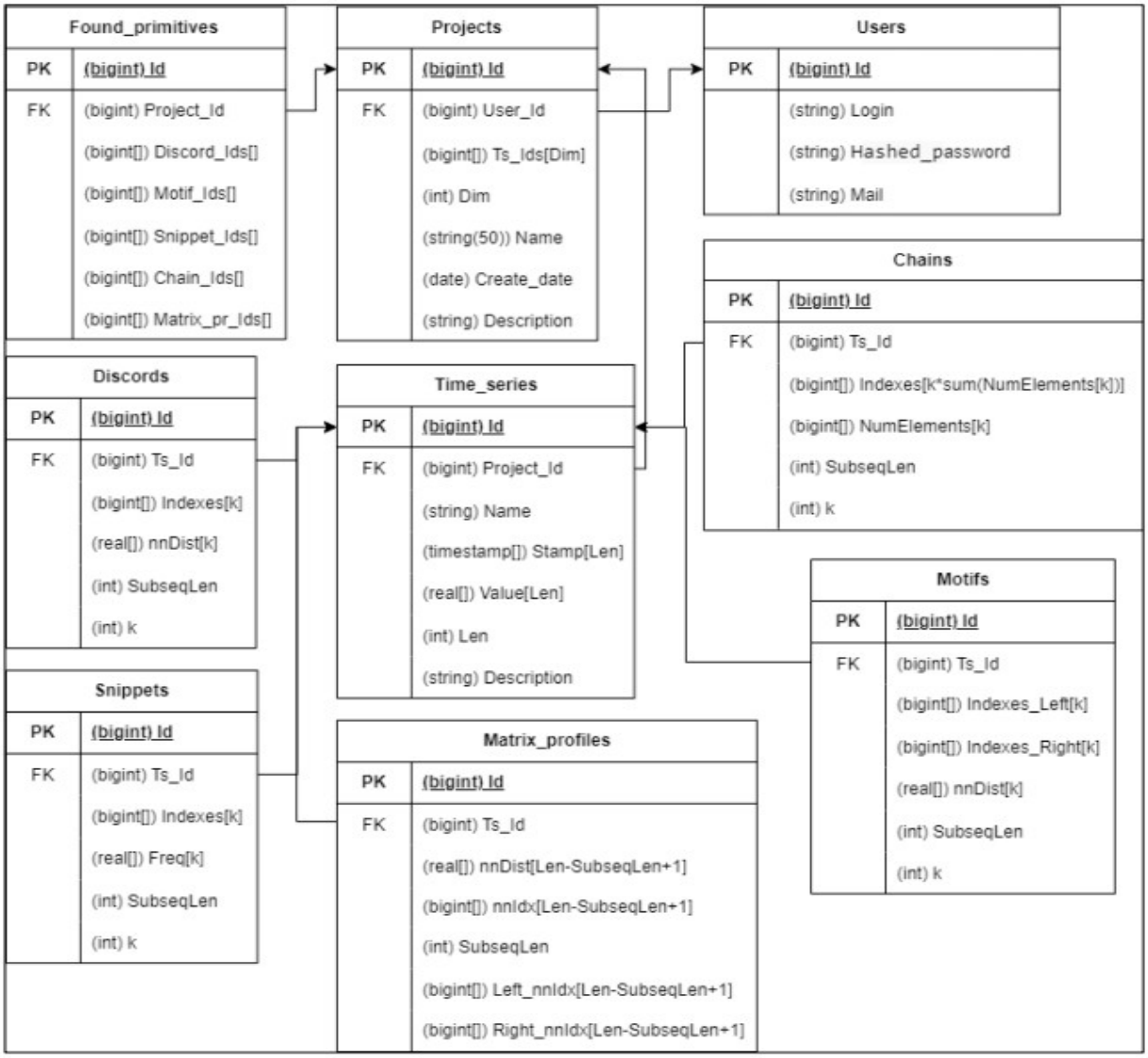

Рис. 2. – Схема базы данных

База данных веб-приложения хранит данные пользователя, проекта, временного ряда, МП, мотива, диссонанса, сниппета, цепочки, найденных примитивов и имеет структуру, которая представления на Error: Reference source not found.

Таблица «Users» предназначена для хранения информации о зарегистрированных пользователях. Она содержит следующие столбцы: уникальный идентификатор, логин, хешированный пароль и адрес электронной почты.

Таблица «Projects» предназначена для хранения информации о созданных проектах пользователей. Она содержит следующие столбцы: уникальный идентификатор, размер временного ряда (если 1, то одномерный ряд, если больше, то многомерный), название, дату создания,

описание, внешний ключ на пользователя, которому принадлежит проект, массив внешних ключей на координаты временного ряда.

Таблица «Time_series» предназначена для хранения информации о временных рядах. Если ряд одномерный, то одна строка хранит информацию о временном ряде, но если ряд многомерный, то одна строка хранит информацию о координатах временного ряда. Она содержит следующие столбцы: уникальный идентификатор, название, массив временных меток, массив значений, длину и описание, а также внешний ключ на проект.

Таблица «Matrix_profiles» предназначена для хранения информации о найденных матричных профилях временных рядов. Она содержит следующие столбцы: уникальный идентификатор, массив расстояний до ближайших соседей, массив индексов ближайших соседей, массив индексов левых ближайших соседей, массив индексов правых ближайших соседей, длину подпоследовательности, а также внешний ключ на координаты временного ряда.

Таблица «Motifs» предназначена для хранения информации о найденных мотивах. Она содержит следующие столбцы: уникальный идентификатор, массивы индексов левой и правой части мотивов, массив расстояний между частями, длину подпоследовательности, количество, а также внешний ключ на координаты временного ряда. Каждая строка таблицы хранит информацию о top-*k* мотивах временного ряда с заданной длиной подпоследовательности.

Таблица «Discords» предназначена для хранения информации о найденных диссонансах. Она содержит следующие столбцы: уникальный идентификатор, массив индексов начала, массив расстояний до ближайших соседей, длину подпоследовательности, количество, а также внешний ключ на координаты временного ряда. Каждая строка таблицы хранит информацию о top-*k* диссонансах временного ряда с заданной длиной подпоследовательности.

Таблица «Snippets» предназначена для хранения информации о найденных сниппетах. Она содержит следующие столбцы: уникальный идентификатор, массив индексов начала, массив покрытий, длину подпоследовательности, количество, а также внешний ключ на координаты временного ряда. Каждая строка таблицы хранит информацию о top-*k* сниппетах временного ряда с заданной длиной подпоследовательности.

Таблица «Chains» предназначена для хранения информации о найденных цепочках. Она содержит следующие столбцы: уникальный идентификатор, массив индексов начала, массив количества элементов,

длину подпоследовательности, количество, а также внешний ключ на координаты временного ряда. Каждая строка таблицы хранит информацию о top-k цепочках временного ряда с заданной длиной подпоследовательности.

Таблица «Found_primitives» предназначена для хранения информации о найденных примитивах. Она содержит следующие столбцы: уникальный идентификатор, внешний ключ на проект, массив внешних ключей найденных диссонансов, массив внешних ключей найденных мотивов, массив внешних ключей найденных сниппетов, массив внешних ключей найденных цепочек, массив внешних ключей найденных матричных профилей.

3.3. Реализация API

API – Application Programming Interface, что значит программный интерфейс приложения. API называется определенный набор протоколов, подпрограмм и инструментов для создания программным приложений. Основной особенностью API является его способность настраивать взаимодействие компонентов различных информационно-аналитических систем, т.е. он обеспечивает эффективный процесс коммуникаций между программами, которые используют функции и ресурсы друг друга.

Допускается два способа реализации взаимодействия между клиентским приложением и сервером: с использованием фреймворка FastApi, который соответствует архитектурному стилю REST (REpresentational State Transfer), или с использованием системы удаленного вызова процедур gRPC (Remote Procedure Call). Для реализации данного расширения был выбран фреймворк FastApi, так как он предоставляет более быструю интеграцию новых функций поиска примитивов по сравнению с gRPC, и используемый им протокол HTTP (HyperText Transfer Protocol) является более широко поддерживаемым. Архитектурно в системе можно выделить 3 слоя: слой репозитория, слой бизнес-логики и слой представлений. Одними из основных задач API является загрузка временных рядов и поиск примитивов.

Данная реализация позволяет использовать расширение в качестве микросервиса для внедрения в уже существующие системы. В данном случае работа с расширением будет реализована при помощи взаимодействия с его API. API будет предоставлять системе доступ к функциям интеллектуального анализа временных рядов (вычисление МП и поиск примитивов).

Для поиска аналитических примитивов используется библиотека Stumpy. Поиск примитивов происходит в несколько этапов. На первом этапе будет получен список примитивов, который содержит следующие данные: ID координаты, для которой нужно найти примитив, название примитива, длина подпоследовательности и число, обозначающее количество искомых примитивов. После этого создается словарь, где ключ – это первая буква примитива, а значением является список примитивов из полученных данных, которые необходимо найти. Затем из базы данных будут получены координаты временного ряда по id координат, для этих координат получаем матричные профили и найденные примитивы (например, если хотим найти top-*k* диссонансов, то из базы берем только найденные диссонансы). После берем значение по каждому ключу словаря и в цикле проверяем найден ли такой примитив ранее, если примитив был найден ранее, тогда берем следующий примитив. Иначе проверяем найден ли МП с такой длинной подпоследовательности, если он был найден, то используем его, иначе вычисляем МП. Затем выполняется поиск примитива, в конце все найденные примитивы записываются в базу данных.

Запросы к базе данных осуществляются с помощью ORM SQLAlchemy [14.]. SQLAlchemy – это библиотека, которая облегчает взаимодействие между программами на Python и базами данных. В большинстве случаев эта библиотека используется как средство объектно-реляционного отображения (ORM), которое преобразует классы Python в таблицы в реляционных базах данных и автоматически преобразует вызовы функций в инструкции SQL. Для миграций используется библиотека Alembic [15.].

В качестве метода аутентификации используется аутентификация с помощью JWT-токенов. JWT-токены – это открытый стандарт для создания токенов доступа, основанный на формате JSON. Как правило, используется для передачи данных для аутентификации в клиент-серверных приложениях. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для подтверждения своей личности.

3.4. Реализация пользовательского интерфейса

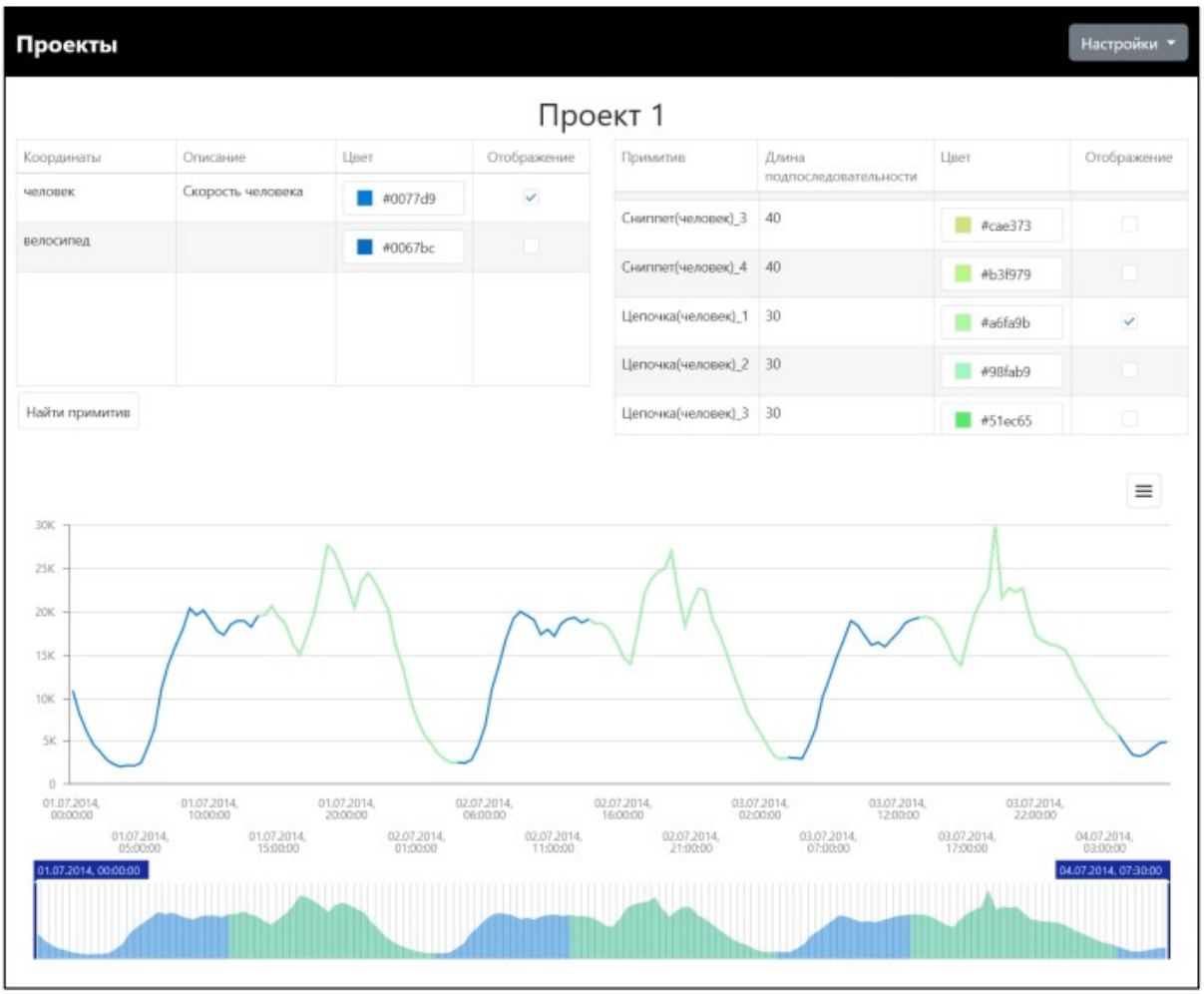

Рис. 3. – Форма визуализации найденных аналитических примитивов

Для реализации оболочки был использован фреймворк Vue.js [16.]. Все графики, формы и валидация форм были реализованы с помощью библиотеки компонентов DevExtreme [17.]. На Рис. 3 представлена форма для визуализации найденных примитивов, содержащая пример найденной цепочки. Таблица в левой верхней части формы позволяет скрывать координаты ряда или задавать им пользовательские цвета. Таблица в правой верхней части формы предоставляет аналогичные возможности для найденных примитивов ряда. Пользователю также доступно масштабирование графика и изменения диапазона отображения меток. Присутствует возможность сохранения графика в различных форматах и его печати.

4. Экспериментальное исследование

В данном разделе представлены результаты вычислительных экспериментов, исследующих зависимость времени выполнения поиска аналитических примитивов от длины подпоследовательности.

Исследование проводилось на временном ряде, содержащем информацию о среднем числе пассажиров Нью-Йоркского такси за осень 2014 г., который содержит 10 320 точек.

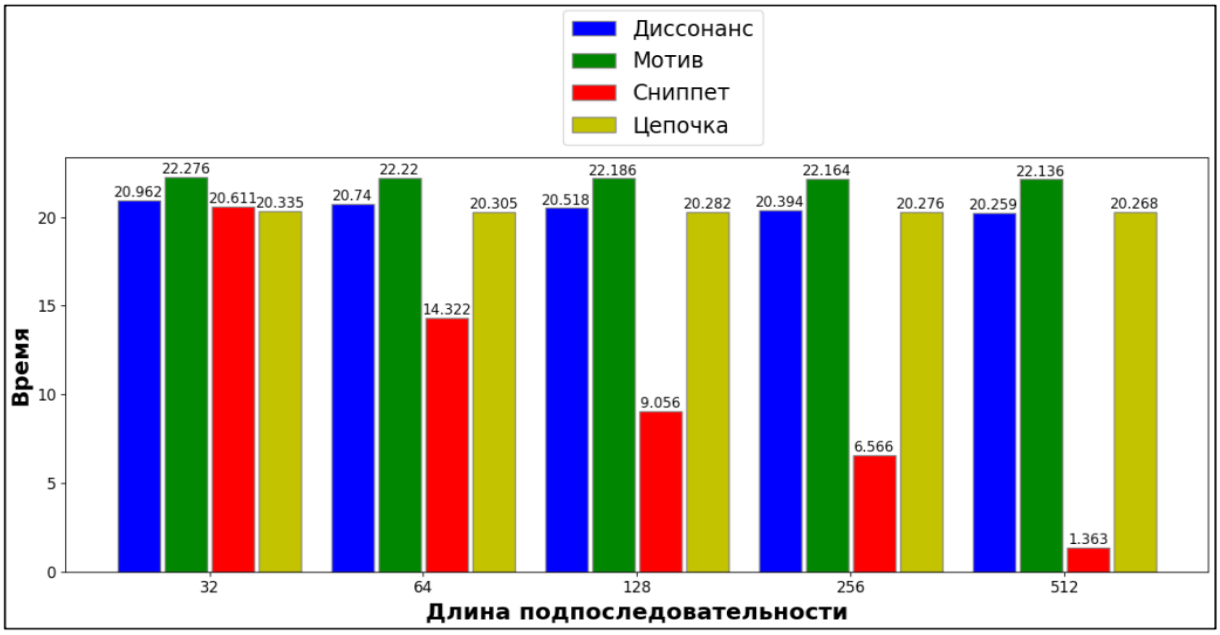

Рис. 4. Время выполнения при отсутствии вычисленного МП

На Рис. 4 представлена столбчатая диаграмма, на которой показана зависимость времени выполнения функции поиска примитива от длины подпоследовательности и примитива, когда МП не был вычислен.

На Рис. 5 представлена столбчатая диаграмма, на которой показана зависимость времени выполнения функции поиска примитива от длины подпоследовательности и примитива, когда МП был уже вычислен. Исходя из результатов, видно, что предвычисленный МП дает выигрыш во времени поиска примитивов (за исключением сниппетов, так как алгоритм их поиска не предусматривает использование предвычисленного МП).

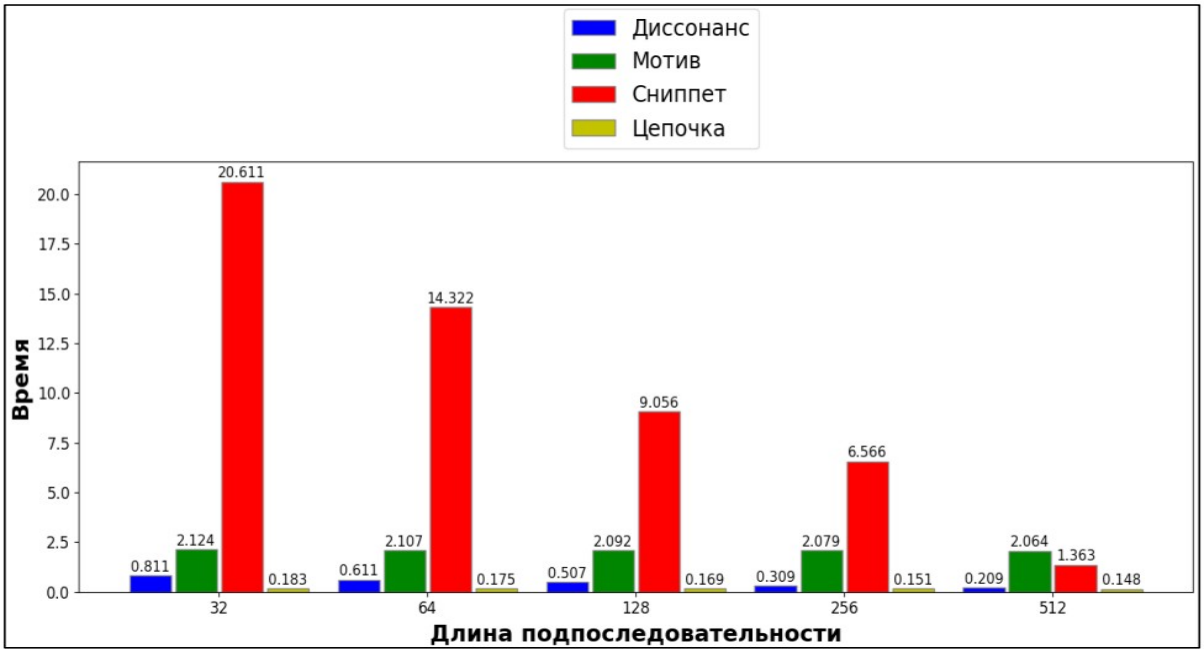

Рис. 5. Время выполнения с предвычисленным МП

Заключение

В статье представлено проблемно-ориентированное расширение свободной СУБД PostgreSQL, поддерживающее хранение и интеллектуальный анализ временных рядов. Данное расширение основано на внедрении в СУБД концепции матричного профиля (МП) временного ряда. МП служит основой для поиска различных аналитических примитивов временного ряда (диссонансы, сниппеты и др.) и решения различных задач, специфичных для конкретной предметной области (восстановление, прогноз рядов и др.). Была спроектирована и реализована база данных для хранения МП и аналитических примитивов, найденных на их основе. Расширение предоставляет пользователю функционал визуализации рядов, МП и найденных на их основе аналитических примитивов. Были проведены эксперименты по оценке производительности алгоритмом поиска примитивов, которые используются в расширении. Реализованное расширение может быть использовано в качестве микросервиса для внедрения в уже существующие системы. В дальнейшей разработке расширения планируется добавление новых примитивов интеллектуального анализа данных и реализация более гибкой настройка поиска примитивов.

Исходный код оболочки для автоматизации выполнения интеллектуального анализа временных рядов доступен по адресу https://github.com/yurtinaa/TSDB.

Литература

1. Shukla R.K., Sharma P., Samaiya N., Kherajani M. Web Usage Mining-A Study of Web data pattern detecting methodologies and its applications in 44 Data Mining. // 2nd International Conference on Data, Engineering and Applications (IDEA), Bhopal, India, 2020. – 1–6 pp
2. Stumpy. [Электронный ресурс] URL: https://pypi.org/project/stumpy/ (дата обращения: 11.01.2024 г.).
3. Matplotlib. [Электронный ресурс] URL: https://pypi.org/project/matplotlib/ (дата обращения: 11.08.2024 г.).
4. Influx data. [Электронный ресурс] URL: https://www.influxdata.com/ (дата обращения: 12.01.2024 г.).
5. ClickHouse. [Электронный ресурс] URL: https://dbengines.com/en/system/ClickHouse (дата обращения: 12.08.2024 г.).
6. Graphite. [Электронный ресурс] URL: https://graphiteapp.org/ (дата обращения: 16.08.2024 г.).
7. Azure Time Series Insights. [Электронный ресурс] URL: https://www.sqlshack.com/introduction-to-azure-time-series-insights/ (дата обращения: 16.01.2024 г.).
8. Zhu Y., Gharghabi S., Silva D.F., Dau H.A., Yeh C.-C.M., Senobari N.S., Almaslukh A., Kamgar K., Zimmerman Z., Funning G., Mueen A., Keogh E. The Swiss army knife of time series data mining: Ten useful things you can do with the matrix profile and ten lines of code. // Data Min. Knowl. Discov., 2020. – 949–979 pp.
9. Иванова Е.В., Цымблер М.Л. Внедрение концепции матричного профиля в реляционную СУБД для интеллектуального анализа временных рядов. // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика, 2021. – Т. 10, № 3. – С. 72–87.
10. Keogh E., Lin J., Fu A. HOT SAX: efficiently finding the most unusual time series subsequence. // Proceedings of the 5th IEEE International Conference on Data Mining, ICDM’05 (Houston, Texas, November, 27–30, 2005), 2005. – 8 pp.
11. Imani S., Madrid F., Ding W., Crouter S.E., Keogh E.J. Introducing time series snippets: a new primitive for summarizing long time series. // Data Min. Knowl. Discov, 2020. – 1713–1743 pp.
12. Imani S., Madrid F., Ding W., Crouter S.E., Keogh E.J., Wu X., Ong Y., Aggarwal C.C., Chen H. Matrix Profile XIII: Time Series Snippets: A New Primitive for Time Series Data Mining. // 2018 IEEE International Conference on Big Knowledge, ICBK 2018, Singapore, November 17–18, 2018, IEEE Computer Society, 2018. – 382–389 pp.
13. Zhu Y., Imamura M., Nikovski D., Keogh E., Matrix Profile VII: Time Series Chains: A New Primitive for Time Series Data Mining (Best Student Paper Award). // 2017 IEEE International Conference on Data Mining (ICDM), New Orleans, LA, USA, 2017. – 695–704 pp.
14. SqlAlchemy. [Электронный ресурс] URL: https://docs.sqlalchemy.org/en/20/orm/ (дата обращения: 20.07.2024 г.)
15. Alembic. [Электронный ресурс] URL: https://pypi.org/project/alembic/ (дата обращения: 15.06.2024 г.).
16. Vue.js - The Progressive JavaScript Framework [Электронный ресурс] URL: https://vuejs.org/ (дата обращения: 10.06.2024 г.).
17. DevExtreme - JavaScript UI Components [Электронный ресурс] URL: https://js.devexpress.com/ (дата обращения: 14.06.2024 г.)