Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего профессионального образования

«Московский физико-технический институт

(государственный университет)»

Факультет управления и прикладной математики

Кафедра теоретической и прикладной информатики

ОРГАНИЗАЦИЯ РАСПРЕДЕЛЕННОГО ХРАНИЛИЩА, ОПТИМИЗИРОВАННОГО ПОД СТАТИСТИЧЕСКИЙ АНАЛИЗ

Выпускная квалификационная работа

(магистерская диссертация)

Направление подготовки: 03.04.01 Прикладная математика и физика

Выполнила:

студентка 973 группы _____ Алексеева Мария Андреевна

Научный руководитель:

д.ф.-м.н., профессор _____ Тормасов Александр Геннадьевич

Научный консультант: _____ Непорада Андрей Леонидович

Москва 2015

Оглавление

Введение 4
1. Постановка задачи 5
1.1. Распределенные хранилища 5
1.1.1. Традиционное хранилище данных 5
1.1.2. Big data 9
1.1.3. NoSQL 10
1.1.4. Map-Reduce (Hadoop) 11
1.2. Статистический анализ 13
1.2.1. Основные методы статистического анализа 13
1.2.2. Репрезентативная выборка и статистический анализ 16
1.3. Статистический анализ в распределенном хранилище и постановка задачи 17
2. Математическая модель 19
2.1. Формализация задачи 19
2.2. Критерии однородности 19
3. Создание модели хранилища 23
3.1. Модель распределенного хранилища 23
3.2. Модуль распределения данных 24
3.3. Модуль контроля репрезентативности данных 25
3.4. Алгоритм случайного распределения данных 25
3.5. Собственный алгоритм распределения данных 26
3.6. Вычисление функции распределения хи-квадрат 28
3.7. Проблема заполнения пустого хранилища 30
3.8. Оптимизация алгоритма распределения данных 30
4. Результаты проведенных тестов 33
4.1. Тесты для случайного распределения 33
4.2. Тесты для собственного алгоритма распределения 39
5. Расширение задачи 46
5.1. Репрезентативность по двум и более распределениям на одном узле 46

5.2. Репрезентативность двух и более измерений на разных узлах 48
Заключение 49
Список литературы 50

Введение

Данная работа посвящена вопросу организации распределенного хранилища, оптимизированного под статистический анализ. В последнее время широко востребована тема хранения больших объемов данных, предназначенных не для быстрого доступа, а для анализа в практике принятия бизнес-решений. Именно для этой задачи служат хранилища (data warehouses). Когда количество данных становится особенно большим, данные все сложнее хранить в одном месте, а операции над ними занимают значительно большее время. Тогда выбор делается в сторону распределенного хранилища, которое позволяет не только оптимизировать хранение большого объема данных, но и улучшить скорость анализа данных за счет распараллеливания работы над ними.

Вид анализа, производящегося над данными, зависит от цели и задач принятия решений. В данной работе рассмотрен статистический анализ. Этот вид анализа обладает одним важным свойством: результат анализа, произведенного над репрезентативной выборкой из всех данных, будет совпадать с результатом этого же анализа над всеми данными. Именно это свойство лежит в основе подхода к организации нашего хранилища, оптимизированного под статистический анализ.

1. Постановка задачи

В этой главе ведется обсуждение постановки самой задачи, рассматриваются проблемы и вопросы, связанные с возможным решением, а также приводится краткая теория, касающаяся распределенных хранилищ данных и статистического анализа.

1.1. Распределенные хранилища

Прежде чем говорить о распределенных хранилищах данных, следует рассмотреть классическое хранилище, как общий случай хранения данных.

1.1.1. Традиционное хранилище данных

Традиционное хранилище данных — это предметно-ориентированная база данных, предназначенная для подготовки отчетов и бизнес-анализа [1]. Обычное хранилище строится на базе систем управления базами данных и систем поддержки принятия решений. Данные, поступающие в хранилище, доступны только для чтения и анализа. Выделяется 4 основных принципа организации хранилища [1]:

1. Проблемно-предметная ориентация: данные хранятся в соответствии с областями, которые они описывают;
2. Интегрированность: данные удовлетворяют всем требованиям предприятия в целом;
3. Некорректируемость: данные в хранилище не корректируются и не удаляются;
4. Зависимость от времени: данные корректны только в том случае, когда они привязаны к определенному промежутку и моменту времени.

Схема организации традиционного хранилища представлена на рис. 1 [2].

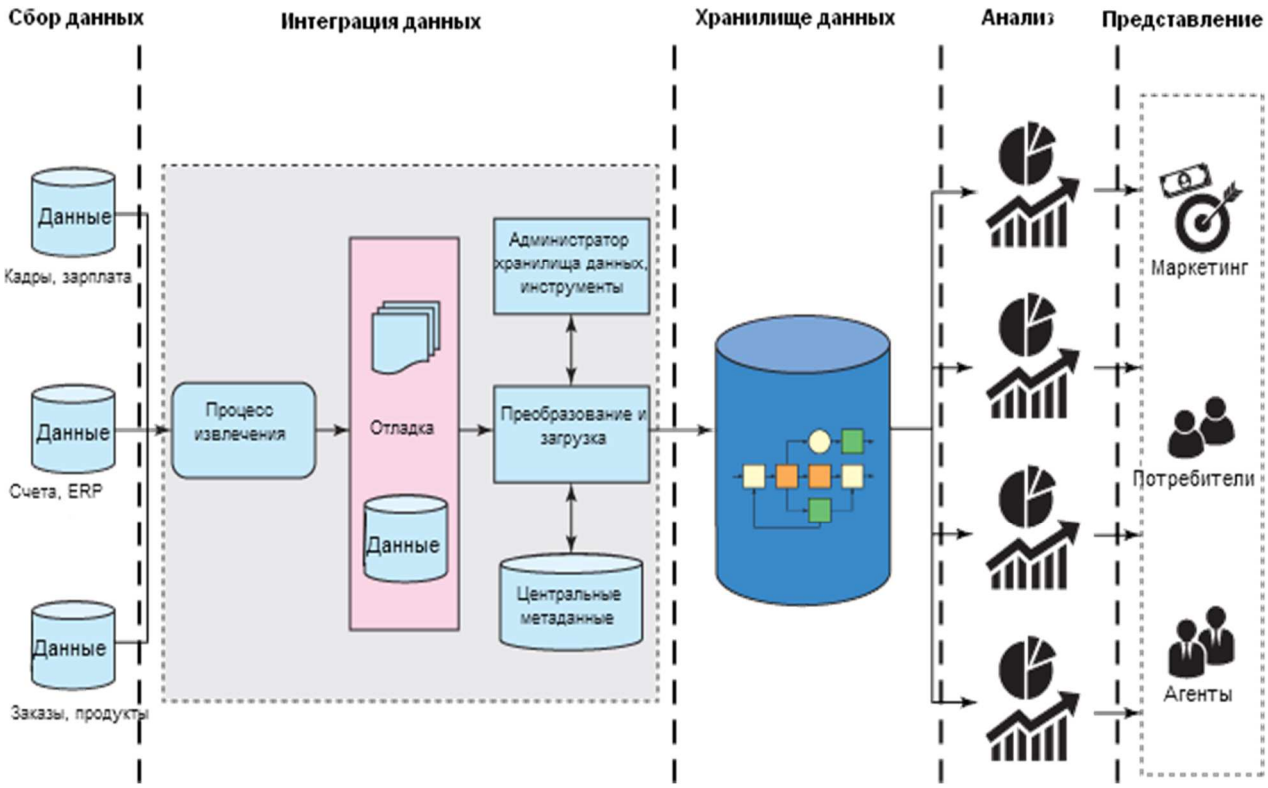

Рисунок 1. *Стандартная архитектура традиционного хранилища данных*

В организации хранилища выделяется пять уровней [2]:

- **Уровень сбора данных** состоит из модулей для получения данных от систем-источников, но не занимается обработкой и распределением;
- **Уровень интеграции** данных интегрирует данные для передачи на уровень хранения данных от источника (от уровня сбора данных);
- **Уровень хранилища** данных получает данные от уровня интеграции и хранит их согласно выбранной модели;
- **Аналитический уровень** представляет данные в формате куба для упрощения анализа пользователем;

- **Уровень представления** – это приложения или порталы, отображающие данные в читабельном и удобном для анализа виде для пользователя посредством, например, веб-страниц.

Существуют два архитектурных направления в организации хранилища данных — нормализованные хранилища данных и хранилища с измерениями. В нормализованных хранилищах данные находятся в предметно-ориентированных таблицах третьей нормальной формы. В таких хранилищах есть определенное достоинство — простота создания и управления, однако их недостатком является большое количество таблиц (как следствие нормализации).

Хранилища с измерениями используют схему «звезда» (рис. 2) или схему «снежинка» (рис. 3). При этом в центре «звезды» находятся данные, а измерения образуют лучи. Различные таблицы фактов совместно используют таблицы измерений, что значительно облегчает операции объединения данных из нескольких предметных таблиц фактов. Достоинства данной архитектуры — это простота и понятность для разработчиков, пользователей и скорость доступа к данным при сложном анализе. Недостатками же являются более сложные процедуры подготовки и загрузки данных, а также управление и изменение измерений данных.

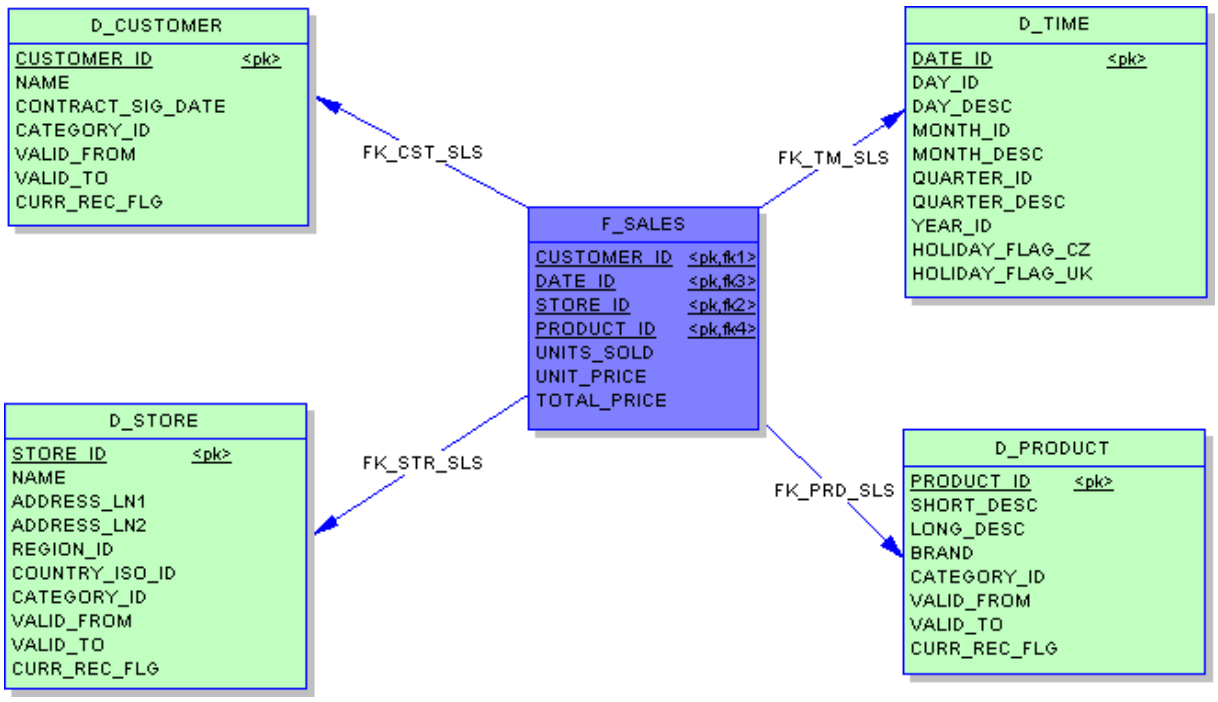

Рисунок 2. *Схема организации хранилища "звезда"*

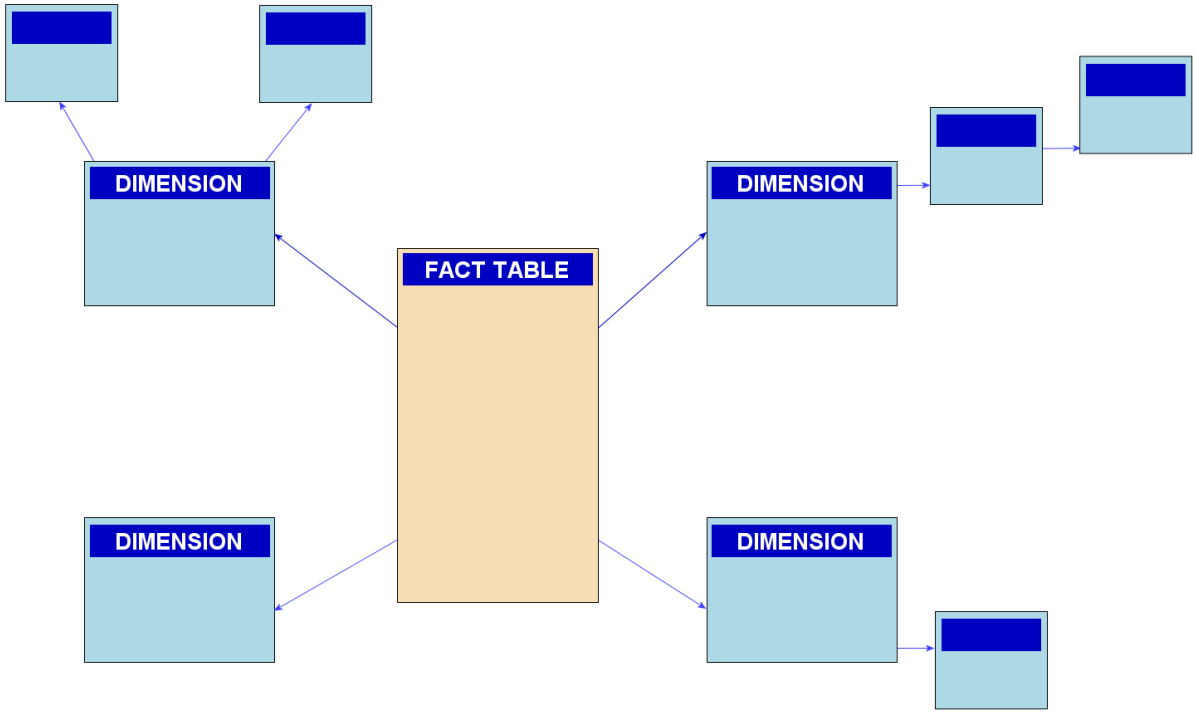

Рисунок 3. *Схема "снежинка"*

Основными источниками данных для хранилища являются традиционные системы регистрации операций, отдельные документы, наборы данных. Так же стоит отметить, что если хранилище используется в исследователь-

ской работе, а не только в бизнесе, источниками могут быть записи показаний некоторых приборов или записи об исследовании некоторых объектов. Таким образом, вид данных, поступающих в хранилище для анализа и хранения, определяется областью их применения.

В хранилище над данными возможны следующие операции:

- Извлечение — перемещение информации от источников данных в отдельную БД, приведение их к единому формату;
- Преобразование — подготовка информации к хранению в оптимальной форме для реализации запроса, необходимого для принятия решений;
- Загрузка — помещение данных в хранилище, производиться атомарно, путем добавления новых фактов или корректировкой существующих;
- Анализ: OLAP, Data Mining, сводные отчеты;
- Представление результатов анализа.

Однако в последнее время в области обработки данных происходят значительные изменения, приводящие к новым требованиям к организации хранилищ: объем данных, как и частота их поступления, растут; повышается спрос на ресурсоемкие вычисления. Так как основной проблемой этих требований является большой объем данных, то область технологий, работающая с этими требованиями, называется «большие данные» (big data).

1.1.2. Big data

Одной из технологий, работающей в рамках задачи «big data», является распределенное хранилище. Такие хранилища данных представляют собой архитектуру клиент-сервер, где клиент – это уровень интеграции и представления, а сервер – набор узлов, осуществляющих хранение данных

и выполняющих вычисления над этими данными. Такая организация хранилища позволяет хранить как можно большие объемы данных и легко масштабироваться в случае переполнения узлов путем добавления нового узла, а также производить параллельный анализ над данными на каждом узле, предоставляя ресурсоемкие и точные вычисления.

Граница области применения традиционного и распределенного хранилищ сильно размыта. Но рекомендуется переходить к решению задачи больших данных, когда существенны [2]:

- необходимость анализировать данные из новых источников;
- повышение сложности данных (разнообразие типов данных; объемы данных; скорость создания данных; достоверность данных, поступающих из нескольких источников);
- усложнение задач анализа;
- растущая доступность экономически эффективных вычислительных; средств и систем хранения данных.

В решении задачи хранения и анализа больших данных можно выделить два основных подхода: NoSQL и Map-Reduce (Hadoop).

1.1.3. NoSQL

Рассматривать NoSQL подход необходимо в сравнении с подходом использования реляционной базы данных. Известно [4], что стандартная реляционная база данных не обладает существенным для решения вопроса big data свойством – масштабируемостью. NoSQL выгодно отличается на фоне RDBMS именно благодаря своей высокой производительности и гибкой обработке больших масс данных. Это достигается за счет отказа от высокой структурированности, присущей реляционным базам данных.

Хранилища данных, построенных на системе NoSQL, подразделяют на следующие типы [5]:

- **Хранилище «ключ – значение»** используется для хранения изображений, создания специализированных файловых систем, в качестве кэшей для объектов и в системах, спроектированных с прицелом на масштабируемость.
- **Хранилище семейств колонок (или Bigtable-подобные базы данных),** где данные хранятся в виде разреженной матрицы, строки и столбцы которой используются как ключи.
- **Документно-ориентированные СУБД** служат для хранения иерархических структур данных, используются в системах управления содержимым, документальном поиске и т.п.
- **Базы данных на основе графов** применяются для задач, в которых данные имеют большое количество связей: социальные сети и прочее.

Помимо NoSQL существует и другой подход – MapReduce парадигма, о которой будет рассказано в следующем пункте.

1.1.4. Map-Reduce (Hadoop)

Вычислительная парадигма Map-Reduce, реализованная в системе Hadoop, не имеет ничего общего с СУБД и NoSQL в вопросе организации.

Сам по себе Hadoop – это набор утилит, библиотек и фреймворк для разработки и выполнения распределенных программ, работающих на кластерах из сотен и тысяч узлов. Состоит из четырех модулей: Hadoop Common (связующее ПО), HDFS (распределенная файловая система), YARN (система для планирования заданий и управления кластером) и Hadoop Map-Reduce

(платформа для программирования и выполнения распределенных Map-Reduce вычислений).

С помощью Hadoop организовывается распределенное хранилище. За хранение и доступ к данным отвечает HDFS, за анализ над данными Map-Reduce. Схема выполнения вычислений над данными для Hadoop приведена на рисунке 4.

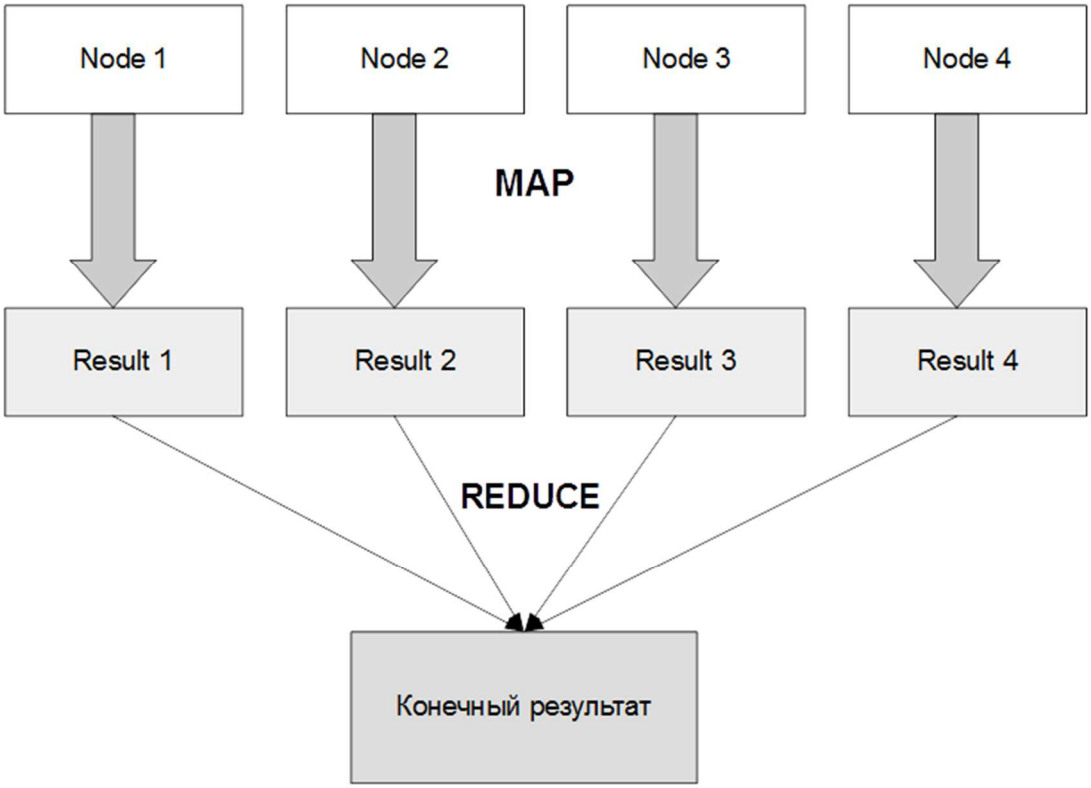

Рисунок 4. Схема парадигмы Map-Reduce

Вычисления над данными разбиваются на два этапа.

- **Первый этап (Map):** Базовый обработчик, реализованный разработчиком, преобразует исходные пары «ключ-значение» в промежуточный набор пар «ключ-значение» на каждом узле.
- **Второй этап (Reduce):** Другой обработчик сводит промежуточный набор пар в окончательный.

Рассмотрим пример: нам необходимо среди всех числовых данных на узлах посчитать среднее арифметическое. В этом случае базовый обработчик подсчитывает среднее на каждом узле, а второй обработчик получает результат среднего за счет вычисления на результатах работы первого обработчика.

Отметим также, что Hadoop обладает свойством масштабируемости и способностью быстро и параллельно выполнять вычисления над всем массивом данных.

1.2. Статистический анализ

Среди большого разнообразия видов и способов добывания знаний из данных (data mining) таких, как методы классификации, моделирования, прогнозирования, выделяются статистические методы анализа данных.

Отличительной чертой статистического анализа является то, что он всегда производится над некоторой выборкой данных. Предполагается, что данные – это наблюдения некоторых случайных величин или некоторые переменные со статистической ошибкой. Это обстоятельство и позволяет рассматривать статистический анализ отдельно от всего data mining в рамках обсуждения организации хранилища.

1.2.1. Основные методы статистического анализа

Существует множество методов статистического анализа, как одномерного, так и многомерного. Рассмотрим некоторые из них.

- ***Дескриптивный (описательный) анализ***

  Данный вид анализа производится путем составления частотных таблиц, вычисление статистических характеристик (математическое ожидание, дисперсия и другие) или графическое представление. Описательный анализ производится на одном измерении и «описывает» свойства

функции распределения случайной величины, над выборкой которой производится анализ.

- ***Корреляционный анализ***

Корреляционный анализ используется при анализе тесноты связи между двумя и более переменными. Тесно связан с регрессионным анализом. Существуют следующие ограничения на проведение корреляционного анализа:

- Применение возможно при наличии достаточного количества наблюдений для изучения (должно в 5-6 раз превышать число факторов)
- Необходимо, чтобы совокупность значений всех факторных и результативного признаков подчинялась многомерному нормальному распределению [9]
- Исходная совокупность значений должна быть качественно однородной [10]

Следует помнить, что сам по себе факт корреляционной зависимости не означает, что одна из переменных предшествует или является причиной изменений, или то, что переменные вообще причинно связаны между собой, а не наблюдается действие третьего фактора [10].

- ***Регрессионный анализ***

Регрессионный анализ – статистический метод исследования влияния одной или нескольких независимых переменных на зависимую. Целью регрессионного анализа является установление не самого факта зависимости, а его характера, так как для применения анализа уже необходим факт наличия зависимости.

- ***Факторный анализ***

Факторный анализ – это своего рода обобщение регрессионного анализа, когда изучается взаимосвязь между значениями переменных. Предполагается, что известные переменные зависят от меньшего количества неизвестных переменных и случайно ошибки. Для выполнения факторного анализа так же необходима однородность выборки.

- ***Дисперсионный анализ***

Дисперсионный анализ используется для поиска зависимостей в экспериментальных данных путем исследования значимости различий в средних значениях. Позволяет сравнивать средние значения трех и более групп.

- ***Компонентный анализ***

Компонентный анализ - многомерный статистический метод снижения размерности, применяемый для изучения взаимосвязей между значениями количественных переменных. Задача компонентного анализа состоит в преобразовании исходной системы взаимосвязанных переменных в новую систему некоррелированных обобщенных показателей или ортогональных показателей [11].

- **Дискриминантный анализ**

Дискриминантный анализ – это целый набор методов статистического анализа для решения задач распознавания образов. Используется для принятия решения о том, какие переменные разделяют возникающие наборы данных. В отличие от кластерного анализа, здесь группы известны априори.

- ***Временной ряд***

  Вообще временной ряд – это собранный в разные моменты времени статистический материал о значении каких-либо параметров (в простейшем случае одного) исследуемого процесса. Каждая единица статистического материала называется измерением или отсчётом, также допустимо называть его уровнем на указанный с ним момент времени. Во временном ряде для каждого отсчёта должно быть указано время измерения или номер измерения по порядку. Временной ряд существенно отличается от простой выборки данных, так как при анализе учитывается взаимосвязь измерений со временем, а не только статистическое разнообразие и статистические характеристики выборки [12].

Все виды статистического анализа объединяет одно важное свойство, которое будет описано в следующем параграфе.

1.2.2. Репрезентативная выборка и статистический анализ

Когда речь идет о предмете математической статистики и, особенно, о статистическом анализе, важно правильно понимать основной предмет изучения – данные. В нашем случае важно точно определить, что подразумевается под словами «генеральная совокупность», «выборка» и «подвыборка».

Генеральной совокупностью называется множество всех возможных значений или реализаций исследуемой случайной величины. *Выборка* - это часть генеральной совокупности. *Репрезентативная выборка* – это выборка из генеральной совокупности такая, что её все статистические свойства и признаки совпадают со статистическими свойствами и признаками генеральной совокупности.

Если вести речь о случайной величине, то репрезентативная выборка – это набор данных, подчиняющийся тому же закону распределения, что и случайная величина.

Подвыборкой называется часть выборки случайной величины. Подвыбока будет репрезентативной, когда её распределение совпадает с распределением выборки.

Важнейшим свойством статистического анализа является то, что он проводится априори над некоторой выборкой, описывающей наблюдение случайной величины. Значит, можно взять репрезентативную подвыборку этой выборки и, проведя анализ над ней, получить аналогичные анализу над общей выборкой результаты. Это свойство и будет использоваться при организации распределенного хранилища.

1.3. Статистический анализ в распределенном хранилище и постановка задачи

Вопрос статистического анализа в распределенном хранилище решается двумя путями: либо производится анализ всех данных в хранилище (если метод анализа можно представить виде Map-Reduce парадигмы), либо из хранилища выделяется репрезентативная выборка, над которой и производятся все расчеты. Если говорить о решении для распределенного хранилища Hadoop, то проводить статистический анализ можно с помощью инструмента, основанного на языке анализа данных R, - RHIPE [16].

Однако в данной работе предлагается применить иной подход к организации хранилища: формировать репрезентативную выборку в момент распределения данных, а не перед началом анализа. Поэтому задача ставится следующим образом: организовать распределение данных в хранилище таким образом, чтобы на выбранном узле сформировать репрезентатив-

ную выборку для выбранного измерения. В процессе формирования выборки необходимо сохранить равномерность распределения данных по узлам хранилища.

Таким образом, в организации распределенного хранилища, оптимизированного под статистический анализ, будет уделяться главное внимание вопросу распределения данных, нежели хранения, связи между узлами, отказоустойчивости и других моментов, которые в данном случае не играют большой роли в вопросе статистического анализа. Поэтому в ходе работы над этой темой была построена система распределения данных в хранилище и модуль для её тестирования и проверки итоговых результатов.

Однако, прежде чем описывать реализованную модель, обратимся к математической модели поставленной задачи и формализуем требования к результатам работы.

2. Математическая модель

2.1. Формализация задачи

Путь у нас есть распределенное хранилище, состоящее из n $(\mathrm{n} > 1)$ узлов. Есть объем N записей (данных) известной природы и вида. Каждая запись имеет хотя бы одно измерение дискретного или непрерывного вида, известного многообразия или интервала $[\min, \max]$ соответственно. Без ограничений общности репрезентативная выборка будет создаваться на первом узле.

Рассмотрим пример. Пусть у нас имеются показания термометра с различных метеостанций за различные даты. Каждая запись задается тремя параметрами: номер станции, дата, среднесуточная температура. Допустим, нас интересует статистический анализ температуры: поэтому мы формируем репрезентативную выборку для этого измерения.

Возникает вопрос, как измерять качество репрезентативности выборки. Для этого воспользуемся критериями однородности, известными из математической статистики.

2.2. Критерии однородности

Путь у нас есть две выборки μ и η размером n_{μ} и n_{η} соответственно для случайных величин, распределение которых нам неизвестно. Чтобы определить, имеют ли обе выборки одно и тоже определение или нет, служат критерии однородности. Один из этих критериев - критерий однородности хи-квадрат – позволяет утверждать с некоторой вероятностью, принадлежат ли выборки одному распределению. Приведем его описание для общего случае нескольких выборок.

Сразу отметим, что доказательство корректности и состоятельности критерия хи-квадрат излагается в [17][18]. Критерий хи-квадрат применяют к анализу данных, имеющих дискретную структуру, конкретнее, когда в опытах наблюдается некоторый переменных признак, принимающий конечное число $\mathrm{K} \geq 2$ различных значений. Но к такой схеме можно свести любую модель, применяя предварительно метод группировки данных [17], поэтому метод хи-квадрат применим на самом деле к анализу данных любой природы, т.е. является в этом смысле универсальным. Кроме того, с помощью этого метода можно анализировать любое конечное число выборок. Итак, пусть осуществлено k серий независимых наблюдений, состоящих из $\mathrm{n}_1, \dots, \mathrm{n}_\mathrm{k}$ наблюдений соответственно, и пусть $\vartheta_\mathrm{i} = (\vartheta_{\mathrm{i}1}, \dots, \vartheta_{\mathrm{i}\mathrm{N}})$ – частоты исходов i-ой серии, а $\mathrm{p}_\mathrm{i} = (\mathrm{p}_{\mathrm{i}1}, \dots, \mathrm{p}_{\mathrm{i}\mathrm{N}})$ – их вероятности (i = 1, … , k). Тогда гипотеза однородности означает утверждение, что вероятности исходов не менялись от серии к серии, т.е.

$$\mathrm{H}_0\colon \mathrm{p}_1 = \cdots = \mathrm{p}_\mathrm{k} = \mathrm{p} = (\mathrm{p}_1, \dots, \mathrm{p}_\mathrm{N}),$$

где p – некоторый (неизвестный) вектор вероятностей. При этом $\mathrm{p}_1 + \cdots + \mathrm{p}_\mathrm{N} = 1$. Критерий проверки этой гипотезы состоит в том, что все данные объединяются в одну выборку объема $\mathrm{n} = \mathrm{n}_1 + \cdots + \mathrm{n}_\mathrm{k}$ с частотами исходов $\vartheta_{*\mathrm{j}}, \dots, \vartheta_{*\mathrm{N}}$. В итоге в [17] получена тестовая статистика критерия однородности хи-квадрат

$$X^2_{n_1 \dots n_k} = \mathrm{n} \sum_{i=1}^{k} \sum_{j=1}^{N} \frac{1}{\mathrm{n}_\mathrm{i} \vartheta_{*\mathrm{j}}} \left(\vartheta_{\mathrm{ij}} - \frac{\mathrm{n}_\mathrm{i} \vartheta_{*\mathrm{j}}}{n} \right)^2$$

с помощью которой и строится критическая область критерия:

$$\left\{ \breve{X}^2_{n_1 \dots n_k} > t_\alpha \right\}.$$

Для нахождения критической границы t_α при заданном уровне значимости [18] α используется следующий предельный результат [19]:

при $\mathrm{n_i} \to \infty$ $(\mathrm{i} = 1, \dots, \mathrm{k})$:

$$\mathrm{L}\left(\check{X}^2_{n_1 \dots n_k} \middle| \mathrm{H}_0\right) \to \chi^2\big((\mathrm{k} - 1)(\mathrm{N} - 1)\big).$$

В силу этого результата при больших n_i можно полагать $t_\alpha = \chi^2_{1-\alpha,(k-1)(N-1)}$, где статистика $\check{X}^2_{n_1 \dots n_k}$ вычисляется по формуле, приведенной выше. Вероятность ошибочно отклонить при этом истинную гипотезу приблизительно равна α, если n достаточно велико.

Чтобы применить данный критерий к нашему случаю, рассмотрим его действие на примере со среднесуточной температурой.

Есть общая выборка из N записей, а есть её подмножество – выборка из N_1 записей. Каждая запись имеет характеристику – среднесуточную температуру на выбранной станции и в выбранную дату. Нам известно, что $T \in [T_{min}, T_{max}]$. Разобьем этот отрезок на D равных отрезков и будем говорить, что температура имеет значение i, если её величина попадает на $i-$й интервал. Таким образом мы превратили непрерывную величину в дискретную. Для каждой записи мы знаем номер интервала, к которому принадлежит температура. Подсчитаем для каждого температурного отрезка количество записей для общей выборки и для выборки на первом узле и приведем их в таблице:

Интервал	1	2	3	.	.	.	D-2	D-1	D
Общая выборка	α_1	α_2	α_3				α_{D-2}	α_{D-1}	α_D
Выборка на первом узле	β_1	β_2	β_3				β_{D-2}	β_{D-1}	β_D

Таблица 1. *Частотная таблица для среднесуточной температуры*

Критерий однородности Хи-квадрат позволяет определить, репрезентативна ли выборка на первом узле по отношению к общей выборке данных.

Посчитаем статистику для нашего случая: k = 2, N = D, если общая выборка размера n_0, а n_1 – размер выборки на первом узле:

$$\check{X}^2_{n_0,n_1} = (n_0 + n_1) \sum_{j=1}^{D} \left(\frac{1}{n_0(\alpha_j + \beta_j)} \left(\alpha_j + \frac{n_0(\alpha_j + \beta_j)}{n_0 + n_1} \right)^2 + \frac{1}{n_1(\alpha_j + \beta_j)} \left(\beta_j + \frac{n_1(\alpha_j + \beta_j)}{n_0 + n_1} \right)^2 \right) .$$

Выполнив необходимые вычисления, мы получим значение статистики для нашего случае. Далее нам необходимо обратиться к таблице квантилей распределения хи-квадрат. Напомним, что квантиль – это число, при котором функция распределения хи-квадрат равна заданной вероятности α. Пример таблицы:

	0,1	**0,2**	**0,3**	**0,4**	**0,5**	**0,6**	**0,7**	**0,8**	**0,9**
1	0,02	0,06	0,15	0,28	0,45	0,71	1,07	1,64	2,71
2	0,21	0,45	0,71	1,02	1,39	1,83	2,41	3,22	4,61
3	0,58	1,01	1,42	1,87	2,37	2,95	3,66	4,64	6,25
4	1,06	1,65	2,19	2,75	3,36	4,04	4,88	5,99	7,78
5	1,61	2,34	3	3,66	4,35	5,13	6,06	7,29	9,24
6	2,2	3,07	3,83	4,57	5,35	6,21	7,23	8,56	10,64
7	2,83	3,82	4,67	5,49	6,35	7,28	8,38	9,8	12,02
8	3,49	4,59	5,53	6,42	7,34	8,35	9,52	11,03	13,36
9	4,17	5,38	6,39	7,36	8,34	9,41	10,66	12,24	14,68
10	4,87	6,18	7,27	8,3	9,34	10,47	11,78	13,44	15,99

Таблица 2. *Таблица квантилей хи-квадрат.*

В нашем случае число степеней свободы есть $(2-1)(D-1) = D-1$. Значит, для строки с индексом, равным D-1, мы ищем ближайшее по значению значение квантиля и определяем α. Полученное значение α и будет обозначать вероятность того, что выборка на первом узле репрезентативно представляет общую выборку данных, то есть имеет то же распределение.

На основе формализованной формулировки задачи можно создавать саму модель хранилища, уделяя пристальное внимание модулю распределения данных и контролю качества выборки. Об этом будет идти речь в следующей главе.

3. Создание модели хранилища

В этой главе будет описан процесс создания модели хранилища, приведен алгоритм модуля распределения данных в двух вариантах, а также показано, как решалась проблема вычисления функции распределения хи-квадрат.

3.1. Модель распределенного хранилища

Организация распределенного хранилища будет построена на архитектуре «клиент-сервер», где в роли клиента выступает модули принятия, распределения данных и организации анализа на узлах, а в роли серверов выступают узлы хранилища, на которых хранятся данные и производится анализ, инициируемый клиентом. Пользователь осуществляет работу с клиентом, для простоты предположим, что клиент один.

Изобразим схематически структуру нашего распределенного хранилища, уделяя особое внимание стороне клиента.

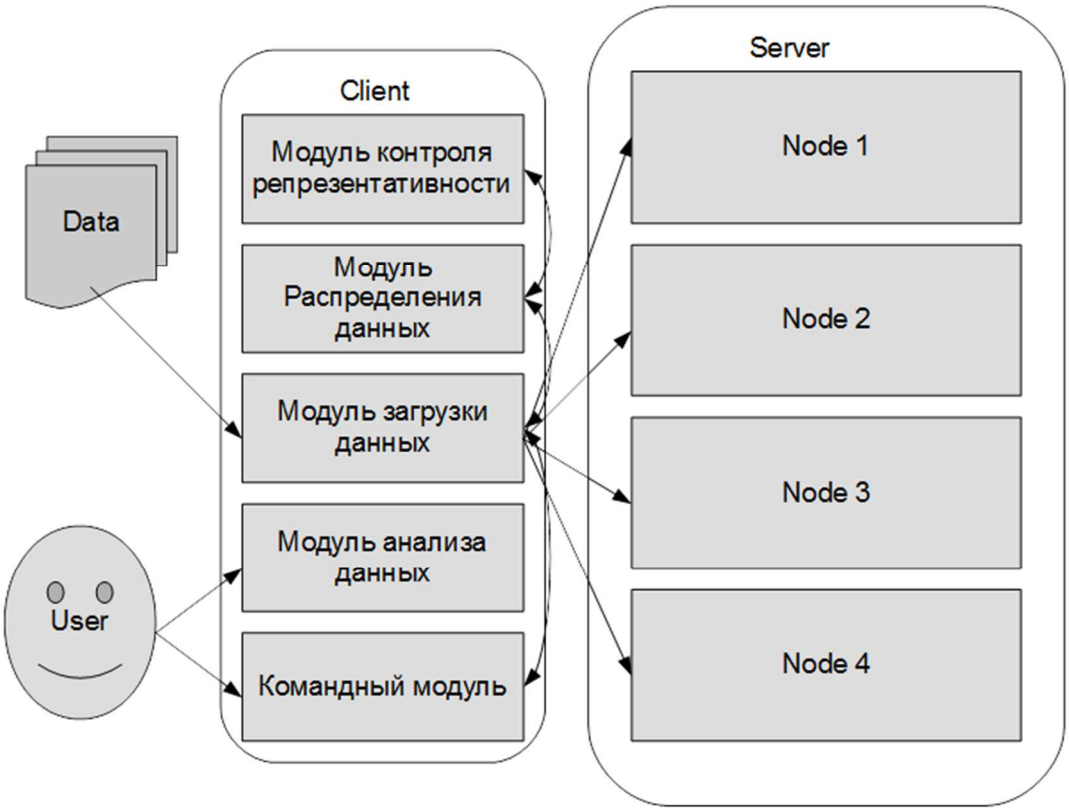

Рисунок 3. *Схема организации распределенного хранилища*

Особое внимание следует уделить модулю распределения и контроля репрезентативности данных. О них и пойдет речь далее.

3.2. Модуль распределения данных

Основной модуль хранилища, интересующий нас в данном исследовании, это модуль распределения данных. На вход модуль получает одну запись, на выходе предоставляет номер узла, куда необходимо эту запись положить. Напомним, что нас интересует сохранение репрезентативности выборки на первом узле и равномерное распределение всех данных в хранилище одновременно.

При инициализации модуля ему необходимо передавать число узлов в хранилище, репрезентативность какого измерения нас интересует и минимальную вероятность репрезентативности выборки на первом узле.

3.3. Модуль контроля репрезентативности данных

Данный модуль необходим для тестирования модуля распределения. Для каждой записи, поступающей в хранилище, модуль записывает значение её главного измерения и узел, на который эта запись была распределена.

При запросе модуль вычисляет вероятность репрезентативности выборки на заданном узле по сравнению со всем массивом данных: вычисляется статистика и по квантилям определяется полученный уровень вероятности.

Про подсчет квантилей распределения хи-квадрат будет рассказано в пункте о вычислении функции распределения хи-квадрат.

3.4. Алгоритм случайного распределения данных

Смысл алгоритма случайного распределения данных прост – мы пробуем сформировать репрезентативную выборку по определению – случайным образом выбирая данные из общей выборки. Для большого количества данных ожидается хороший показатель репрезентативности, однако при небольшом количестве данных или при большом количестве узлов ожидается непредсказуемый результат.

При инициализации модуль получает количество узлов в хранилище. При поступлении одной записи с помощью стандартных функций рандомизации языка Python случайно выбирается номер узла, на который следует положить эту запись. Результаты тестирования этого алгоритма приведены в следующей главе, а пока изложим принцип организации собственного алгоритма распределения.

3.5. Собственный алгоритм распределения данных

Чтобы поддерживать репрезентативность выборки на первом узле, будем использовать критерий хи-квадрат (описан в предыдущей главе) как числовой критерий для принятия решений и контроля репрезентативности. Отметим, что в отличие от случайного распределения для этого алгоритма уровень минимальной вероятности репрезентативности α должен быть задан заранее, до распределения данных на хранилище. На каждом шаге заполнения хранилища, получая каждую запись отдельно, необходимо сделать следующий выбор: положить ли данные на первый узел или на остальные. В случае второго решения данные распределяются случайным образом среди оставшихся узлов для обеспечения равномерности нагрузки на узлы. Предлагается подсчитывать статистическую сумму для первого и второго случаев, и на основе этих данных принять необходимое решение. Допустим, мы получили значение статистической суммы для первого узла Σ_1, для всех данных Σ и значение функции распределения хи-квадрат для данного количества интервалов измерения и вероятности репрезентативности $\alpha - \mathrm{X}^2_{n,1-\alpha}$. Так же нам известно количество данных, уже хранящихся на всех узлах $N_1, .., N_n$и общее количество данных на хранилище $N_0 = N_1 + \cdots + N_n$.

Принятие решение о номере узла проходит в несколько этапов:

- Определяем, сохраняется ли репрезентативность выборки при добавлении записи на первый узел.
- Если сохраняется, проверяем, сохранится ли выборка, если данные не класть на первый узел. Если сохраняется, распределяем данные согласно вопросу равномерной загруженности хранилища, иначе выбираем первый узел.

- Если не сохраняется, проверяем сохранение репрезентативности в случае выбора не первого узла. Если репрезентативность остается, выбираем этот вариант. Если нет, то выбираем тот вариант, который обеспечивает большую репрезентативность.

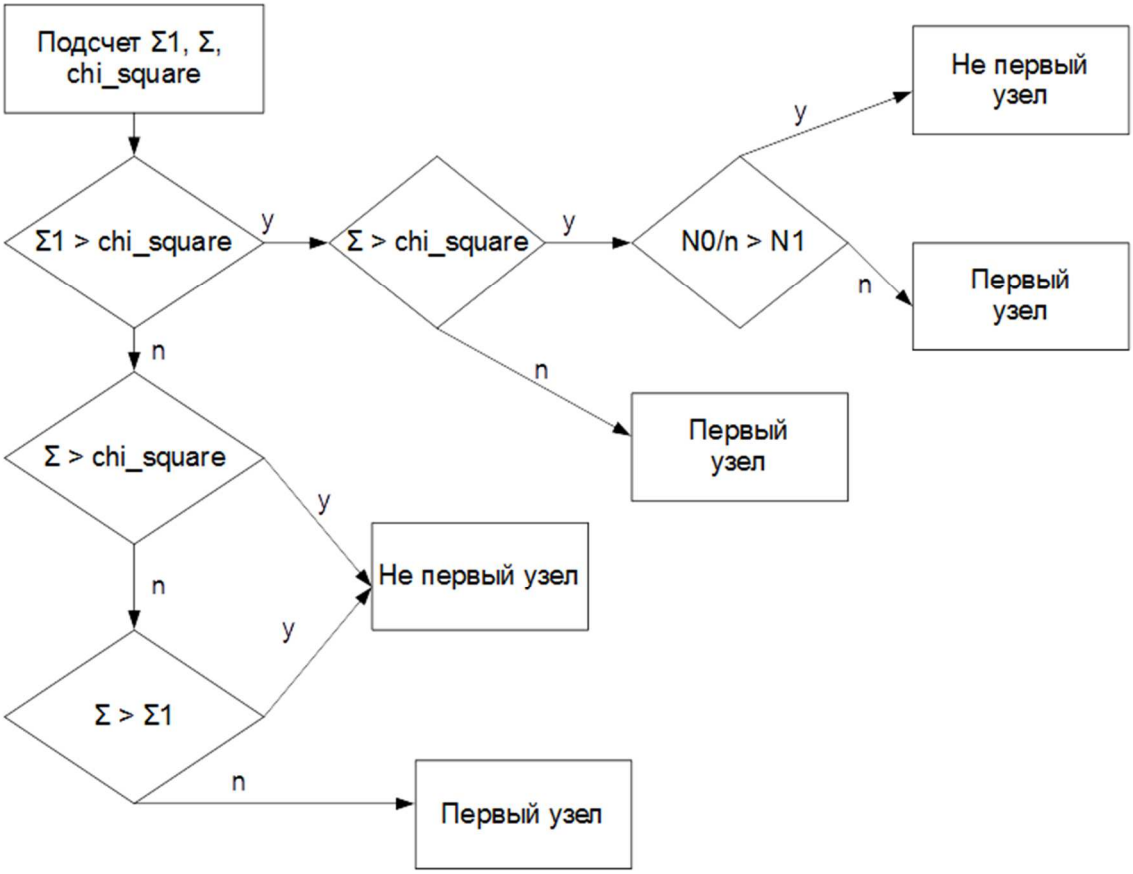

Рисунок 5. *Блок-схема алгоритма распределения данных*

Таким образом, при сохранении репрезентативности в обоих вариантах мы должны обратить внимание на вопрос равномерного распределения данных в хранилище. Если же в обоих случаях нужную степень вероятности репрезентативности сохранить не удается, то выбираем тот вариант, который ближе к необходимому нам уровню. В других ситуациях решение принимается в пользу более репрезентативного результата.

Отметим также, что при заполнении хранилища «с нуля» алгоритм принятия решения будет другой, пока не наберется нужное разнообразие данных в хранилище. Как именно изменится алгоритм, будет описано ниже.

3.6. Вычисление функции распределения хи-квадрат

Важным вопросом в организации нашего хранилища является вычисление функции распределения хи-квадрат и подсчет квантилей этого распределения. Функция распределения хи-квадрат выражается следующим образом:

$$F_{\chi^2(k)}(x) = \frac{\gamma\left(\frac{k}{2}, \frac{x}{2}\right)}{\Gamma\left(\frac{k}{2}\right)},$$

где $\Gamma(z)$ и $\gamma(a, z)$ гамма-функция и неполная гамма функция соответственно. Они равны следующему:

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt, z \in \mathbb{C}: Re(z) > 0,$$

$$\gamma(a, z) = \int_0^z e^{-t} t^{a-1} dt.$$

Очевидно, что даже численное приблизительное вычисление для каждого шага анализа данных затруднительно. Поэтому в нашей модели мы воспользовались аппроксимациями для квантилей распределения хи-квадрат.

Для получения приближенных значений квантилей распределения хи-квадрат $\chi^2_{\alpha,n}$ существуют аппроксимации:

- **Аппроксимация Корниша-Фишера [20]:**

$$\chi^2_{\alpha,n} = n + A\sqrt{n} + B + \frac{C}{\sqrt{n}} + \frac{D}{n} + \frac{E}{n\sqrt{n}},$$

  где:

$$A = d\sqrt{2}$$

$$B = \frac{2}{3}(d^2 - 1)$$

$$C = d * \frac{d^2 - 7}{9\sqrt{2}}$$

$$D = \frac{6d^4 + 14d^2 - 32}{405}$$

$$E = d * \frac{9d^4 + 256d^2 - 433}{4860\sqrt{2}}$$

$$d = \begin{cases} 2.0637 * \left(ln\frac{1}{1-\alpha} - 0.16\right)^{0,4274}, \ 0.5 \ \leq \alpha \leq 0.999 \\ -2.0637 * \left(ln\frac{1}{\alpha} - 0.16\right)^{0,4274}, \ 0.001 \ \leq \alpha < 0.5 \end{cases}$$

- **Аппроксимация Голдштейна [21]**

$$\chi^2_{\alpha,n} = n * \left[\sum_{i=0}^{6} n^{-\frac{i}{2}} * d^i * \left(a_i + \frac{b_i}{n} + \frac{c_i}{n^2}\right)\right]^3$$

где d определяется аналогично, а коэффициенты a, b, c приведены в таблице:

i	a	b	C
1	1.0000886	-0.2237368	-0.01513904
2	0.4713941	0.02607083	-0.008986007
3	0.0001348028	0.01128186	0.02277679
4	-0.008553069	-0.01153761	-0.01323293
5	0.00312558	0.005169654	-0.006950356
6	-0.0008426812	0.00253001	0.001060438
7	0.00009780499	-0.001450117	0.001565326

Таблица 3. *Коэффициенты для аппроксимации Голдштейна*

В реализованной модели для данной работы использовалась аппроксимация Корниша-Фишера как наиболее часто встречающаяся для составления таблиц квантилей.

Чтобы не приходилось выполнять вычисления на каждом шаге добавления данных в хранилище, можно, зная заранее максимальное количество измерений D и требуемую вероятность репрезентативности α, вычислить все возможно необходимые для дальнейших вычислений квантили распределения хи-квадрат.

3.7. Проблема заполнения пустого хранилища

В процессе заполнения пустого хранилища возникает следующая проблема: при малом количестве данных высока вероятность того, что эти данные по измерению попадут в один интервал. В этом случае критерий хи-квадрат не применим, так как мы получаем нулевую степень свободы для вычисления квантилей. Следовательно, в данной ситуации необходимо принимать решения иным способом. В нашей модели реализовано решение, позволяющее заполнение только первого узла в случае, если все данные в хранилище принадлежат одному интервалу измерения. Это обеспечит максимальную репрезентативность первого узла. Однако, если хранилище будет довольно долго заполняться такими данными, возникнет сильная перегрузка на первом узле по сравнению с другими узлами, что, говоря вообще, недопустимо. Но так как в данной работе мы рассматриваем вопрос статистического анализа как основной, мы используем именно это решение. Если понадобится сохранять равномерность распределения данных в хранилище и другие возможные условия, следует задать «вес» этим условиям и в ситуации, описанной выше, принимать решения согласно «взвешиванию» нарушения условий.

3.8. Оптимизация алгоритма распределения данных

При большом потоке данных для заполнения хранилища нам крайне важна скорость принятия решения. Поэтому нам необходимо как можно лучше оптимизировать наш алгоритм. Сделаем это следующим образом: вместо того, чтобы считать каждый раз две статистических суммы, будем хранить значение суммы после предыдущего шага и вычислять лишь добавку к сумме для случая добавления в общую выборку и добавления в обе выборки.

Напомним выражение для статистической суммы при α_i и β_i количестве записей для интервала критерия i в общей выборке и в выборке первого узла соответственно. Тогда общее количество данных и количество данных на первом узле будет:

$$n_0 = \sum_{i=1}^{D} \alpha_i \text{ и } n_1 = \sum_{i=1}^{D} \beta_i$$

соответственно. Запишем выражение для статистической суммы:

$$\check{X}^2_{n_0,n_1} = (n_0 + n_1) \sum_{j=1}^{D} \left(\frac{1}{n_0(\alpha_j+\beta_j)} \left(\alpha_j + \frac{n_0(\alpha_j+\beta_j)}{n_0+n_1} \right)^2 + \frac{1}{n_1(\alpha_j+\beta_j)} \left(\beta_j + \frac{n_1(\alpha_j+\beta_j)}{n_0+n_1} \right)^2 \right).$$

Упростим данную формулу, получим следующее:

$$\check{X}^2_{n_0,n_1} = (n_0 + n_1) \sum_{j=1}^{D} \frac{1}{(\alpha_j+\beta_j)} \left[\frac{\alpha_j^2}{n_0} + \frac{\beta_j^2}{n_1} \right] + 3(n_0 + n_1).$$

Теперь значительно проще рассмотреть случай добавления элемента с интервалом измерения r в общую выборку и в обе выборки:

$$\check{X}^2_{n_0+1,n_1} = (n_0 + n_1 + 1) \sum_{j=1, j\neq r}^{D} \frac{1}{(\alpha_j + \beta_j)} \left[\frac{\alpha_j^2}{n_0 + 1} + \frac{\beta_j^2}{n_1} \right] + 3(n_0 + n_1 + 1) + \frac{(n_0 + n_1 + 1)}{(\alpha_r + 1 + \beta_r)} \left[\frac{(\alpha_r + 1)^2}{n_0 + 1} + \frac{\beta_j^2}{n_1} \right]$$

$$\check{X}^2_{n_0+1,n_1+1} = (n_0 + n_1 + 2) \sum_{j=1, j\neq r}^{D} \frac{1}{(\alpha_j+\beta_j)} \left[\frac{\alpha_j^2}{n_0+1} + \frac{\beta_j^2}{n_1+1} \right] + 3(n_0 + n_1 + 2) + \frac{(n_0+n_1+2)}{(\alpha_r+2+\beta_r)} \left[\frac{(\alpha_r+1)^2}{n_0+1} + \frac{(\beta_r+1)^2}{n_1+1} \right].$$

Найдем и упростим разницы $\delta_0 = \check{X}^2_{n_0+1,n_1} - \check{X}^2_{n_0,n_1}$ и $\delta_1 = \check{X}^2_{n_0+1,n_1+1} - \check{X}^2_{n_0,n_1}$. Получим:

$$\delta_0 = \frac{n_1}{n_0(n_0+1)}\sum_{j\neq r}^{D}\frac{\alpha_j^2}{\alpha_j + \beta_j} + \frac{1}{n_1}\sum_{j\neq r}^{D}\frac{\beta_j^2}{\alpha_j + \beta_j} + \Delta_1 + 3$$

где $\Delta_1 = \frac{n_0+n_1+1}{\alpha_r+\beta_r+1}\left[\frac{(\alpha_r+1)^2}{n_0+1} + \frac{\beta_r^2}{n_1}\right] - \frac{n_0+n_1}{\alpha_r+\beta_r}\left[\frac{\alpha_r^2}{n_0} + \frac{\beta_r^2}{n_1}\right]$.

$$\delta_1 = \frac{n_1 - n_0}{n_0(n_0+1)}\sum_{j\neq r}^{D}\frac{\alpha_j^2}{\alpha_j + \beta_j} + \frac{n_0 - n_1}{n_1(n_1+1)}\sum_{j\neq r}^{D}\frac{\beta_j^2}{\alpha_j + \beta_j} + \Delta_2 + 6$$

где $\Delta_1 = \frac{n_0+n_1+2}{\alpha_r+\beta_r+2}\left[\frac{(\alpha_r+1)^2}{n_0+1} + \frac{(\beta_r+1)^2}{n_1+1}\right] - \frac{n_0+n_1}{\alpha_r+\beta_r}\left[\frac{\alpha_r^2}{n_0} + \frac{\beta_r^2}{n_1}\right]$.

Таким образом, мы можем дополнительно хранить значения:

$$\gamma_0 = \sum_{j=1}^{D}\frac{\alpha_j^2}{\alpha_j + \beta_j},$$

$$\gamma_1 = \sum_{j=1}^{D}\frac{\beta_j^2}{\alpha_j + \beta_j}$$

С помощью которых вычислять статистическую сумму за время O(1), а не за O(D). При малых D это упрощение не имеет смысла, однако при D > 20 имеет смысл перейти к этому оптимизированному способу расчета.

4. Результаты проведенных тестов

Модель хранилища была реализована на языке Python. Выполнялись тесты на сохранение репрезентативности выборки при добавлении в хранилище метеорологических данных: показания среднесуточной температуры за выбранную дату на выбранной станции. Общее количество записей составило 62 млн, что позволяло оценить работу алгоритма случайного распределения и собственного алгоритма на большой общей выборке данных.

Сначала были проведены тесты случайного распределения, чтобы выяснить те области параметров хранилища, в которых требуется улучшение репрезентативности за счет собственного алгоритма. Затем, были проведены тесты на собственном алгоритме и результаты этих тестов сравнивались с результатами случайного распределения.

Отметим, что в нашем случае было 3 параметра, которые варьировались:

- Количество загружаемых в хранилище данных N;
- Количество узлов в хранилище k;
- Количество интервалов измерения, по которому должна сохраняться репрезентативная выборка M;

Для собственного алгоритма есть еще один варьируемый параметр: минимальная вероятность репрезентативности $\alpha_{\min}$ выборки на первом узле.

Полученные результаты рассмотрим подробнее.

4.1. Тесты для случайного распределения

Сначала были проведены тесты для небольшого количества данных:

- N от 10 до 10240

- K от 2 до 5
- M от 12 до 120

Для случая малого количества данных результат предсказуем: случайное распределение дает непредсказуемый результат по вероятности репрезентативности:

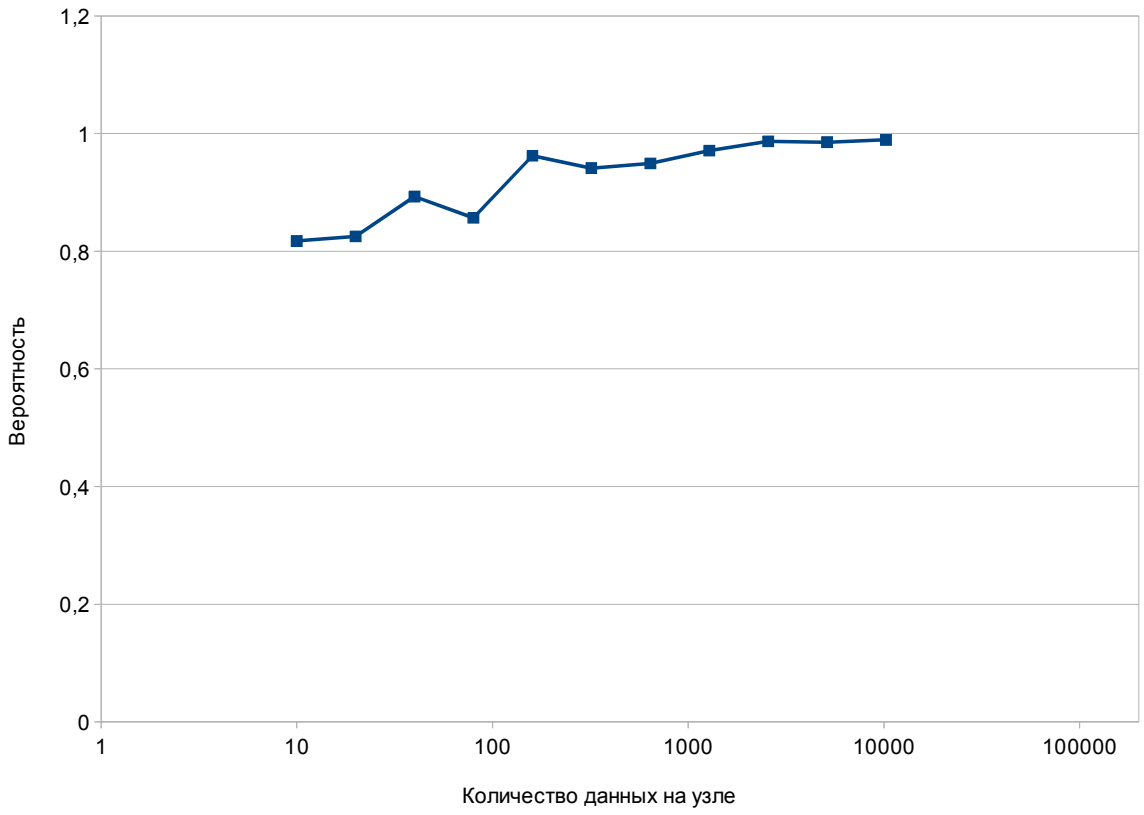

Рисунок 6. *Зависимость вероятности репрезентативности выборки на узле от количества узлов для M = 60, k = 2*

Следует отметить, что для каждого набора параметров было проведено 10 независимых измерений и в анализе использовался средний результат. Также отметим, что ошибка вычислений вероятности составила 0,01.

Для случая других параметров результаты те же:

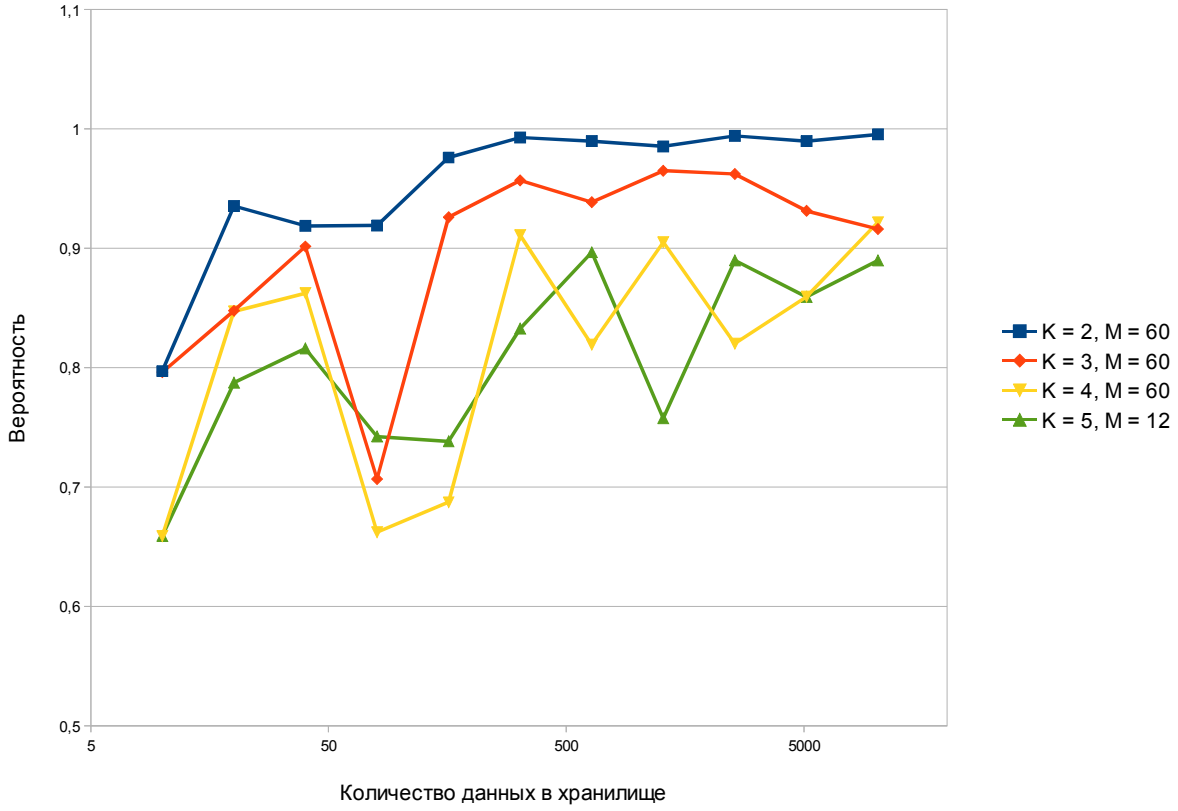

Рисунок 7. *Зависимость вероятности репрезентативности данных на первом узле от количества данных в хранилище для разного количества серверов*

Ситуация похожа как и для разного количества серверов (рис. 7), так и для разного количества интервалов (рис. 8).

Из этих графиков видно, что есть зависимость репрезентативности выборки от количества серверов и от количества интервалов. Но её характер исследовался на большом количества данных и будет описан ниже.

Из рисунков 6 – 8 видно, что при малом количестве данных вероятность репрезентативности выборки на первом узле не может быть удовлетворительной. Однако, данная область мало интересна для случая распределенного хранилища, так как количество записей, равное 10 000 не является сколь угодно близким к реальному количеству записей в хранилище.

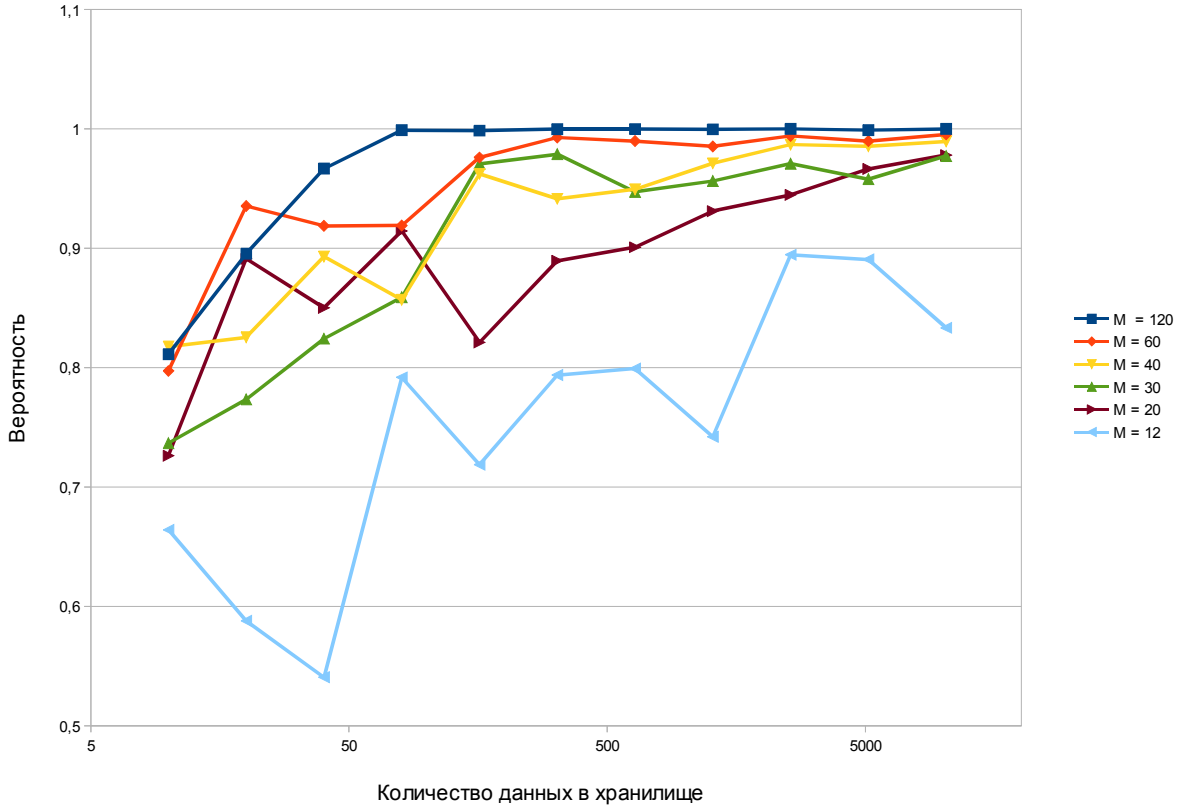

Рисунок 8. *Зависимость вероятности репрезентативности распределения данных на первом узле от количества данных в хранилище для разных количеств интервалов*

Далее был проведен тест для большего количества данных, а именно следующих значений параметров:

- N от 10 000 до 10 240 000
- K от 2 до 4
- M от 12 до 120

В результате было получено следующее (рис.9 и рис. 10). Видно, что качество выборки улучшается с количеством данных, но остается зависимость от количества узлов (рис. 9) и количества интервалов распределения (рис. 10).

Особенно непредсказуемый результат получается для случая малого количества интервалов измерения.

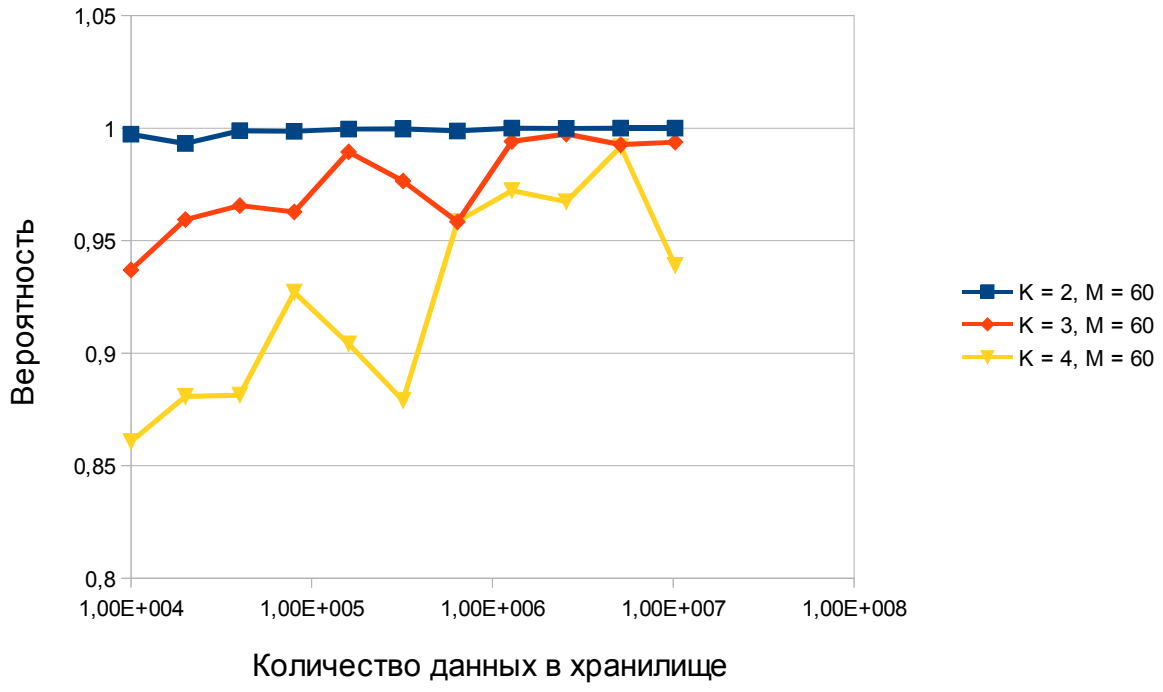

Рисунок 9. *Зависимость вероятности репрезентативности выборки от количества данных в хранилище для различных количеств узлов*

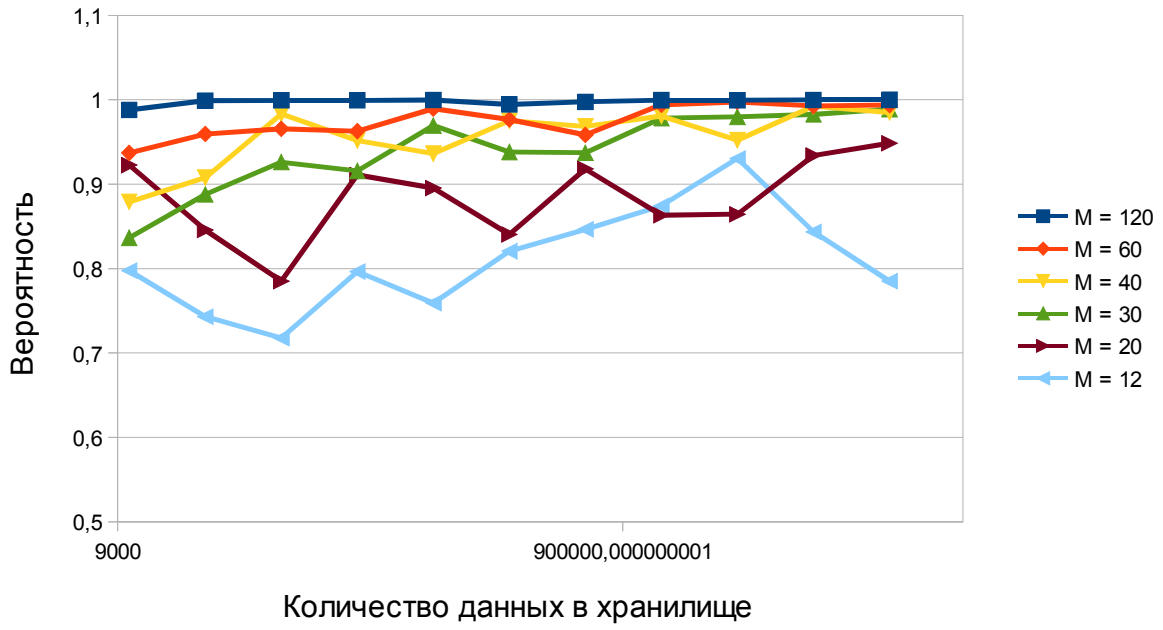

Рисунок 10. *Зависимость вероятности репрезентативности от количества данных в хранилище для разного количества интервалов измерения*

Также были проведены измерения для большего количества данных (до 62 000 000) и построен общий график зависимости вероятности репрезентативности выборки от количества данных и от количества интервалов критерия. Количество серверов было фиксированным и равным 4. Сам график приведен на рис. 11.

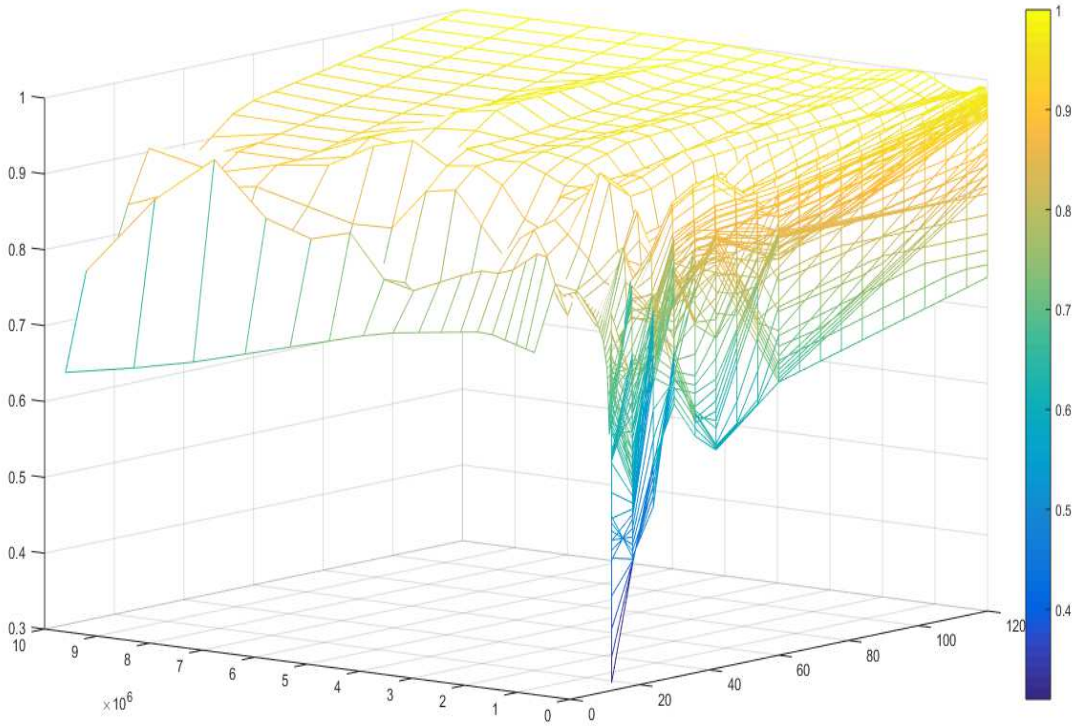

Рисунок 11. *График зависимости вероятности репрезентативности выборки от количества данных и количества интервалов измерения*

Из общего графика видно, что вероятность репрезентативности низкая у малого количества интервалов измерения даже при большом количестве данных. Следовательно, это та область, где случайное распределение дает плохой результат репрезентативности.

Приведем еще отдельно график зависимости вероятности репрезентативности от количества узлов в хранилище для случая N = 62 000 000 и M = 60 (рис. 12).

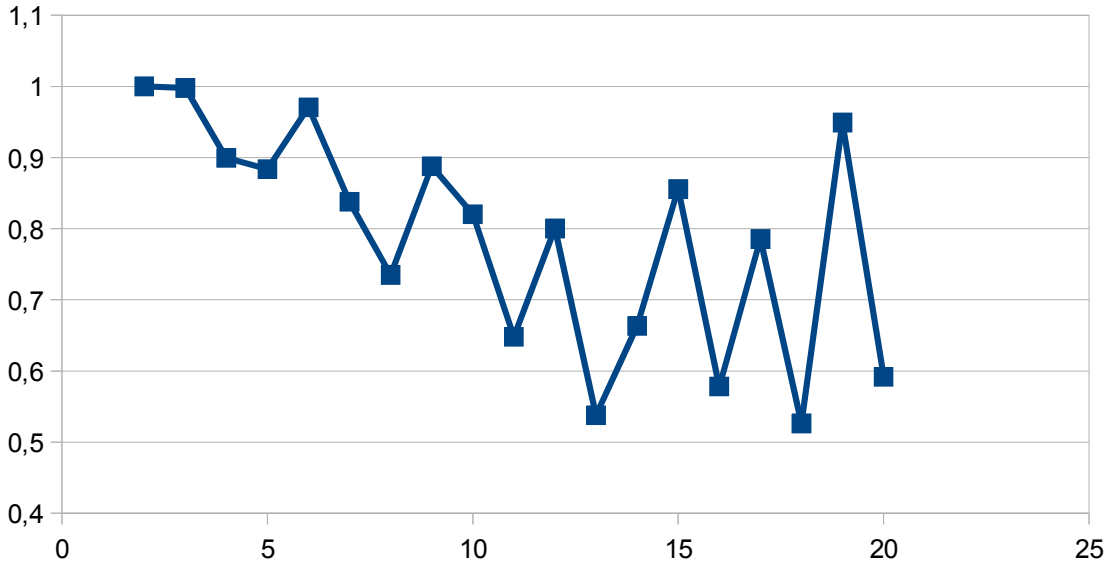

Рисунок 12. *Зависимость вероятности репрезентативности выборки от количества узлов в хранилище*

Выделить общую область для сравнения алгоритмов затруднительно по той причине, что результат работы нашего алгоритма зависит от заданного порога минимальной вероятности репрезентативности. Однако, рассмотрев несколько случаев значения этой вероятности, вполне можно сравнить результаты работы обоих алгоритмов. Что и будет сделано в следующем пункте.

4.2. Тесты для собственного алгоритма распределения

Тесты для собственного алгоритма распределения проводились следующим образом: был включен контролирующий модуль, в котором независимо от модуля распределения подсчитывалась статистическая сумма для критерия хи-квадрат. После принятия решения о распределении очередной записи контролирующий модуль вычислял текущую вероятность репрезентативности выборки на первом узле и запоминал её. В конце теста результаты записывались в итоговый файл. Таким образом, ввиду детер-

минированности нашего алгоритма, мы получаем степень репрезентативности для всех количеств данных от 0 до максимальной для теста. Чтобы проверить алгоритм для разных условий, был проведен тест с другим порядком загрузки данных – начиная не с первого, а с другого файла. Однако на результат при больших N это не повлияло.

Важный результат такого исследования – это возможность понять динамику репрезентативности выборки при добавлении данных в хранилище, а также её зависимость от выбранного минимального порога вероятности. Для наглядности приведем график зависимости вероятности репрезентативности выборки от порядкового номера поступающей записи (рис.13) для M = 120, N(max) = 50000, k = 2 и порога вероятности $\alpha_{min} = 0{,}9$.

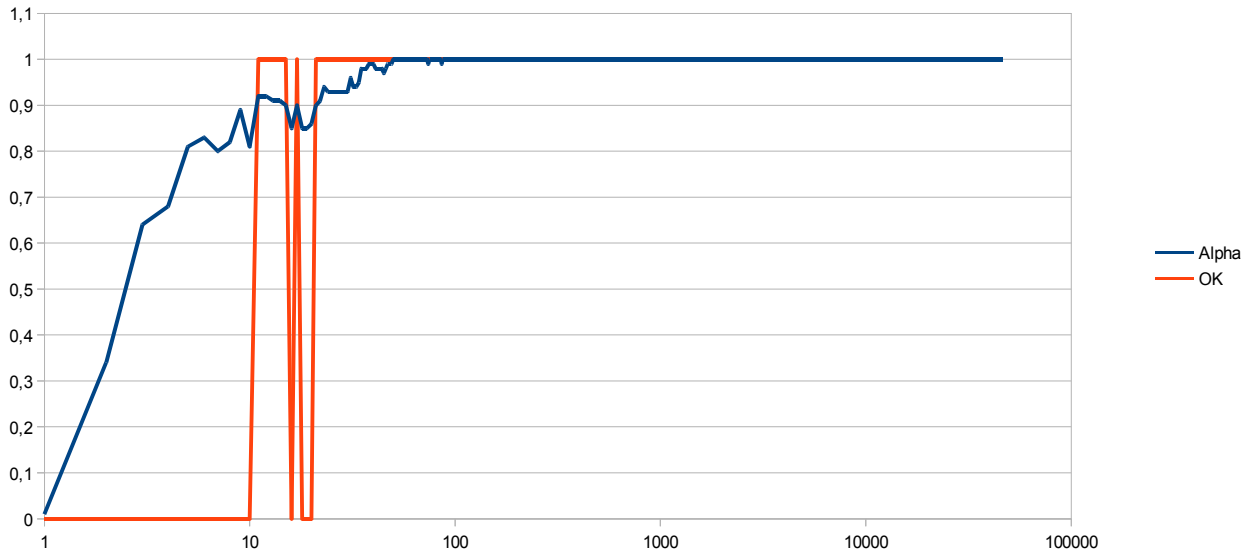

Рисунок 13. *Зависимость вероятности репрезентативности от номера поступающего элемента*

На этом графике синяя линия (Alpha) – вероятность, а красная (OK) – функция от вероятности следующего вида:

$$f(\alpha) = \begin{cases} 1, \alpha \geq \alpha_{min}, \\ 0, \alpha < \alpha_{min}. \end{cases}$$

Такая функция позволяет проследить ситуации, когда нарушается требуемый уровень репрезентативности выборки на первом узле.

Если мы посмотрим на аналогичный результат для случая $\alpha_{min} = 0{,}99$, то увидим, что моменты нарушения репрезентативности происходят «дольше» до достижения определенного количества данных в хранилище (рис. 14).

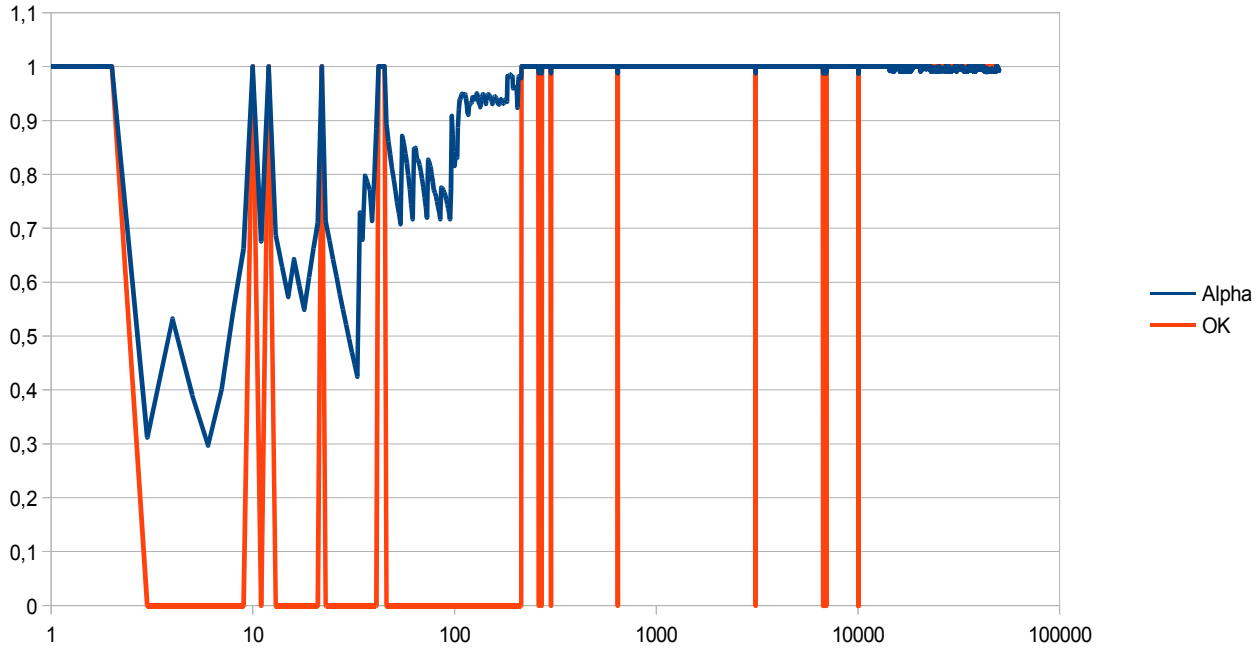

Рисунок 14. *Зависимость вероятности репрезентативности выборки на первом узле от номера поступающей записи*

Ввиду таких различий между результатами для разных пороговых вероятностей, рассмотрим отдельно случай $\alpha_{min} = 0{,}9; 0{,}95; 0{,}975; 0{,}99$.

- **α_{min} = 0,9**

  Приведем график зависимости вероятности репрезентативности выборки на первом узле для алгоритма случайного распределения и нашего алгоритма (рис. 15).

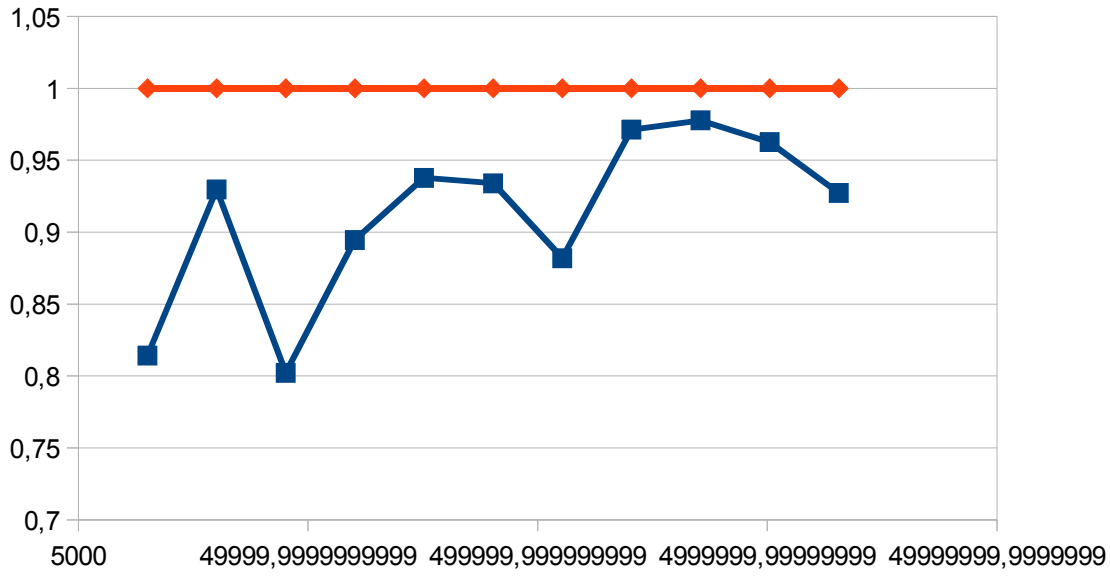

Рисунок 15. *Зависимость вероятности репрезентативности от количества данных для k = 2, M = 100*

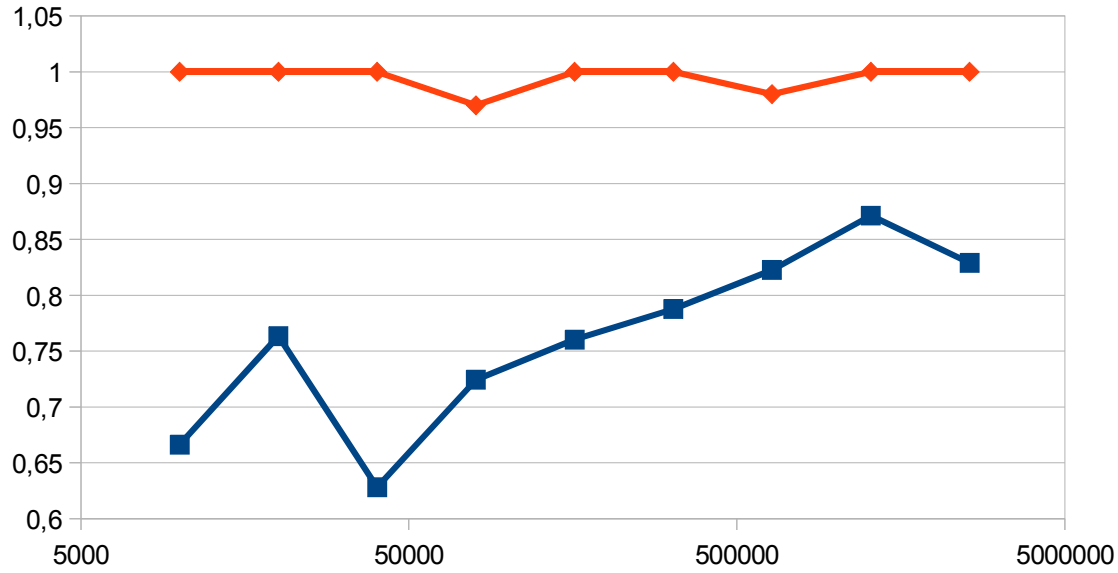

Рисунок 16. *Зависимость вероятности репрезентативности для k = 4, M = 12*

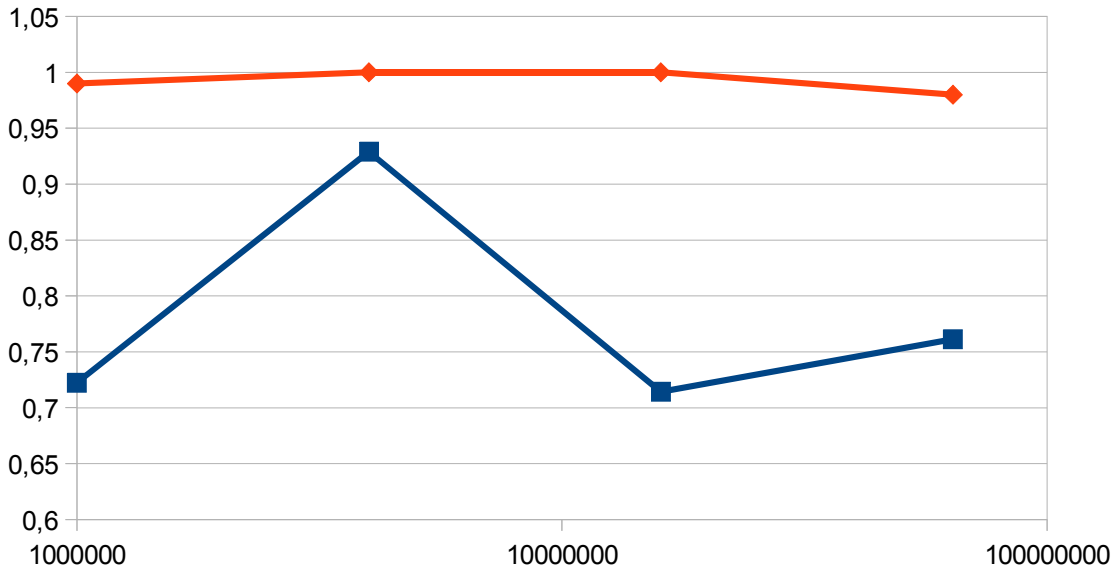

Рисунок 17. *Зависимость вероятности репрезентативности от количества данных при k = 10 M = 40*

Из результатов экспериментов видно, что наш алгоритм дает стабильно лучший результат, чем алгоритм случайного распределения на этих же данных. Это выполняется как и для случая малого количества интервалов критерия, так и для случая большого количества серверов.

- α_{min} **= 0,95**

  При таком заданном пороге ситуация практически не меняется – наш алгоритм выдает стабильно хороший результат, а случайное распределение выходит на хороший результат только при очень большом количестве данных.

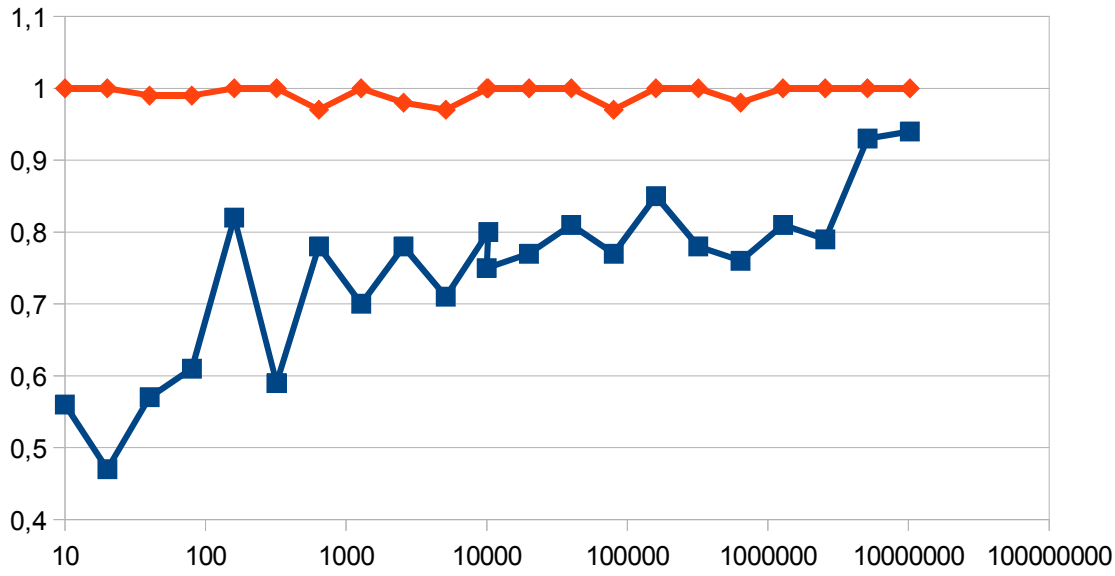

Рисунок 18. *Зависимость вероятности репрезентативности от количества данных при M = 20 k = 4*

- α_{min} **= 0,99**

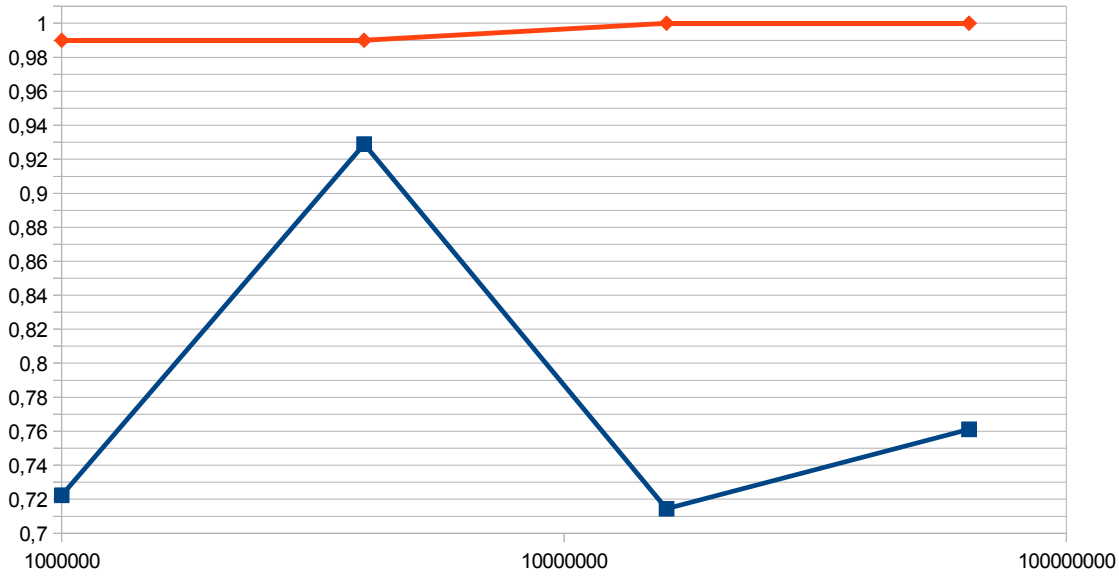

Рисунок 19. *Зависимость вероятности репрезентативности от количества данных в хранилище для k =10, M = 20*

Исходя из экспериментальных данных, мы можем утверждать, что в случае с небольшим количеством критериев (M < 30) и/или с большим количеством серверов (k > 8) алгоритм случайного распределения данных дает результат хуже и не стабильнее, чем наш алгоритм.

Таким образом, мы можем утверждать, что разработка собственного алгоритма, обеспечивающего репрезентативность выборки на выбранном узле для выбранного измерения выполнена успешно.

Однако протестирована лишь незначительная часть возможностей хранилища, оптимизированного под статистический анализ. Действительно, обеспечивая репрезентативность лишь по одному измерению, мы позволяем проводить только одномерный статистический анализ. Например, мы можем только проанализировать распределение всех показателей среднесуточной температуры, не различая, в каких местах и когда они были сняты. Это своего рода измерение «средней температуры по больнице», которое может быть выполнено средствами Hadoop над всеми данными. А такие интересные и сложные виды анализа, как факторный или корреляционный, которые плохо реализуются средствами Hadoop и MapReduce, проводить на такой выборке мы не можем. Однако, наше решение легко масштабируется на случай поддержки репрезентативности нескольких измерений на одном узле. Каким образом – будет рассказано в следующей главе.

5. Расширение задачи

Как было сказано выше, обеспечение репрезентативности только одного критерия не дает нам столь широких возможностей анализа, как хотелось бы. Действительно, мы можем выполнять на выборке по одному измерению лишь определительный анализ: узнать её математическое ожидание, дисперсию и другие параметры распределения, однако факторный, корреляционный и регрессионный виды анализа нам недоступны.

Оказывается, наше решение с помощью критерия хи-квадрат хорошо масштабируется на случай поддержки репрезентативности сложной выборки для двух и более измерений одновременно. Рассмотрим этот случай подробно.

5.1. Репрезентативность по двум и более распределениям на одном узле

Будем рассматривать задачу анализа на данных, которые использовались в тестах. Помимо среднесуточной температуры, каждая запись определяется датой замера и номером станции. Допустим, мы хотим не только знать среднюю температуру вообще, а исследовать зависимость параметров температуры от номера станции. Рассмотрим, как это можно сделать средствами хи-квадрат.

Так как критерий хи-квадрат не опирается на то, что нам известно распределение данных, а для его применения важно знать только частотные характеристики попадания данных под определенные интервалы и критерии, то мы можем «расширить» наши интервалы таким образом, чтобы для каждого интересующего нас интервала температуры рассматривать и распространенность её в выбранной нами станции.

Например, у нас была следующая частотная таблица для нашей выборки на узле:

№ интервала	1	2	3	4	5	6	7
Количество данных, попавших в интервал	n_1	n_2	n_3	n_4	n_5	n_6	n_7

Таблица 3. *Таблица частот для одного измерения*

Теперь, рассматривая еще и номер станции, мы можем составить следующую таблицу частот:

№ интервала	1	2	3	4	5	6	7
Станция №1	n_{11}	n_{21}	n_{31}	n_{41}	n_{51}	n_{61}	n_{71}
Станция №2	n_{12}	n_{22}	n_{32}	n_{42}	n_{52}	n_{62}	n_{72}
Станция №3	n_{13}	n_{23}	n_{33}	n_{43}	n_{53}	n_{63}	n_{73}
Станция №4	n_{14}	n_{24}	n_{34}	n_{44}	n_{54}	n_{64}	n_{74}
Станция №5	n_{15}	n_{25}	n_{35}	n_{45}	n_{55}	n_{65}	n_{75}

Таблица 4. *Таблица количества данных для заданного интервала и номера станции*

При этом выполняется следующее: $\mathrm{n_i} = \sum_{j=1}^{5} n_{ij}$ для всех i от 1 до 7.

Статистическая сумма тогда подсчитывается не по количеству интервалов M, а по количеству интервалов, помноженному на количество станций (или, в общем случае, количество интервалов второго измерения).

Если мы захотим проводить анализ не только в зависимости от номера станции, но и в зависимости от года, когда снималась температура, наша таблица «превратится» в параллелепипед, по третей стороне которого будет варьироваться год измерения. Тогда статистическая сумма будет высчитываться по количеству интервалов температуры, помноженной на количество станций и количество лет измерений.

Очевидно, что вычисления на каждом шаге добавления данных возрастают с увеличением количества измерений, репрезентативность которых мы хотим обеспечить на выбранном узле. Тогда поможет оптимизация, описанная в пункте 3.7.

5.2. Репрезентативность двух и более измерений на разных узлах

В нашей работе мы затрагивали вопрос равномерного распределения данных как одно из требований к алгоритму распределения. При этом приоритет репрезентативности данных был для обеспечения репрезентативности, и лишь при условии сохранения репрезентативности, решение о распределении данных принималось согласно требованию равномерной загрузки хранилища. Но помимо этого требования, мы можем поставить другие требования к хранилищу, выполнение которых нам необходимо.

Одним из таких требований может быть репрезентативность выборки по другим измерениям на другом узле. В этом случае мы обязаны задать «вес» наших требований и, исходя из ситуации, принимать решения в сторону того или иного требования. Предположим, на первом узле мы хотим иметь репрезентативную выборку по температуре, на втором – по станциям. Например, для чуть важнее распределение по температуре, нежели чем по станциям, поэтому мы присваиваем этим требованиям вес 0,6 и 0,4 соответственно. Так же нас интересует равномерное распределение данных на хранилище, но только при выполнении первых двух требований.

Допустим, мы посчитали статистическую сумму и квантиль хи-квадрат для первого и второго требования. Мы стоим перед выбором из трех путей: положить данные на первый узел, второй или остальные. Тогда решение принимается исходя из разницы между квантилями и соответствующими статистическими суммами. Выявляя процент и умножая на соответствующий вес, мы можем сделать характеристику решений и выбирать точно исходя из этих характеристик.

Конечно, мы изложили лишь приблизительный способ решения данной проблемы. Этот вопрос требует дальнейшего плотного изучения, моделирования и тестирования, которое будет проводиться в будущих работах.

Заключение

В данной работе было проведено исследование на тему организации распределенного хранилища, оптимизированного под статистический анализ. Была рассмотрена задача распределения данных в хранилище таким образом, чтобы создать репрезентативную выборку по выбранному измерению на выбранном узле распределенного хранилища. Была построена математическая модель задачи, позволяющая использовать критерий хи-квадрат для определения вероятности репрезентативности выборки.

На основе критерия хи-квадрат был построен алгоритм распределения данных, дающий стабильный хороший результат для любых наборов параметров, обеспечивающий не только репрезентативность с заданным порогом, но и равномерность распределения данных по узлам хранилища.

Результаты работы разработанного алгоритма были сравнены с результатами работы алгоритма случайного распределения данных, дающего нестабильный результат в случае малого количества интервалов критерия и/или большого количества узлов в хранилище.

Также было произведено обобщение и расширение текущего алгоритма для случая репрезентативности выборки на узле для двух и более измерений и предложена оптимизация вычисления статистической суммы для этого случая.

В продолжении работы над этой тематикой планируется рассмотреть случай репрезентативности многих критериев и случай репрезентативности на разных узлах на практике для выявления закономерностей и разработки оптимального алгоритма распределения данных в этом случае.

Список литературы

1. Статья «Хранилище данных»: https//ru.wikipedia.org/wiki/Хранилище_данных
2. Цикл статей «Расширение хранилища данных: Большие данные и расширение хранилища данных»: http://www.ibm.com/developerworks/ru/library/ba-augment-data-warehouse1/
3. Статья «Big Data Technology»: https://datajobs.com/what-is-hadoop-and-nosql
4. Статья «NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison» by A B M Moniruzzaman and Syed Akhter Hossain
5. Материал «An Oracle White Paper «Oracle NoSQL Database»: Сентябрь 2011
6. Материалы к презентации «MapReduce – Konzept» Thomas Findling, Thomas König, Universität Leipzig: http://dbs.uni-leipzig.de/file/seminar_0910_findling_K%C3%B6nig.pdf
7. Статья «Статистические методы анализа в клинической практике» Румянцев П.О., Саенко В.А., Румянцева У.В., Чекин С.Ю
8. А.Н.Ширяев. Вероятность. Том 1. МЦНМО, 2007.
9. Елисеева И. И., Юзбашев М. М. Общая теория статистики: Учебник / Под ред. И. И. Елисеевой. — 4-е издание, переработанное и дополненное. — Москва: Финансы и Статистика, 2002. — 480 с.
10. Статья «Компонентный анализ»: http://statmethods.ru/konsalting/statistics-metody/118-komponentnyj-analiz.html

11. Шмойлова Р. А. Общая теория статистики: Учебник. М.: Финансы и статистика, 2002

12. Статья «R and Hadoop Data Analysis - RHadoop»: https://bigha-doop.wordpress.com/2013/02/25/r-and-hadoop-data-analysis-rhadoop/

13. Материалы к презентации «INTEGRATING R AND HADOOP FOR BIG DATA ANALYSIS»: http://r-project.ro/workshop2014/pa-pers/w2014_12.pdf

14. «A Handbook of Statistical Analyses Using R»: http://cran.r-pro-ject.org/web/packages/HSAUR/vignettes/Ch_introduction_to_R.pdf

15. Статья «Divide and Recombine with RHIPE»: https://www.datadr.org/

16. Ивченко Г.И., Медведев Ю.И. «Введение в математическую стати-стику»: учебник. – М.: издательство ЛКИ, 2010. – 600с

17. Натан А.А., Гуз С.А., Горбачев О.Г., Гасников А.В., Черноусова Е.О. Математическая статистика: Учебно-методическое пособие. – М.: МФТИ, 2011, изд. 3-е, исп. и дополненное

18. Крамер Г. «Математические метод статистики». - М.: Мир, 1975.

19. *Golberg H., Levine H.* Approximate formulas for the percentage points and normalization of t and χ^2 // AMS. 1945. V.17. P. 216—225.

20. *Goldstein R.B.* Chi-square quantiles, Algorithm 451 // Commun. Assoc. Comp. 1973. V. 16. P. 483—485.