электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 004.65

Обзор графовых баз данных

***Мочалова М.Д.**, студентка*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Системы обработки информации и управления»

Научный руководитель: Гапанюк Ю.Е., к.т.н, доцент,
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Системы обработки информации и управления»
gapyu@bmstu.ru

В последнее время я все чаще слышу о NoSQL и о графовых базах данных в частности. Но воспользовавшись поиском в интернете с удивлением обнаружила, что статей на эту тему не так и много. Поэтому хотелось бы немного подробнее рассказать об этом виде баз данных.

Рассмотрим несколько современных реализаций графовых моделей данных в виде программных продуктов. На рисунке 1 представлена диаграма Исикавы с примерами для каждой группы. Любая из описанных ниже БД предназначена для удобного хранения и доступа к данным, представленным в виде графов большой размерности (миллионы и более узлов и связей между ними). Однако, по своим функциональным возможностям графовые БД можно классифицировать на следующие виды:

- БД с локальным хранением и обработкой графов;
- БД с распределенным хранением и обработкой данных;
- БД в формате «ключ-значение»;
- документо-ориентированные БД;
- надстройки над *SQL*-ориентированными БД;
- графовые БД с моделью *MapReduce*.

БД с локальным хранением и обработкой графов

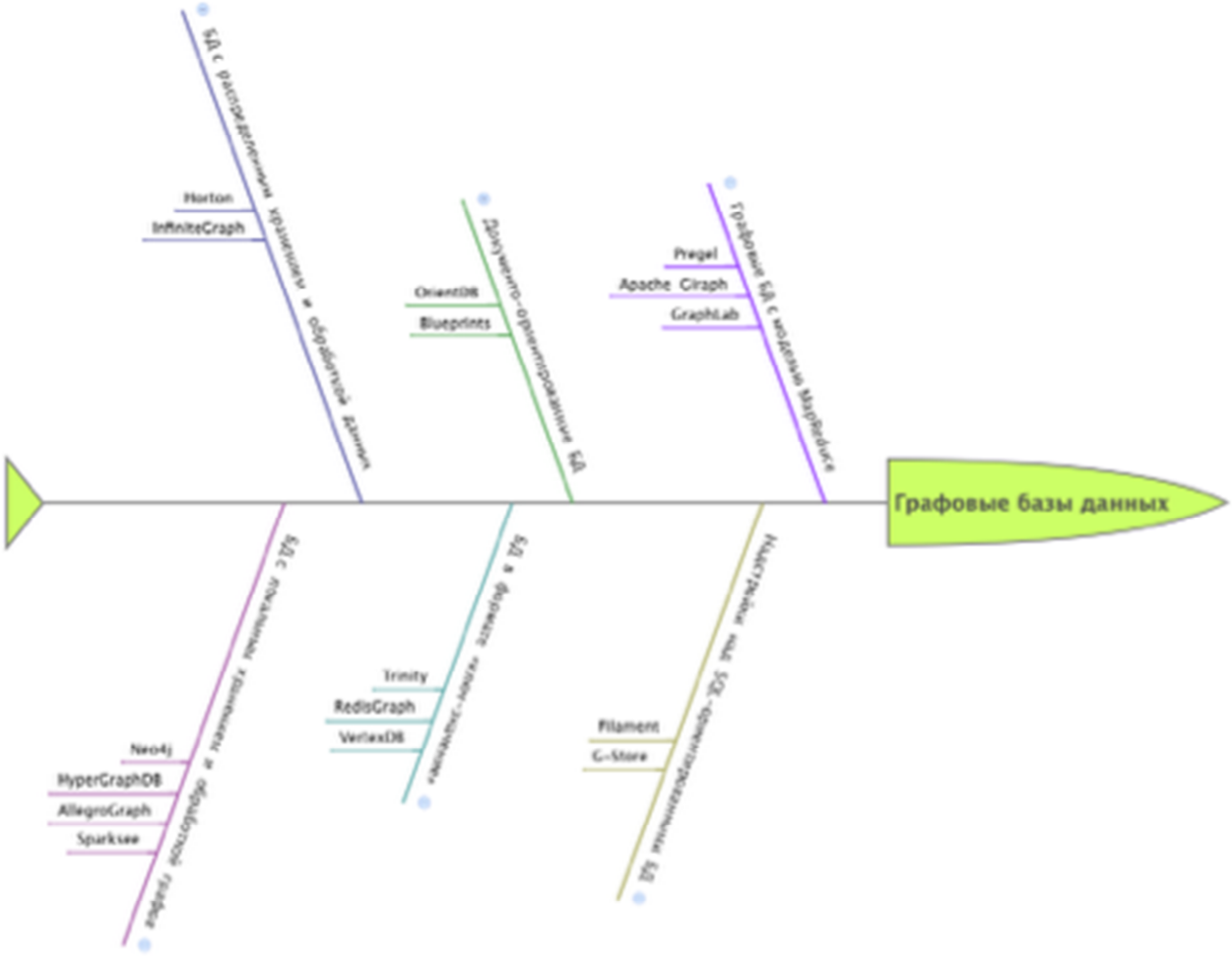

Рис. 1. Диаграма Исикавы графовых быз данных

1. Neo4j
2. *HyperGraphDB*
3. *AllegroGraph*
4. *Sparksee*

Классическим представителем этого вида является полностью транзакционная (*ACID, Atomicity Consistency Isolation Durability,* Атомарность Согласованность Изолированность Надежность) графовая БД *Neo4j* [2].Разработчики описывают свой продукт как «встраиваемая база данных, полнотранзакционный Java-движок, который хранит данные в упорядоченных графах, а не в таблицах». Neo4j имеет свой собственный формат хранилища узлов (node) и связей (relationships). Нативное представление графа, в

отличии от моделирования хранилища на реляционной базе данных, позволяет применять дополнительную оптимизацию в случае данных с более сложной структурой.

К этому же виду баз данных относится *HyperGraphDB*[3] — это расширяемая, портативная, распределенная, встраиваемая система общего назначения со свободным (open-source) механизмом хранения данных. Эта система разработана специально для проектов, использующих возможности искусственного интеллекта и семантического вэба и может использоваться как встраиваемая, объектно-ориентированная база данных для проектов любого масштаба.

Также *AllegroGraph* [4] является представителем данного класса БД.

AllegroGraph - это современная, высокопроизводительная, устойчивая графическая база данных, которая совмещает эффективное использование памяти и хранение на основе жестких дисков, позволяя выполнять миллиарды quads (квадрациклов), сохраняя при этом высокую производительность. AllegroGraph обеспечивает архитектуру протокола REST (существенно расширенный набор Sesame HTTP клиент). Поддерживаются адаптеры для различных языков: Sesame Java, Sesame Jena, Python с использованием Sesame и Lisp (фреймовки). Есть Open Source адаптеры для проектов C #, Ruby, Clojure, Scala и Perl. Архитектура AllegroGraph представлена на рисунке 2.

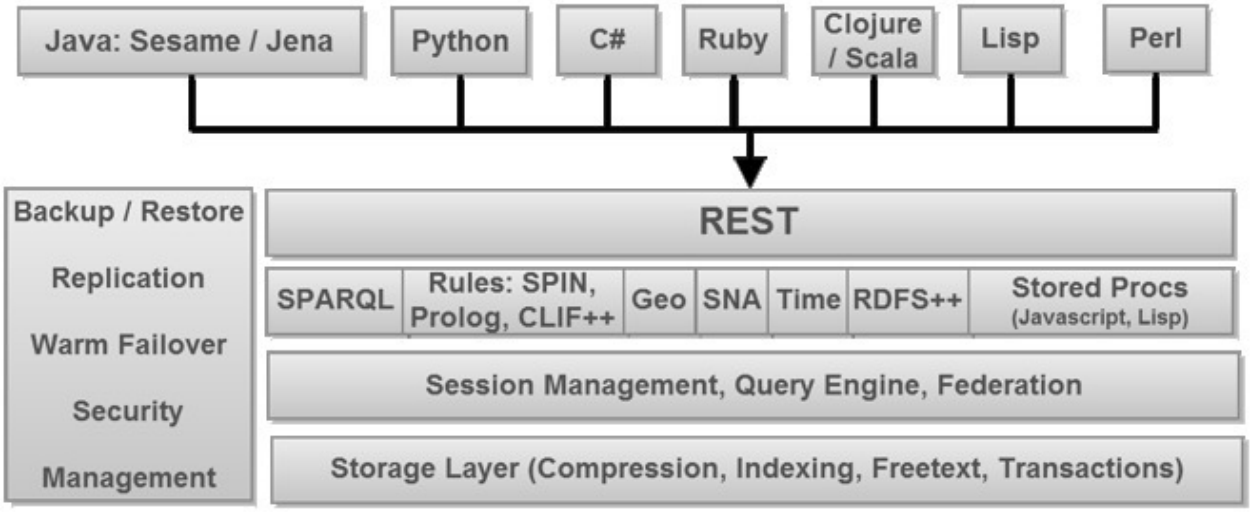

Рис. 2. Архитектура AllegroGraph

Первые две из перечисленных выше являются программами с открытым исходным кодом, а третья имеет бесплатную версию с ограничениями на хранение не более 5 миллионов узлов графа.

К такому виду относится также БД *Sparksee*, поддерживающая целый спектр операционных систем, включая мобильные *iOS* и *Android*. *Sparksee* основана на модели

графовых баз данных, которая в основном характеризуется тремя свойствами: структуры данных являются графы или любые другие структуру, подобные графу; изменение данных и запросы сделаны на основе граф-ориентированных операций; и есть возможность задать ограничение данных, обеспечивающие целостность данных и их связи.

Sparksee [5] граф Размеченный, Направленный, Атрибутный Мультиграф. Размеченный, потому что узлы и ребра графа разделены на типы. Направленный, потому что он поддерживает как направленные ребра, так ненаправленные. Атрибутный, потому что и узлы и ребра Мультиграфа могут иметь атрибуты. Мультиграф обозначает возможность наличия нескольких ребер между одними и теми же узлами, даже если ребра имеют одинаковый тип.

Одной из главных его характеристик является производительность хранилища и возможность поиска для очень больших графов (миллиарды узлов, ребер и атрибутов), реализованных с помощью специализированных структур.

БД с распределенным хранением и обработкой данных

1. *Horton*
2. *InfiniteGraph*

К базам данных второго вида относится исследовательский проект подразделения *Extreme Computing Group* компании *Microsoft Research*, называемый *Horton* (*Querying Large Distributed Graphs*, Обработка больших распределенных графов), основанный на облачной инфраструктуре Orleans.

Также *InfiniteGraph* - распределенная графовая БД, реализованная на Java. Особенностями хранения графов в таких БД является разделение их на подграфы по распределенным сетевым вычислительным комплексам, а затем — организация параллельных вычислений по различным моделям и коммуникационным протоколам. Разработчики использует её для поиска полезных и часто скрытых связей в очень больших объемах сильно-связных данных. InfiniteGraph является кросс-платформенным, масштабируемым решением с возможностью работы в облаке, разработанным для высоких нагрузок. Подходит для приложений и сервисов, решающих различные задачи с графами или отвечающими на такие вопросы как: «как Алиса связана с Бобом?» или «Как дешевле всего добраться из Москвы до Санкт-Петербурга?» с множеством других ограничений на выборку. Эта БД нашла применение как в правительственных сервисах, так и в телеком и других коммерческих сферах.

БД в формате «ключ-значение»

1. *Trinity*
2. *RedisGraph*
3. *VertexDB*

Третий вид графовых БД включает в себя еще один исследовательский проект компании *Microsoft Research* с названием *Trinity*. Это распределенная система обработки графовой информации в глобально адресуемом облаке оперативной памяти, являющаяся хранилищем в формате «ключ-значение» на кластере вычислительных узлов.

База данных *RedisGraph* использует быстрое, размещаемое в оперативной памяти *Redis*-кэш хранилище «ключ-значение» и минималистичный программный интерфейс взаимодействия с ним. Все данные Redis хранит в виде словаря, в котором ключи связаны со своими значениями. Одно из ключевых отличий Redis от других хранилищ данных заключается в том, что значения этих ключей не ограничиваются строками. Поддерживаются следующие абстрактные типы данных:

- Строки (strings). Базовый тип данных Redis. Строки в Redis бинарнобезопасны, могут использоваться так же как числа.
- Списки (lists). Классические списки строк, упорядоченные в порядке вставки, которая возможна как со стороны головы, так и со стороны хвоста списка.
- Множества (sets). Множества строк в математическом понимании: не упорядочены, поддерживают операции вставки, проверки вхождения элемента, пересечения и разницы множеств.
- Хеш-таблицы(hashes). Классические хеш-таблицы или ассоциативные массивы.
- Упорядоченные множества(sorted sets). Упорядоченное множество отличается от обычного тем, что его элементы упорядочены по особому параметру "score".

Тип данных значения определяет, какие операции (команды) доступны для него. Redis поддерживает такие высокоуровневые операции, как объединение и разность наборов, а также их сортировку.

По минималистичности функционала, прежде всего связанного с повышением производительности, не уступает указанной БД и графовая БД *VertexDB*, использующая для запросов *HTTP*-протокол и *JSON* для формата хранения данных графов. Эта БД состоит из узлов, которые представляют собой папки, содержащие пары ключ-значение. Ключи хранятся в алфавитном порядке и могут быть любой строкой не содержащей «/».

Документо-ориентированные БД

1. *OrientDB*
2. *Blueprints*

Для четвертого вида баз данных характерно хранение документов, аналогично как это осуществляется в документо-ориентированных БД, однако, связимежду документами хранятся и обрабатываются посредством графовых моделей и методов. Наиболее известным представителем этого вида БД является *OrientDB*. Эта БД поддерживает *ACID* транзакции, а также интерфейс *Blueprints* для универсального доступа к графовым данным и БД,некоторые из которых ранее перечислены.

OrientDB - открытая СУБД, которая объединяет в себе возможности документо-ориентированной и графо-ориентированной БД. Также поддерживается интерфейс объектно-ориентированной БД, который работает поверх документо-ориентированного слоя.Эта база данных на основе документов, но отношения управляются как в графовых базах с прямым подключением между записями. Она поддерживает схемы: less (слабоструктурированные данные), full (строго задает обязательные поля) и mixed (смешанная). Имеет мощную систему профилирования безопасности, основанную на пользователях и ролях. Поддерживает SQL в качестве языка запросов. Можно вставлять документы, как и любой другой базе данных, основанной на документах, но также поддерживает отношения. Он не использует дорогостоящее JOIN. Вместо этого, OrientDB использует супер-быстрые, постоянные указатели между записями, взятые из мира графовых баз данных. Можно пройти части или целые деревья и графы записей в течение нескольких миллисекунд.

Надстройки над *SQL*-ориентированными БД

1. *Filament*
2. *G-Store*

К графовым БД пятого вида, функционирующим на базе SQL-поддержки относится *Filament*. Более точно, эта БД является библиотекой, предназначенной для хранения и обработки графовых данных, которая выполнена в виде надстройки над базой данных *PostgreSQL* и интерфейса доступа *JDBC*. К такому же виду графовых БД относится *G-Store*, называемая самими разработчиками «*lightweight disk-based manager for graph data*», то есть дисковым менеджером (надстройкой над *PostgreSQL*) для графовых данных.

Графовые БД с моделью MapReduce

1. *Pregel*
2. *Apache Giraph*
3. *GraphLab*

Основной областью применения нереляционных баз данных, к тором относягся графовые БД, являются распределенные системы. Одним из наиболее популярных подходов к распределенной обработке данных является модель MapReduce [6], разработанная Google. Использование этой парадигмы при создании распределенных приложений позволяет программисту избежать необходимые написания специального кода для распараллеливания и синхронизации. MapReduce предполагает разбиение задачи на два шага:

• Map-шаг: главный узел разбивает входные задачи на части и передает остальным узлам для предварительной обработки;

• Reduce-шаг: свертка результатов предварительной обработки. Главный узел получает даные от остальных узлов и формирует окончательный результат выполнения задачи.

Для использования фреймворка MapReduce программисту необходимо написать функции Map и Reduce. Эти фенкции работают с данными, структурированными в виде пар «ключ-значение». Map принимает на вход пару одного типа и возвращает набор пар другого типа (промежуточный результат):

$$Map(Key,Value) \rightarrow list(iKey,iValue),$$

где (Key,Value) - одна запись из набора входных данных, (iKey,iValue) - промежуточная пара ключ/значение. Каждая пара, сгенерированная Map-функцией, отправляется на некоторый узел для выполнения Reduce-шага. Целевой узел определяется посредством применения хеш-фенкции к ключу пары (iKey). На каждом узле полученные пары группируются по ключу. Затем к каждой группе применяется функция Reduce, при выполнении которой формируется окончательный результат fValue:

$$Reduce(iKey,list(iValue)) \rightarrow list(\ fValue)$$

На рисунке 3 приведен классический пример использования модели MapReduce для подсчета частоты встречаемости слов в наборе документов. Map-функция возвращает набор пар «слово - количество» для каждого документа. Затем пары группируются по ключу, в результате чего в каждую группу попадают пары, соответствующие одному и тому же слову. Reduce-функция суммирует значения всех пар в группе и возвращает

окончательный результат, равный количеству вхождений этого слова в текст всех документов набора.

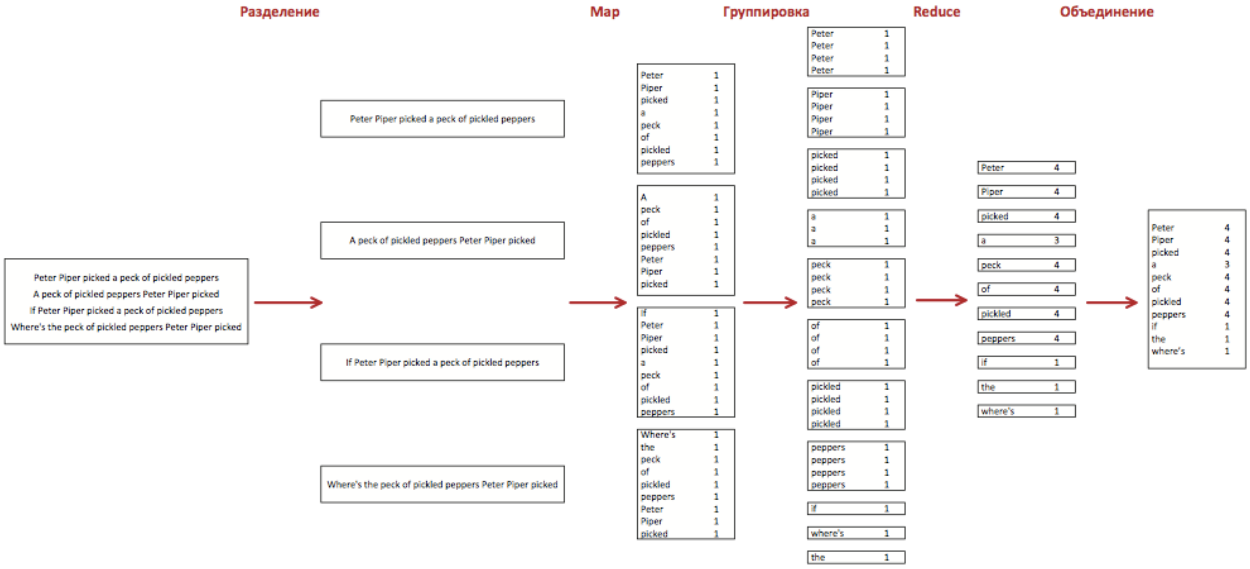

Рис. 3. Использование MapReduce для подсчета частоты встречаемости слов в наборе документов

Модель MapReduce, предоставляющая широкие возможности для распределенной обработки данных, внедрена в ряд БД.

И, наконец, к шестому виду графовых БД, использующих возможности модели распределенных вычислений для обработки больших объемов данных за один проход *MapReduce* относятся проекты: *Pregel*, *Apache Giraph*, *GraphLab*. Проект *Pregel* компании *Google* использует также идеи *BSP* обработки данных. Реализация графовых алгоритмов осуществляется в рамках последовательностей итераций, называемых «супершагами» *BSP*, в результате которых вычисляются в параллельном режиме некоторые, задаваемые пользователями функции вершин графа. Супершаг является неделимой единицей параллельных вычислений. В процессе исполнения вычислений, по мере готовности результатов формируются служебные сообщения, которыми обмениваются между собой функции вершин графов для обозначения смены своих состояний (активное, когда вычисления продолжаются и неактивное, когда вычисления выполнены) по автоматному принципу *vertex state machine* (автомат состояний вершины).

Проект *Apache Giraph*, является, по сути, аналогом проекта *Pregel*, попадающим под лицензию свободного программного обеспечения *Apache License* (*Apache Software Foundation*), который использует *Hadoop*, то есть модель *MapReduce* с использованием

распределенной файловой системы *HDFS* (*Hadoop Distributed File System*), а в качестве менеджера синхронизации состояний вершин применяется сервер синхронизации *Apache Zookeeper*.

GraphLab представляет собой набор программных инструментов для реализации высокопроизводительной параллельной распределенной обработки графовых данных и программирования собственных алгоритмов графовых вычислений с хранением результатов в файловой системе *HDFS*. Эта платформа имеет в своем составе также подсистему машинного обучения, предназначенную для извлечения знаний из графовых данных.

Заключение

Актуальность нереляционных баз данных растёт начиная с 2011 года. При этом популярность реляционных баз данных остается на том же уровне, во многом благодаря уже готовым реализациям проектов их использующих. Данную информацию можно подтвердить активным и растущим сообществом и постоянных обновлениях таких технологий, как Neo4j. MySQL при этом находится все на том же уровне и не вносит в свой продукт ничего значительно нового.

Глубокие связи особенно актуальны в различных социальных проектах, когда нам нужно получать друзей друзей, в задачах поиска маршрутов и т.п. Графовые базы данных призваны решить эти проблемы, когда наши данные могут быть удаленны друг от друга на два и более отношений. Они решаются очень элегантно, когда мы моделируем данные как «вершины графов», а связи как «ребра графа» между этими узлами. Мы можем делать обход графа с помощью давно известных и эффективных алгоритмов

Список литературы

1. Климанская Е.В. Современные платформы интеллектуальной аналитической обработки нформации: графовые базы данных: X Международная научно-практическая конференция «Наука вчера, сегодня, завтра»: материалы. (Новосибирск, 12 марта 2014 г.) Режим доступа: http://sibac.info/13762 (дата обращения 09.04.2015).
2. ОфициальныйсайтNeo4j.Режим доступа:http://www.neo4j.org (дата обращения 09.04.2015).
3. Официальный сайт HyperGraphDB. Режим доступа: http://www.hypergraphdb.org/ (дата обращения 14.04.2015)

4. Супрун Д.Е. Обзор постреляционных СУБД. AllegroGraph // Молодежный научно-технический вестник. Электрон. журн. № 1. 2014. Режим доступа: http://sntbul.bmstu.ru/doc/699509.html (дата обращения 14.04.2015).
5. Официальный сайт Sparsity Technologies - Powering Extreme Data. Режим доступа: http://sparsity-technologies.com (дата обращения 17.04.2015).
6. Козлов И.А. Анализ и классификация нереляционных баз данных // Молодежный научно-технический вестник. Электрон. журн. № 2. 2013. Режим доступа: http://sntbul.bmstu.ru/doc/552121.html (дата обращения 14.04.2015).
7. Самохвалов Э.Н., Ревунков Г.И., Гапанюк Ю.Е. Использование метаграфов для описания семантики и прагматики информационных систем // Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение. №1. 2015. Режим доступа: http://vestnikprib.bmstu.ru/catalog/icec/infth/673.html (дата обращения 09.04.2015).