Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 21.Б07-мм

Добавление новых примитивов в веб-версию Desbordante

СОЛОВЬЁВА Лиана-Юлия Викторовна

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
ассистент кафедры информационно-аналитических систем Г. А. Чернышев

Санкт-Петербург
2024

Оглавление

Введение

В современном мире человек ежедневно сталкивается с большим количеством информации: истории знакомых, конспекты лекций, реклама в автобусе и многое другое. Более того, существуют профессии, которые заключаются в работе с большими массивами данных. И под работой с данными подразумевается не только их хранение, но и выявление в них закономерностей и зависимостей – метаданных.

Человек не в силах быстро справиться с этой задачей. На помощь приходит Desbordante – профайлер данных, который позволяет извлекать метаданные из таблиц с помощью различных примитивов – наборов алгоритмов, показывающих зависимости в данных.

Одним из интерфейсов, через который пользователь может взаимодействовать с Desbordante, является веб-приложение [4].

На момент написания данной работы в веб-версии Desbordante доступны примитивы:

- Functional Dependencies
- Conditional Functional Dependencies
- Association Rules
- Error Detection Pipeline
- Metric Dependency Verification

Команда продолжает развивать проект, добавляя новые алгоритмы и примитивы, в частности, в веб-приложение.

1. Постановка задачи

Целью работы является написание фронтенд-части для добавления в веб-версию Desbordante примитивов Algebraic Constraints (AC) и Approximate Functional Dependencies (AFD). Для её выполнения были поставлены следующие задачи:

1. Ознакомиться с алгоритмами, реализующими примитивы;
2. Создать макеты веб-страниц в Figma;
3. Сверстать веб-страницы.

2. Обзор используемых инструментов

Фронтенд проекта написан на TypeScript. Это язык программирования, разработанный компанией Microsoft, который представляет собой надмножество JavaScript, добавляющее статическую типизацию. Он помогает улучшить качество кода, делая его более надежным и понятным.

React – популярная библиотека JavaScript для создания пользовательских интерфейсов. Она позволяет разрабатывать масштабируемые веб-приложения с использованием компонентного подхода.

В качестве менеджера состояний выбран Jotai, который обеспечивает простой и мощный способ управления состоянием приложения, используя атомарный подход.

Передача данных между фронтенд и бэкенд-частью веб-приложения происходит с использованием языка запросов GraphQL, который позволяет клиентам запрашивать только те данные, которые им нужны.

Для добавления компонентов-полей ввода в проекте применяется библиотека React Hook Form, которая предлагает простой и гибкий способ работы с формами, обеспечивая удобную валидацию.

Для стилизации компонентов выбран язык написания стилей SCSS, который компилируется в CSS, но позволяет создавать более эффективные и легко поддерживаемые стили.

Вышеперечисленные инструменты использовались в проекте и ранее. Однако есть и нововведение. Библиотека, не используемая ранее в проекте – Recharts. Она предназначается для визуализации данных в виде графиков. Recharts обладает простым и понятным API, хорошо интегрируется с React. Кроме того, примеры использования и документация на официальном сайте [5] помогают облегчить написание кода.

Графики, а именно pie charts, использовались в веб-приложении и ранее в примитиве Functional Dependencies, однако их визуализация происходила с помощью библиотеки Chart.js. Основное отличие Recharts заключается в большем количестве возможностей для создания индивидуального дизайна графика, что необходимо для создания гистограммы для примитива Algebraic Constraints.

3. Основные понятия

3.1. Algebraic Constraints

Algebraic Constraints – отношение вида $a_1 \oplus a_2 \in I$, где $a_1,\ a_2$ – атрибуты численного типа, $\oplus \in \{+, -, *, /\}$, I – подмножество множества вещественных чисел. Также есть P – правило, которое определяет, как значения одного атрибута будут образовывать пары со значениями второго.

В Desbordante поиск алгебраических ограничений осуществляется с помощью алгоритма BHUNT, реализованного ранее другим студентом.

Подробнее можно почитать в статье “BHUNT: Automatic Discovery of Fuzzy Algebraic Constraints in Relational Data”[1] и в отчете по учебной практике[6].

3.2. Approximate Functional Dependencies

Approximate Functional Dependencies идейно соотвествует точному поиску зависимостей в данных, однако допускается погрешность, т.е. не для всех строк таблицы данная зависимость будет выполняться.

В Desbordante валидация функциональных зависимостей осуществляется с помощью алгоритма Naive AFD Verifier, реализованного ранее другим студентом.

Подробнее можно почитать в статье “Efficient discovery of approximate dependencies”[2].

При описании результата AFD также используется понятие кластер – набор кортежей, имеющих одинаковые значения на заданных атрибутах.

4. Реализация

Работа над добавлением новых примитивов в веб-версию Desbordante началась с создания макета будущих веб-страниц в графическом онлайн-редакторе Figma.

Удобно при написании веб-страниц видеть то, что требуется создать. Поскольку дизайн веб-страниц нужно согласовывать с научным руководителем, было бы очень трудоемко несколько раз переписывать код в поисках идеального размера и расположения элементов. Кроме того, иногда появляется сразу несколько идей для визуализации страницы, и окончательное решение принимается только после рассмотрения всех вариантов. Такая ситуация произошла с Instance list для Algebraic Constraints.

Взаимодействие пользователя с Desbordante с целью получения метаданных состоит из нескольких шагов:

1. Выбор примитива
2. Выбор файла
3. Выбор алгоритма и его конфигураций
4. Просмотр результата работы алгоритма

Добавление новых примитивов затрагивает все приведенные выше шаги.

Ключевой особенностью данной работы является отсутствие бэкенда.

4.1. Выбор примитива

Работа по добавлению примитива на данном шаге тривиальна: достаточно указать в соответствующем файле название алгоритма и задать его описание, которое появляется при выборе пользователем конкретного примитива.

Дополнительно были внесены небольшие изменения: раньше описание примитива было просто типом string, а в случае необходимости добавление ссылки на статью с более подробным описанием, ссылка указывалась отдельным параметром и вставлялась в конец. Теперь же описание является элементом типа ReactNode, в котором сразу можно добавить ссылки в любом месте, а также стилизовать текст, например, вывод списка.

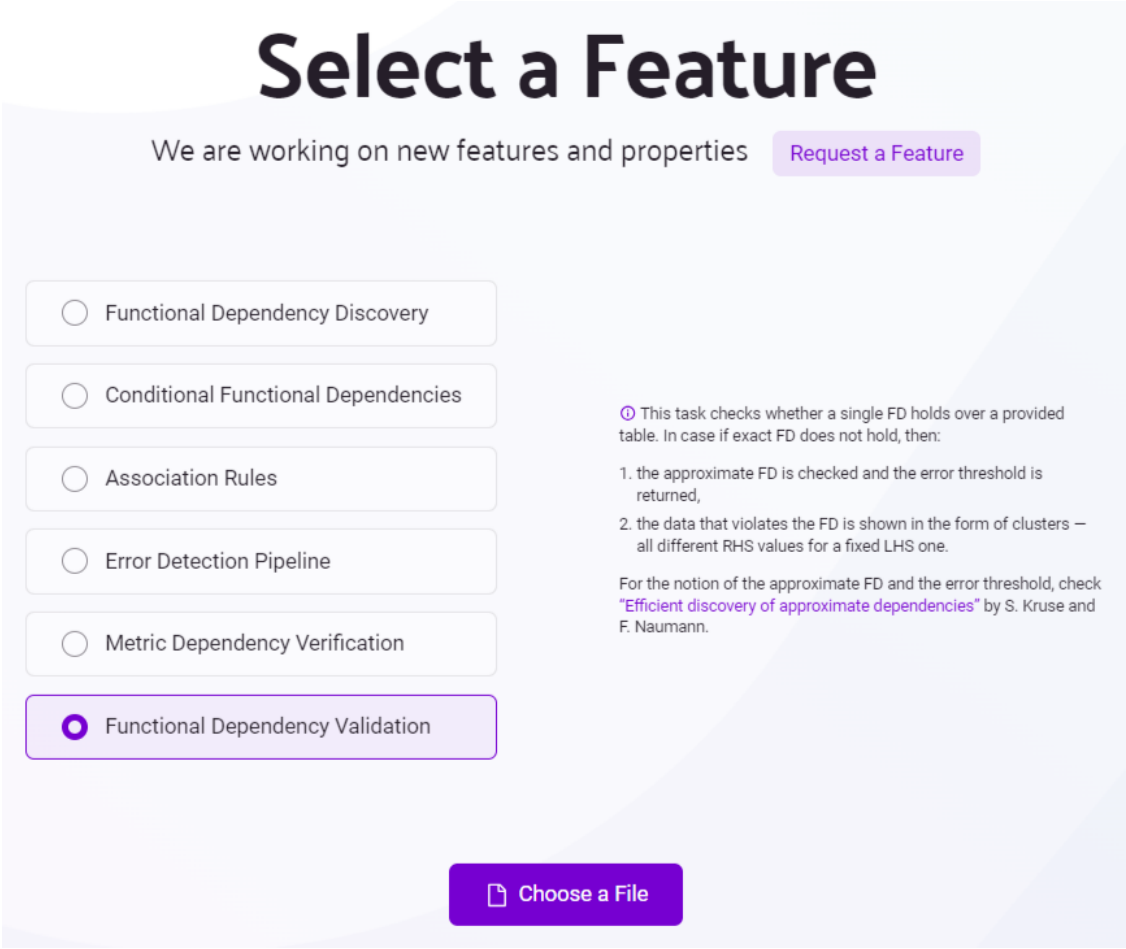

Рис. 1: Выбран примитив AFD

4.2. Выбор файла

Встроенные файлы помечены, какие примитивы можно к ним применить. Поскольку эта информация приходит с сервера, а на нем еще нет информации про новые примитивы, нужно было помечать какой-нибудь датасет доступным для добавляемых примитивов, чтобы перей-

ти к следующему шагу.

4.3. Выбор алгоритма и его конфигураций

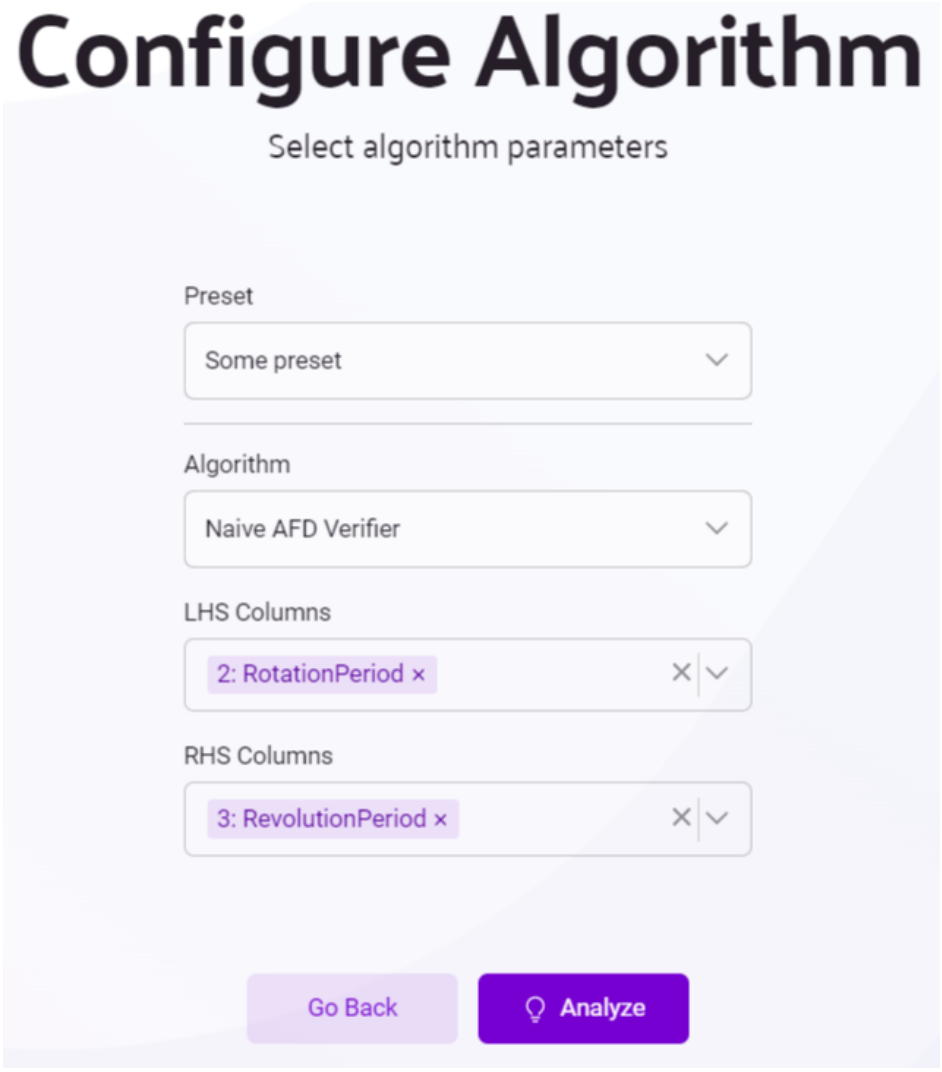

Рис. 2: Конфигурации для AFD

На данный момент оба примитива реализуют по одному алгоритму. Все поля для ввода уже написаны ранее: выпадающий список, в том числе с множественным выбором, ввод числа, выбор числа из определенного диапазона, а также возможность сделать персонализированный компонент. Эта опция понадобилась при реализации страницы конфигураций для Algebraic Constraints, а именно возможность по нажатию на соответствующую кнопку генерировать случайный seed – целое число от 1 до 999999 включительно.

Также были внесены изменения в стилизацию страницы конфигураций. Ранее все поля для ввода располагались в один столбец. И если параметров немного, не более 4, то это смотрелось уместно, однако при увеличении количества параметров (например, у Metric Dependency Verification их 8) все поля для ввода не могли поместиться на экране без прокрутки, что не очень удобно. Было решено преобразовать расположение окошек следующим образом: если у примитива не более 4 параметров, то их вывод остаётся прежним – в один столбец, иначе – разбивается на два. Однако, если размер экрана менее 960 пикселей, то в любом случае будет лишь один столбец. Преобразования были выполнены исключительно с помощью изменения классов стилей.

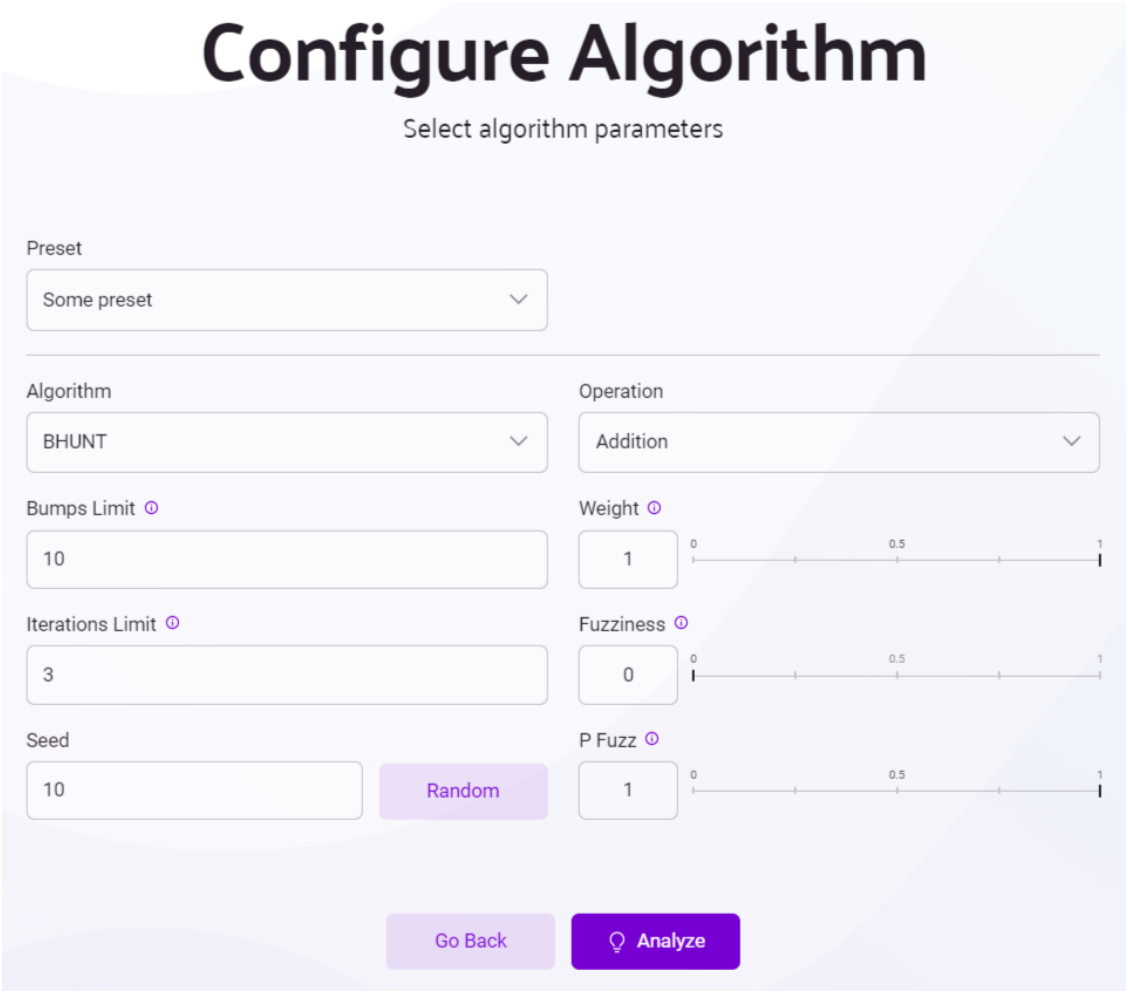

Рис. 3: Конфигурации для AC

4.4. Просмотр результата работы алгоритма

Именно на данном шаге пользователь получает метаданные. Поэтому очень важно, чтобы вывод результата был информативным и интуитивно понятным.

Поскольку в Desbordante уже есть несколько примитивов, некоторые варианты визуализации результатов уже реализованы. Так, например, для AFD не нужно было придумывать что-то кардинально новое, а переиспользовать и внести небольшие изменения в код для MFD[3].

Однако Algebraic Constraints требовал новых идей. Так Instance list имеет абсолютно другой вид, нежели у других примитивов. Также необходимо было добавить вкладку с гистограммами, которые не использовались ранее в проекте.

Результат AC

Результатом работы алгоритма будут все пары атрибутов, для которых он применялся, интервалы, в которых лежат значения указанной арифметической операции, а также индексы строк, которые являются исключениями (они не входят ни в один из результирующих интервалов).

Полученный результат можно сортировать по количеству интервалов и исключений (выбрать можно в модальном окне Ordering).

Для большей наглядности было решено создавать гистограммы по данным, полученным после работы алгоритма.

На скриншоте представлен пример гистограммы. Более тёмные столбики – интервалы (Intervals), светлые – исключения (Outliers). При наведении курсора на столбик-интервал, он окрашивается в фиолетовый, а также все столбики, входящие в ассоциированный интервал (между столбиками-исключениями), окрашиваются в сиреневый. На всплывающей подсказке показывается информация о данном интервале, ассоциированным интервале и количество значений в данном интервале. При наведении на столбик-исключение, он станет чуть темнее, а в всплывающей подсказке выведется сам интервал и количество значений, в него

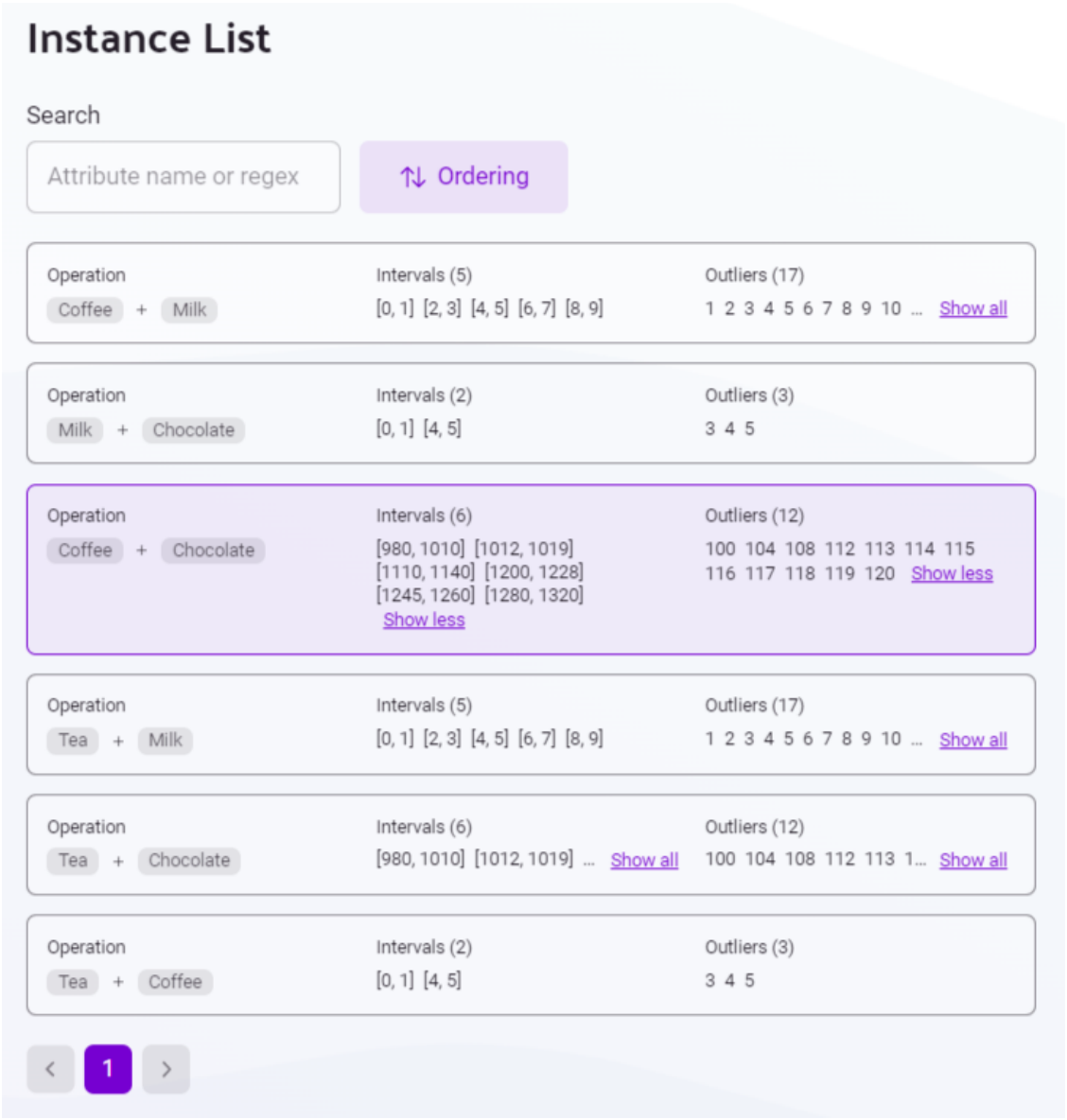

Рис. 4: Instance list

попавших.

Отдельно стоит отметить грануляцию – максимальное количество столбиков. По умолчанию она равняется максимальному количеству ”кочек” (bumps limit) из входных данных алгоритма, но если это значение слишком велико, то гистограмма может быть менее наглядной и понятной, поэтому пользователю предлагается самому указывать желаемое значение.

Рис. 5: Гистограмма с наведением на столбик-интервал

Результат AFD

В результате выводится погрешность (от 0 до 1), количество строк и все кластеры, где нарушается функциональная зависимость. Также пишется для каждого кластера его размер (количество строк в нем), число различных значений в правой части и процент самого частого значения справа.

Оттенками зелёного цвета и галочкой отмечены строки, в которых правая часть является самым частым значением. Оттенками красного цвета и крестиком отмечены все остальные строки.

В модальном окне Visibility можно выбрать сортировку строк по индексам строк и значениям правой части, а также отображать в таблице только столбцы, выбранные в конфигурациях алгоритма (LHS и RHS). Ordering же отвечает за сортировку кластеров по размеру, значениям левой части, количеству различных значений и проценту самого частого в правой части.

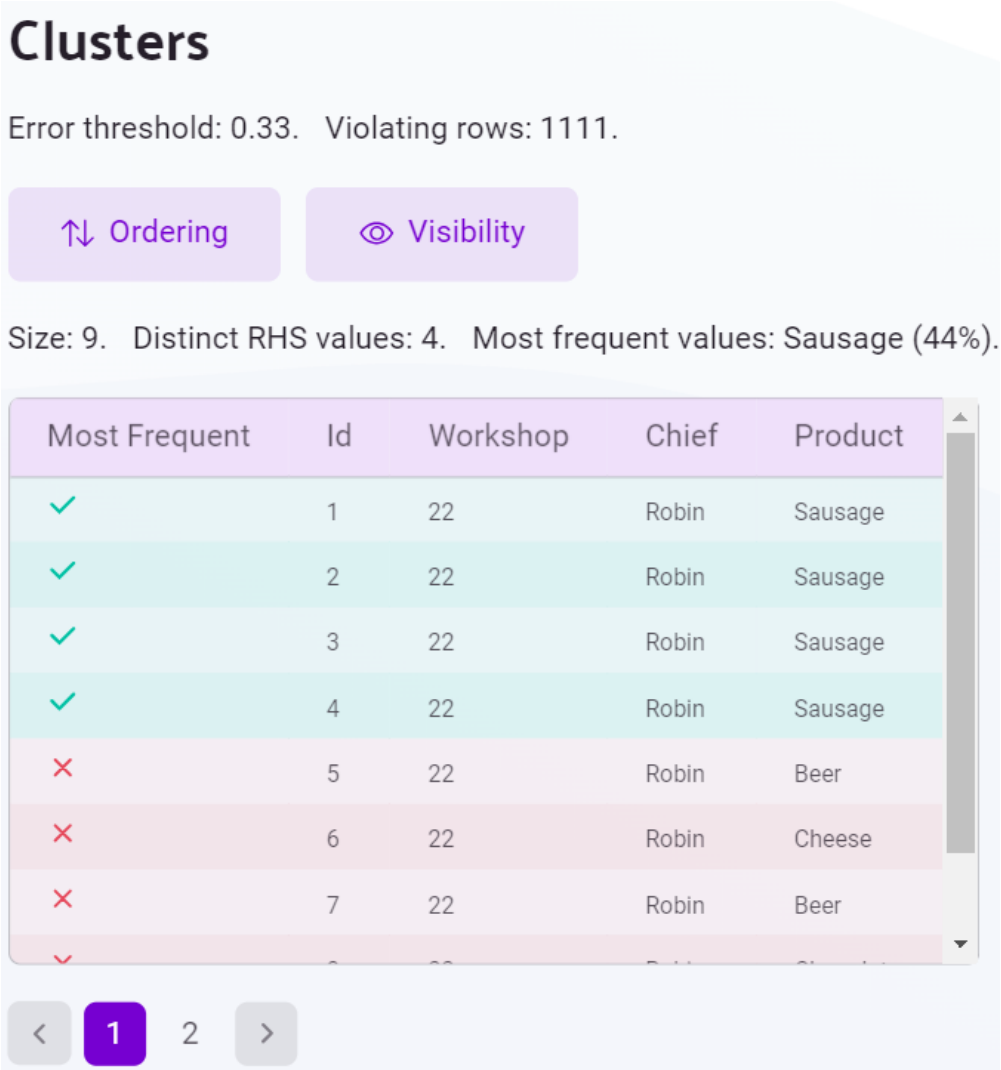

Рис. 6: Результат примитива AFD

На скриншоте предполагается, что левая часть – Workshop и Chief, а правая – Product.

Заключение

В результате работы написана фронтенд-часть для добавления примитивов Algebraic Constraints и Approximate Functional Dependencies в веб-версию Desbordante с учётом отсутствия бэкенда.

Были выполнены следующие задачи:

- Ознакомиться с алгоритмами, реализующими примитивы AC и AFD;
- Создать макет веб-страниц в Figma;
- Сверстать веб-страницы.

После появления бэкенда планируется завершить добавление новых примитивов в веб-версию Desbordante.

Код работы доступен на GitHub [1][2]

[1] AC: https://github.com/vs9h/Desbordante/pull/110

[2] AFD: https://github.com/vs9h/Desbordante/pull/111

Список литературы

[1] Brown Paul G., Haas Peter J. “BHUNT: Automatic Discovery of Fuzzy Algebraic Constraints in Relational Data”. — URL: https://vldb.org/conf/2003/papers/S20P03.pdf (дата обращения: 2023-01-06).

[2] Kruse S., Naumann F. “Efficient discovery of approximate dependencies”. — URL: https://www.vldb.org/pvldb/vol11/p759-kruse.pdf (дата обращения: 2023-01-06).

[3] Белоконный Сергей Александрович. Вывод метрических функциональных зависимостей в веб-интерфейсе Desbordante. — URL: https://github.com/Mstrutov/Desbordante/blob/main/docs/papers/FrontendMFD-SergeyBelokonniy-2022autumn.pdf (дата обращения: 2023-01-06).

[4] Веб-приложение Desbordante. — URL: https://desbordante.unidata-platform.ru/ (дата обращения: 2023-01-06).

[5] Официальный сайт Recharts. — URL: https://recharts.org/en-US/ (дата обращения: 2023-01-06).

[6] Щека Дмитрий Вадимович. Реализации алгоритма BHUNT. — URL: https://github.com/Mstrutov/Desbordante/blob/main/docs/papers/AlgebraicConstraints-DmitriyShcheka-2022spring.pdf (дата обращения: 2023-01-06).