Применение облачных вычислений для анализа данных большого объема в умных городах

[1] *Рензо Массобрио <renzom@fing.edu.uy>*
[1] *Серхио Несмачнов <sergion@fing.edu.uy>*
[2] *Андрей Черных <chernykh@cicese.mx>*
[3] *Арутюн Аветисян <arut@ispras.ru>*
[4] *Глеб Радченко <gleb.radchenko@susu.ru>*
[1] *Республиканский университет Уругвая, Монтевидео 11300, Уругвай*
[2] *Центр научных исследований и высшего образования, Энсенада, B.C. 22860, Мексика*
[3] *Институт системного программирования Российской академии наук, Москва, 109004, Россия*
[4] *Южно-Уральский государственный университет, Челябинск, 454080, Россия*

Аннотация. В этой статье рассматривается вопрос применения анализа данных большого объема с использованием облачных вычислений для решения задач анализа дорожного траффика в контексте «умных» городов. Предложенное решение базируется на модели параллельных вычислений MapReduce, реализованной на платформе Hadoop. Анализируются два экспериментальных случая: оценка качества общественного транспорта на основе анализа истории местоположения автобусов, и оценка мобильности пассажиров при помощи анализа истории покупок билетов с транспортных карт. Оба эксперимента используют реальную базу данных системы общественного транспорта Монтевидео в Уругвае. Результаты эксперимента показали, что рассмотренная модель действительно позволяет эффективно обрабатывать большие объемы данных.

Ключевые слова: облачные вычисления; big data; умные города; интеллектуальные транспортные системы, большие объемы данных.

DOI: 10.15514/ISPRAS-2016-28(6)-9

Для цитирования: Массобрио Р., Несмачнов С., Черных А., Аветисян А., Радченко Г. Применение облачных вычислений для анализа данных большого объема в умных

городах. Труды ИСП РАН, том 28, вып. 6, 2016 г., стр. 121-140. DOI: 10.15514/ISPRAS-2016-28(6)-9

1. Введение

Одной из задач умных городов является использование информационных и коммуникационных технологий для управления ресурсами городского хозяйства с целью поднятия качества услуг, предоставляемых горожанам. Использование этих технологий позволяет снизить инфраструктурные и эксплуатационные расходы, повысить эффективность использования ресурсов и способствовать улучшению взаимодействия жителей и администрации [1]. Чаще всего такие технологии применяются в области транспортного сервиса из-за его ключевой роли в жизни современного города.

Сегодня во многих развитых городах общественный транспорт решает одну из важнейших задач в формировании городской инфраструктуры, обеспечивая мобильность жителей [2]. Особенно большую роль общественный транспорт играет в густонаселенных районах. Однако, многие транспортные системы не соответствуют постоянно растущим требованиям. Чтобы решить эту проблему, правительство должно иметь глубокое понимание всей ситуации и деталей, включая ежедневную статистику использования транспорта [3]. Однако, ссылаясь на недостаток финансов и человеческих ресурсов, администрация обычно оперирует чрезвычайно ограниченным объемом частитчно устаревшей информации для принятия решений. Чаще всего, данные собираются, но не анализируются, что приводит к тому, что они используются для улучшения инфраструктуры общественного или личного транспорта в «сыром» виде. По этой причине, процесс принятия решений относительно транспортной ситуации обычно затягивается. Модель умного города позволяет анализировать данные из многих источников, чтобы понять транспортную ситуацию в городах.

Интеллектуальные транспортные системы (ИТС) играют ключевую роль в транспортных системах умных городов. ИТС – это системы, объединяющие в себе синергетические технологии, искуственный интеллект и инженерные принципы, применяемые к транспортным системам для увеличения пропускной способности, безопасности, эффективности и уменьшения воздействия на окружающую среду [4]. ИТС позволяют собирать данные о транспорте и способах передвижении в городах [5]. В больших городских зонах ИТС генерирует огромное количество информации, из которой необходимо извлечь полезную информацию о передвижении жителей.

В этой статье представлена платформа для эффективного анализа больших данных ИТС за счёт преимуществ облачных вычислений. Облачные вычисления включают в себя систему методов, которые позволяют использовать все преимущества множества вычислительных элементов для комплексного решения задач посредством распределенных вычислений. Когда исследователи сталкиваются со сложными задачами, такими как обработка

больших объёмов данных, распознавание образов, глубинное обучение, на помощь приходят распределенные вычисления с высокой производительностью, достигаемой кооперативным подходом. Такой эффект достигается разделением большой задачи на множество маленьких подзадач, решаемых параллельно на разных вычислительных элементах, для увеличения скорости обработки [6]. За последние десять лет были разработаны различные платформы для анализа больших данных с использованием распределенных вычислений на базе облачных систем [7].

В текущей работе мы адаптировали стандартную платформу для обработки данных большого объема, содержащихся в ИТС умного города. Данный эксперимент – это наш первый шаг к внедрению и увеличению эффективности обработки больших данных в рамках умных городов, что будет полезно и жителям, и администрациям городов.

Нами были проанализированы два экспериментальных случая: оценка качества общественного транспорта на основе анализа истории местоположения автобусов и оценка мобильности пассажиров при помощи анализа истории покупок билетов с транспортных карт. Оба эксперимента используют реальную базу данных системы общественного транспорта Монтевидео в Уругвае.

Статья организована следующим образом. В разделе 2 описываются основные концепции облачных вычислений, представляется парадигма MapReduce и платформа Hadoop. В разделе 3 представлен обзор смежных источников по вопросам, связанным с умными городами и обработки данных ИТС. Отдельный акцент сделан на распределенной и облачной обработке таких данных. В разделе 4 представлена модель облачной системы анализа данных ИТС. Раздел 5 посвящен обзору двух экспериментальных случаев и анализу полученных результатов. В разделе 6 представлены основные результаты исследования, определены основные направления дальнейшего развития работы.

2. Облачные вычисления, парадигма MapReduce и платформа Hadoop

В этом разделе описаны основные концепции облачных вычислений, модель MapReduce и ее реализация в рамках платформы Hadoop.

2.1 Распределенные вычисления и облачные вычисления

Распределенные вычисления – это обобщающее понятие для вычислительной модели и перечня программных алгоритмов, основанных на использовании множества вычислительных элементов, соединенных в сеть для решения задач, требующих больших ресурсов [8]. Распределенные процессы, выполняемые на базе данных вычислительных элементов, взаимодействуют, синхронизируют и координируют вычислительный процесс посредством

функциональной декомпозиции или же декомпозиции предметной области. За последние двадцать лет, распределенные вычисления были успешно внедрены в различных прикладных областях [6].

Облачные вычисления являются особым типом распределенных вычислений – они используют географически распределенные ресурсы и Интернет для выполнения запросов пользователей [9]. Суть облачных вычислений заключается в использовании сервисов по запросу, а пользователи могут получить к ним доступ с любого устройства из любой точки мира, используя общий пул реконфигурируемых ресурсов (таких как вычислительная мощность, память, сеть, приложения и др.).

Облачные вычисления очень часто используются и для работы с большими объемами данных. Одна из самых распространенных вычислительных моделей для обработки больших объемов данных – это модель MapReduce.

2.2 MapReduce

MapReduce – это парадигма программирования для обработки данных большого объема при помощи параллельных (или распределенных) алгоритмов на базе вычислительных кластеров, вычислительных сетей и облачных вычислительных систем.

Парадигма MapReduce состоит из двух простых операций. Первая операция, с помощью функции распределения (Map), выполняет (параллельно, на множестве вычислительных элементов) перечень задач, таких как фильтрация, сортировка и (или) непосредственно вычисления. Вторая операция - это сведение (Reduce) результатов выполнения функции распределения [10].

Вычисления, которые используют парадигму MapReduce, называют задачами MapReduce. Такая задача имеет два этапа решения: распределение и сведение. Первый этап выполняется в четыре шага: чтение данных, анализ распределения, комбинирование и непосредственное распределение. Результат этих процедур – список узлов и данных, которые обрабатываются на этих узлах. Второй этап состоит также из четырех шагов: перемещение, сортировка, сведение и вывод результата вычислений.

Библиотеки для парадигмы MapReduce написаны на множестве языков программирования. Самая распространенная - Apache Hadoop, описана ниже.

2.3 Hadoop

Сегодня Hadoop – это одна из самых популярных платформ, использующих модель MapReduce для обработки больших объемов данных. Hadoop – это распределенная вычислительная система, которая использует файловую систему с открытым исходным кодом Hadoop Distributed File System (HDFS) [11].

Hadoop предоставляет высокий уровень абстракции для описания задач. Это облегчает работу пользователям по реализации распределенных алгоритмов,

даже тем, у кого недостаточный уровень знаний в области распределенных вычислений. Такие задачи могут быть запущены на множестве вычислительных узлов, и обрабатывать большие объемы данных в реальном времени. Hadoop включает в себя экосистему, которая предоставляет дополнительные возможности для управления задачами, распределенного программирования, интерфейсы баз данных и другие функции.

2.4 Реализация парадигмы MapReduce на платформе Hadoop

Реализация парадигмы MapReduce на платформе Hadoop является самой распространенной и известной. Хотя MapReduce и проста для понимания, ее не всегда просто реализовать в виде алгоритмов для функций распределения и сведения.

В рамках платформы Hadoop, распределитель обычно принимает сырые данные в файловой системе HDFS. Данные, по умолчанию в текстовом формате, для распределителя представляют собой строки с ключом, равным байтовому смещению начала строки от начала файла. Задача MapReduce состоит из входных данных, кода программы и процедур для распределения. Пользователь может реализовать свои модули разделения, методы чтения записей, форматы входных данных и комбинаторы в зависимости от потребностей [7].

Вычислительный кластер в Hadoop состоит из корневого узла и подчиненных ему узлов. Корневой узел поддерживает несколько возможных ролей: менеджер задач, исполнитель, узел имен, и узел хранения. Подчиненные узлы могуть играть одну из двух ролей – узел хранения или исполнитель. Менеджер задач координирует все задачи системы с помощью списка задач исполнителей. Исполнитель запускает задачи и отправляет отчет о прогрессе работы менеджеру задач, который ведет статистику прогресса каждой задачи. Если задача завершается неправильно, менеджер задач может перезапустить ее на другом исполнителе. Именные узлы и узлы хранения относятся к кластеру HDFS.

Выполнение задачи MapReduce на платформе Hadoop выглядит следующим образом. В первую очередь, задача создается на клиентском узле, который запущен на виртуальной машине Java Virtual Machine (JVM). Далее, эта задача передает новую задачу менеджеру задач, который анализирует весь список запущенных и возвращает идентификатор новой задачи, после чегофайл, который необходимо выполнить и его кэш копируются на узлы. Далее, когда задача распределена, менеджер задач с помощью идентификатора инициализирует процесс обработки и подает входные данные. Исполнители возвращают менеджеру информацию о возможности запуска и доступной мощности. В зависимости от ответа менеджер задач назначает узлу выполнение или сведение. Узел-исполнитель извлекает ресурсы для обработки задачи и запускает новую JVM и выполняет функцию выполнения или сведения.

3. Обзор работ по смежным тематикам

Ряд статей, выпущенных в недавнее время, описывают применение распределенных и облачных вычислений для обработки больших данных, сгенерированных различными ИТС. В этом разделе мы проведем краткий обзор этих работ.

Преимущества анализа больших данных для систем общественного транспорта были представлены в работе [12]. Авторы проанализировали различные источники информации: траектория движения транспорта (координаты GPS, скорость передвижения), отчеты о неисправностях, передвижения людей (с помощью GPS и Wi-Fi сигналов), социальные сети (текстовые записи, адреса) и веб-логи (идентификаторы пользователей, комментарии). В статье описаны все преимущества и недостатки каждого источника информации. Также рассмотрено несколько новых идей для улучшения системы общественного транспорта с применением парадигмы ИТС, включая краудсорсинг для сбора и анализа актуальных данных о движении транспорта, сопровождение водителя и анализ поведения пассажиров. Кроме того, в статье представлен вывод о том, как внедрить технологию в систему общественного транспорта, чтобы она была совместима со следующими поколениями ИТС, которые повысят безопасность и эффективность поездок.

Методы интеллектуального вычисления недавно нашли применение при проектировании ИТС.

В работе [13] предложен метод последовательного поиска для прогнозирования ситуации на дорогах с использованием системы обнаружения транспортных средств (Vehicle Detection System) и алгоритма классификации k ближайших соседей (k nearest neighbors - kNN). Такое сочетание значительно превосходит традиционное использование алгоритма kNN, обеспечивая более точные результаты, сохраняя при этом высокую эффективность и стабильность.

В работе [14] представлено применение метода анализа данных на основе случайных лесов и Байесовского вывода для обработки больших объемов данных в системе микроволнового обнаружения транспорта (Microwave Vehicle Detection System). Главной целью работы является обнаружение факторов, провоцирующих аварии, в реальном времени. При анализе данных в час-пик, применяется модель надежности, учитывается увеличение объема и снижение средней скорости транспортного потока, индекс затора. Основной вывод анализа заключается в том, что пробки чаще всего являются основной причиной столкновений сзади.

В работе [15] представлен подход обучения с учителем, с использованием метода опорных векторов и Байесовской классификации, для построения системы прогноза транспортного потока в реальном времени. Сначала сырые данные проходят два этапа подготовки и фильтрацию шума. Это стандартный подход для подготовки данных в подобных исследованиях, и мы также будем

применять его в нашем эксперименте. Затем, модель транспортного потока исследуется Байесовской платформой. Методами регрессионного анализа моделируется пространственно-временная зависимость и отношения между дорогами. Эффективность такого метода исследуется на транспортных данных Кёнбусона, железной дороги Сеул-Пусан в Южной Корее. Результаты эксперимента показали, что подход с использованием метода опорных векторов в для оценки превосходит традиционные методы линейной регрессии с точки зрения точности.

В работе [16] была предложена модель для эффективного прогнозирования скорости движения на заданной местности. Она использует историю из различных источников, в том числе данные из различных ИТС, погодные условия и особые события, происходящие в городе. Для получения точных результатов модель прогнозирования должна часто обновляться, чтобы использовать самые последние данные. Модель прогнозирования сочетает в себе алгоритм классификации k ближайших соседей и регрессию на основе Гауссовского процесса. Кроме того, результаты рассчитываются с использованием модели MapReduce, реализованной на платформе Hadoop. Экспериментальная оценка была выполнена на основе реального сценария, используя данные, полученные на Research Data Exchange – платформе для публикации данных ИТС. Данные собраны на участке дороги I5N в Сан Диего, штат Калифорния, США. Информация содержит скорость и величину расхода топлива, полученные с помощью петлевого детектора на дороге, а также данные о видимости, полученные с ближайшей метеорологической станции. Результаты экспериментов показали, что предложенный метод может точно предсказать скорость движения потока со средней ошибкой менее двух миль в час. Кроме того, за счет использования платформы Hadoop в кластерной инфраструктуре, а не на одной вычислительной машине, было достигнуто уменьшение времени обработки на 69%.

В работе [17] представлены результаты исследований проблем краткосрочного прогнозирования транспортного потока в реальном времени. В решении использовался алгоритм k ближайших соседей в распределенной среде MapReduce на платформе Hadoop. Предлагаемое решение рассматривает пространственно-временную корреляцию в транспортном потоке, т.е. текущий поток на определенном участке дороги зависит от прошлого (временное измерение) и от потока на соседних участках дороги (пространственное измерение). В реализованном алгоритме, эти два фактора можно контролировать с помощью весов. Экспериментальный анализ проводился с использованием данных траекторий более 12 000 такси в Пекине, оборудованных GPS-датчиками, в 15-дневный период в ноябре 2012 года. Первые 14 дней данных использовались для обучения системы, а последний – для вычисления результатов. Предложенный алгоритм позволил уменьшить среднюю абсолютную ошибку от 8,5% до 11,5% в среднем в сравнении с тремя существующими методами, основанными на алгоритме k ближайших

соседей. Кроме того, предлагаемое решение в лучшем случае достигает вычислительную эффективность 0,84.

В работе [23] предложено использовать многокритериальные ячеистые генетические алгоритмы MOcell для оптимизации расписаний автобусного парка с автобусами различной вместимости.

В представленном обзоре, нами были определены наиболее распространенные методы анализа больших данных для обработки данных ИТС. Для определения моделей транспортного потока и генерации полезной информации для прогнозирования, применяются такие методы машинного обучения, как метод регрессии, k ближайших соседей и Байесовский вывод. Тем не менее, есть несколько работ, уделяющих особое внимание улучшению системы общественного транспорта, особенно с точки зрения пассажиров. В этом контексте, наша работа продолжает тему статей, предлагая конкретную модель для анализа данных ИТС в облаке, улучшающую систему общественного транспорта.

4. Предлагаемая модель

В этом разделе описана предполагаемая структура платформы распределенных облачных вычислений для обработки данных ИТС с целью улучшения систем общественного транспорта.

4.1 Архитектура системы

Задача разбивается на два этапа: 1) этап предварительных вычислений для подготовки входных данных к следующему шагу; 2) этап обработки данных ИТС с использованием распределенных облачных вычислений.

Для организации вычислений и определения структуры используется подход Master/Slave (ведущий/ведомый). На рис. 1 представлена концептуальная схема разрабатываемой платформы.

На этапе предварительных вычислений ведущий процесс подготавливает данные, фильтруя те записи, которые содержат ненужную информацию, а затем передает записи на этап распределенных вычислений. Процесс фильтрации может различаться в зависимости от поставленной задачи (см. раздел 5 "Практические примеры").

На этапе распределенных вычислений используется подход декомпозиции по данным. После этапа предварительных вычислений отфильтрованные и отсортированные данные разбиваются на части и передаются нескольким вычислительным элементам. Ведущий процесс является ответственным за распределение данных и назначение каждой части данных ведомым процессам для обработки. Каждый ведомый процесс получает от ведущего часть данных. Такая структура соответствует категории вычислительных систем Single Program Multiple Data (SPMD), согласно которой ведомые

процессы совместно выполняют одну и ту же задачу на разных блоках данных.

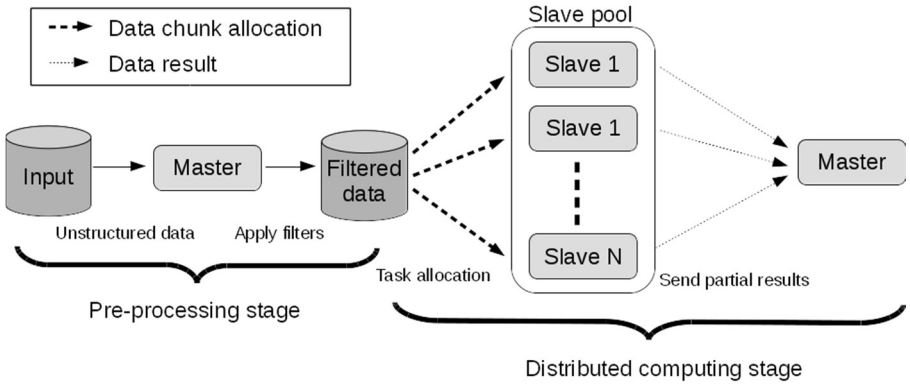

Рис. 1. Концептуальная схема разрабатываемой платформы

Fig. 1. Conceptual scheme of the developed platform

4.2 Реализация с применением MapReduce

При разработке распределенной вычислительной системы для анализа данных ИТС использовалась платформа Hadoop и была применена парадигма MapReduce. Система соответствует данной парадигме, так как между ведомыми процессами взаимодействий не происходит, а взаимодействия между ведущим и ведомыми процессами ограничены лишь распределением данных и сбором результатов вычислений. В Hadoop реализован стандартный подход MapReduce, в котором используется один главный узел и несколько рабочих узлов. Главный узел, используя процесс JobTracker, посылает задачи различным процессам TaskTracker, каждый из которых связан с определенным рабочим узлом. Как только все ведомые процессы заканчивают назначенные им задания, каждый процесс TaskTracker передает результаты обратно процессу JobTracker в главный узел.

В Hadoop реализован механизм обеспечения отказоустойчивости. Дополнительно, для улучшения работы механизма отказоустойчивости при работе с данными ИТС используются следующие особенности: 1) возможность отбрасывания поврежденных входных данных в случае наличия поврежденной информации в записи; 2) встроенный механизм репликации HDFS для хранения данных в различных вычислительных узлах.

5. Практические примеры

В этом разделе описывается применение разработанной платформы для обработки двух разных типов данных ИТС. В обоих случаях

экспериментальная оценка проводится с использованием соответствующего набора реальных данных ИТС г. Монтевидео, Уругвай.

5.1 Метрики качества обслуживания транспортной системы с использованием данных о местоположении автобусов

В этом практическом примере рассматривается задача вычисления метрик качества обслуживания системы общественного транспорта с использованием данных GPS-навигаторов, установленных в автобусах [24]. Данные содержат информацию о местоположении каждого автобуса в конкретный момент времени. Эта информация обновляется каждые 10-30 секунд. Для определения эффективности системы общественного транспорта основной задачей является введение соответствующих метрик, таких как: 1) реальное время, затрачиваемое каждым автобусом на путь между заранее заданными и отмеченными местами в городе; 2) статистическая информация о задержках каждого автобуса на конкретном маршруте для определения перегруженных мест. Данные должны быть должным образом организованы для возможности определения закономерностей объема потока пассажиров и загруженности дорог в различные дни недели, а также в различное время дня.

Существует как минимум две целевые группы, которым разработанная система принесет пользу: пассажиры и городская администрация. С помощью информации, которая получена на основе обработки исторических данных и данных реального времени, пользователь системы общественного транспорта сможет принять более выгодные решения о собственном перемещении (например, выбрать определенный автобусный маршрут, совершить пересадку). Эта информация может предоставляться посредством интеллектуального мобильного приложения или сайта. Для городской администрации такая информация полезна для планирования долгосрочных изменений автобусных маршрутов, расписания движения, положения автобусных остановок, а также для выявления конкретных проблемных ситуаций.

Схема разработанной системы представлена на рис. 2. Автобусы посылают данные о местоположении на облачный сервер. На сервере происходит MapReduce-обработка собранных данных GPS различных автобусов в реальном времени. Результаты вычислений передаются в мобильное приложения для использования конечными пользователями и мониторинговое приложение для использования городской администрацией.

На этапе предварительных вычислений, ведущий процесс подготавливает данные, фильтруя те записи, которые хранят ненужную информацию (например, поврежденные данные GPS). Дополнительно данные фильтруются таким образом, чтобы включать только записи из временного промежутка, задаваемого пользователем. В конце этапа записи сортируются по идентификационному номеру автобуса, что позволяет увеличить эффективность работы следующего этапа.

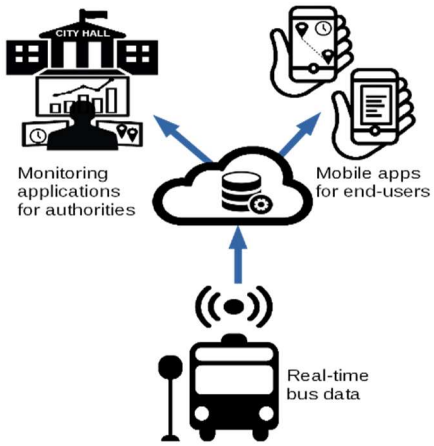

Рис. 2. Архитектура системы облачной обработки данных ИТС

Fig. 2. Architecture of the ITS cloud data processing system

На этапе распределенных вычислений, ведущий процесс разделяет набор отфильтрованных записей GPS, полученных на предыдущем этапе, на части и распределяет их между ведомыми процессами. Каждая часть включает записи о местоположении и времени конкретного автобусного маршрута. Таким образом, обеспечивается независимость ведомых процессов, что увеличивает вычислительную производительность. Каждый ведомый процесс вычисляет время, затраченное на путь между каждыми отмеченными местами на маршруте. На шаге свертки, результаты вычислений используются для определения статистики качества обслуживания, которая будет предоставлена пользователю. Результаты сортируются по маршрутам автобусов и по отмеченным местам. В конце этапа, каждый ведомый процесс возвращает полученные результаты ведущему процессу, который должен сгруппировать результаты и вывести итоговый результат пользователю.

Экспериментальная оценка производится на основе облачной инфраструктуры Cluster FING, предоставленной Республиканским университетом Уругвая [18], для вычислений используются 24-ядерные процессоры AMD Opteron 6172 Magny Cours с тактовой частотой 2,26 ГГц, 24 Гб оперативной памяти и операционная система CentOS Linux 5.2.

Для экспериментального анализа используются реальные исторические данные ИТС, предоставленные правительством г. Монтевидео, Уругвай. Автобусные компании Монтевидео обязаны посылать данные о местоположении автобусов и о продажах билетов правительству города. Автобусная сеть Монтевидео достаточно сложна и насчитывает 1383 автобусных маршрута и 4718 автобусных остановок.

В этом практическом примере рассматривается полный набор данных о продажах билетов и местоположении автобусов за 2015 год, в котором содержится около 200 Гб данных. Данные о местоположении автобуса содержат информацию о месте нахождения каждого автобуса с интервалами от 10 до 30 секунд.

Полный набор данных разделяется на части, которые используются для определения различных сценариев, представляющихся для оценки производительности разработанной системы с различными размерами входных файлов, различными временными промежутками и различным количеством процессов предварительной обработки (Map) и процессов распределенных вычислений (Reduce).

Таблица 1. Результаты замеров производительности

Table 1. Performance measurement results.

#I	#D	#M	#R	T_1(s)	T_N(s)	S_N	E_N
10	3	14	8	1333.9	253.1	5.27	0.22
10	3	22	22	1333.9	143	9.33	0.39
10	30	14	8	2108.6	178	11.84	0.49
10	30	22	22	2108.6	187.3	11.26	0.47
20	3	14	8	2449	351.1	6.98	0.29
20	3	22	22	2449	189.8	12.9	0.54
20	30	14	8	3324.5	275.6	12.06	0.5
20	30	22	22	3324.5	238.8	13.92	0.58
20	60	14	8	4762	300.8	15.83	0.66
20	60	22	22	4762	264.7	17.99	0.75
30	3	14	8	3588.5	546.9	6.56	0.27
30	3	22	22	3588.5	179.6	19.99	0.83
30	30	14	8	5052.9	359.6	14.05	0.59

#I	#D	#M	#R	T_1(s)	T_N(s)	S_N	E_N
30	30	22	22	5052.9	281.1	17.98	0.75
30	60	14	8	5927.9	383.4	15.46	0.64
30	60	22	22	5927.9	311.4	19.04	0.79
30	90	14	8	7536.9	416.6	18.09	0.75
30	90	22	22	7536.9	349.2	21.58	0.9
60	3	14	8	7249.6	944	7.68	0.32
60	3	22	22	7249.6	362.1	20.02	0.83
60	60	14	8	10037.1	672.6	14.92	0.62
60	60	22	22	10037.1	531.4	18.89	0.79
60	90	14	8	11941.6	709.6	16.83	0.7
60	90	22	22	11941.6	648.9	18.4	0.77
60	180	14	8	19060.8	913.7	20.86	0.87
60	180	22	22	19060.8	860.3	22.16	0.92

Работа происходит на наборах данных ИТС объемом 10, 20, 30 и 60 Гб и на временных промежутках длительностью 3 дня, а также 1, 2, 3 и 6 месяцев. Для оценки производительности параллельного алгоритма используются стандартные метрики ускорения и эффективности.

В таблице 1 представлены результаты замеров вычислительной производительности системы при использовании различных сценариев и следующих изменяемых параметров: размер входных данных в Гб (#I), временной промежуток в днях (#D), количество процессов Map (#M) и Reduce (#R). Для каждого сценария определялись такие результаты, как время выполнения в секундах с использованием 1 ядра (T1) и 24 ядер (TN),

ускорение (SN) и эффективность (EN). В таблице приведены средние значения после пяти независимых запусков каждого сценария.

Результаты в таблице 1 демонстрируют значительное увеличение производительности при использовании выбранного подхода по сравнению с последовательной реализацией, особенно на больших наборах данных. Ускорение при использовании 24 ядер достигает значения 22.16, что соответствует значению эффективности 0.92. Распределенная реализация позволяет сократить время работы на больших объемах данных с 6 часов до 14 минут. Такое увеличение производительности просто необходимо для быстрого реагирования городской администрации на возникновение непредвиденных ситуаций и для анализа различных метрик и сценариев, как обычными пользователями, так и администраторами системы.

Использование 22 процессов Map и 22 процессов Reduce позволяет достигнуть наибольшей эффективности. Время выполнения возрастает на 15% в лучшем случае и на 9% в среднем по сравнению со сценариями, в которых используются 14 процессов Map и 8 процессов Reduce. При работе с малыми объемами данных процессы низко нагружены с вычислительной и пространственной точек зрения, и заметного выигрыша во времени выполнения по сравнению с последовательной реализацией нет.

5.2 Построение матриц отправления-прибытия с использованием исторических данных смарт-карт

Для решения задач оптимизации систем городского транспорта необходимо понимать шаблоны передвижения и распределения горожан. Обычно эта информация представляется в виде следующих матриц: 1) матриц отправления прибытия (Origin-Destination - OD-матриц), которые отражают количество людей, передвигающихся из конкретной точки города в определенный временной промежуток [19]; 2) матриц распределения, которые отражают количество автобусных билетов, проданных на каждой остановке в городе.

Обычно эти матрицы строятся на основе опросов пассажиров и водителей. Однако такой подход не дает полного видения ситуации, не предоставляет актуальную информацию и требует больших вложений от городского правительства.

В этом практическом примере предлагается иной подход для построения матриц распределения и OD-матриц [25]. Мы вычисляем и обновляем их, используя обработку таких данных ИТС, как количество билетов, проданных с использованием смарт-карт и без их использования, а также данных о местоположении автобусов. Учитывая высокую вычислительную сложность обработки больших объемов данных ИТС, нами будет применяться модель распределенных вычислений, описанная в разделе 4.

Основной проблемой при генерации матриц распределения и OD-матриц с использованием данных о продажах билетов является то, что пассажиры

применяют смарт-карту только при посадке и не используют ее при выходе из автобуса. Следовательно, хоть начальный пункт каждой поездки известен, необходимо определить и конечный пункт. Более того, пассажиры, у которых есть смарт-карта, не обязаны применять ее для оплаты билета, они могут расплатиться наличными. Следовательно, записи о продажах билетов не хранят полную информацию, связанную с конкретным пассажиром, поэтому нельзя отследить несколько поездок одного и того же пассажира.

Модель для построения OD-матриц основана на восстановлении последовательности поездок для пассажиров, использующих смарт-карту. Подобный подход описан в соответствующей литературе [20, 21, 22]. Мы предполагаем, что у каждой смарт-карты есть только один пользователь. Рассматриваемый подход основан на обработке каждой поездки, получении данных о начальном пункте поездки и точном/приближенном определении конечного пункта поездки. Для приближенного определения вводятся следующие понятия: поездка с пересадками и прямая поездка. Основные детали вводимых понятий описаны далее.

Поездка с пересадками. В такой поездке пассажир при посадке в первый автобус оплачивает поездку смарт-картой, которая имеет уникальный идентификационный номер. Далее пассажир за ограниченный промежуток времени, указанный в билете, может сделать одну и более пересадок без покупки нового билета, а просто предоставив свою смарт-карту.

Такой подход позволяет отследить, хранит ли запись информацию о новой поездке (т.е. о покупке нового билета) или же она хранит информацию о пересадке (т.е. о подтверждении смарт-карты). Мы предполагаем, что пассажиры не делают длительных пеших перемещений во время поездок и что пассажир, совершающий пересадку, выходит на остановке, которая находится ближе всего к той остановке, на которой он сядет на следующий автобус. Посадка на автобус после пересадки записывается в систему. Приблизительное место пересадки с одного автобуса на другой определяется как ближайшая автобусная остановка конкретного маршрута.

Прямая поездка. В такой поездке не происходит пересадок. Дополнительно, последний этап поездки с пересадками рассматривается в качестве прямой поездки. В обоих случаях сложность состоит в точном определении пункта назначения. Для определения пунктов назначения мы вводим два предположения, которые часто используются в соответствующей литературе: 1) пассажиры начинают новую поездку на автобусной остановке, находящейся ближе всего к пункту назначения предыдущей поездки; 2) в конце дня пассажиры возвращаются на автобусную остановку, которая была пунктом отправки первой поездки текущего дня. Для определения пунктов назначения мы пытаемся построить "цепь" поездок каждого пассажира в определенный день. Для этого просматриваются все поездки, совершенные каждым пассажиром за 24 часа. Для каждой новой поездки мы пытаемся определить точку высадки, для этого ищется автобусная остановка, находящаяся в заранее

заданном радиусе от остановки предыдущей поездки. Также хранится лог поездок, которые не могут быть выстроены в "цепь" для определения эффективности метода.

В этом практическом примере на этапе предварительных вычислений, описанном в разделе 4, фильтруются те данные о продажах, которые несут неточную информацию. Мы предполагаем, что измерение GPS неточно, если положение автобуса удалено более чем на 50 метров от его маршрута. Отфильтрованные данные разбиваются на части по идентификационным номерам смарт-карт и передаются ведомым процессам. Таким образом, ведомый процесс обрабатывает информацию обо всех поездках определенного пассажира, поэтому взаимодействие между ведомыми процессами отсутствует. Для улучшения производительности в конце этапа предварительных вычислений записи сортируются по дате для последовательной передачи ведомым процессам.

Для экспериментального анализа использовался полный набор данных ИТС за январь 2015 года, который включал данные о продажах билетов и местоположении автобусов. В наборе данных находилась информация о более чем 500 тысячах смарт-карт (что соответствует более чем 13 миллионам поездок).

В описанной параллельной модели ведущий/ведомый необходимо выбрать объем задач, назначаемый каждому ведомому процессу. Подходящий объем задач предоставляет хорошее распределение нагрузки и уменьшение количества ненужных связей между ведущим и ведомыми процессами. Были проведены пять независимых вычислений с различным объемом задач и использованием различного числа ядер.

Полученные экспериментальные результаты показывают, что применение модели распределенных вычислений позволяет значительно улучшить эффективность обработки данных по сравнению с последовательной реализацией. Для прямых поездок с использованием объема задач в 5000 поездок и 24 ядер было получено значение ускорения 16.41. Результаты подтверждают улучшение времени работы с использованием распределенной вычислительной системы и нескольких вычислительных узлов.

Также результаты показывают, что размер объема задач значительно влияет на общее время работы алгоритма, при меньших размерах достигаются лучшие результаты. Дальнейшие эксперименты должны быть направлены на установление краевых значений, после которых затраты на связи между ведущими и ведомыми процессами негативно сказываются на времени выполнения.

6. Заключение и дальнейшие исследования

В этой работе была разработана и реализована распределенная вычислительная система для облачной обработки больших объемов данных

ИТС с использованием парадигмы MapReduce на платформе Hadoop. Выполнен обзор литературы с обсуждением предыдущих попыток использования распределенных вычислений для обработки данных ИТС в контексте умных городов. Эффективность разработанной модели отражена и с использованием двух практических примеров: 1) вычисление метрик качества обслуживания системы общественного транспорта с использованием данных о местоположении автобусов; 2) построение OD-матриц на основе данных о продажах билетов. В обоих случаях распределенная модель позволяет значительно уменьшить время обработки больших объемов исторических данных. Экспериментальный анализ в обоих случаях был произведен с использованием реальных данных ИТС г. Монтевидео, Уругвай.

Возможные направления дальнейших исследований: 1) использование различных источников данных ИТС; 2) разработка приложения для горожан с интуитивно понятным доступом к полученной информации; 3) применение полученной информации для решения оптимизационных задач, например, прокладка автобусных маршрутов, изменение мест автобусных остановок, расписание автобусов и т.д.

Благодарности. Работа выполнена при поддержке Правительства Российской Федерации, Акт 211, контракт № 02.A03.21.0011 и CONACYT (Consejo Nacional de Ciencia y Tecnología, México), грант номер 178415.

Список литературы

[1]. Deakin, M., & Al Waer, H. (2011). From intelligent to smart cities. Intelligent Buildings International, 3(3), 140-152.
[2]. Grava, S. (2003). Urban transportation systems. Choices for communities.
[3]. Chen, C., Ma, J., Susilo, Y., Liu, Y., Wang, M.: (2016). The promises of big data and small data for travel behavior (aka human mobility) analysis. Transportation Research Part C: Emerging Technologies 68, 285–299.
[4]. Sussman, J. S. (2008). Perspectives on intelligent transportation systems (ITS). Springer Science & Business Media.
[5]. Figueiredo, L., Jesus, I., Machado, J. T., Ferreira, J., & de Carvalho, J. M. (2001). Towards the development of intelligent transportation systems. In Intelligent transportation systems (Vol. 88, pp. 1206-1211).
[6]. Foster I. (1995). *Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
[7]. White T. (2009). *Hadoop: The Definitive Guide* (1st ed.). O'Reilly Media, Inc..
[8]. Attiya H. & Welch J. (2004). *Distributed Computing: Fundamentals, Simulations and Advanced Topics*. John Wiley & Sons.
[9]. Buyya R., Broberg J., & Goscinski. A. M. (2011). *Cloud Computing Principles and Paradigms*. Wiley Publishing.
[10]. Dean J. & Ghemawat S. (2008). MapReduce: simplified data processing on large clusters. *Commun. ACM* 51, 1 (January 2008), 107-113.

[11]. Shafer, J., Rixner, S., & Cox, A. L. (2010). The hadoop distributed filesystem: Balancing portability and performance. In *IEEE International Symposium on Performance Analysis of Systems & Software* (pp. 122-133).
[12]. Zheng, X., Chen, W., Wang, P., Shen, D., Chen, S., Wang, X., ... & Yang, L. (2016). Big data for social transportation. IEEE Transactions on Intelligent Transportation Systems, 17(3), 620-630.
[13]. Oh, S., Byon, Y. J., & Yeo, H. (2016). Improvement of Search Strategy With K-Nearest Neighbors Approach for Traffic State Prediction. IEEE Transactions on Intelligent Transportation Systems, 17(4), 1146-1156.
[14]. Shi, Q., & Abdel-Aty, M. (2015). Big data applications in real-time traffic operation and safety monitoring and improvement on urban expressways. Transportation Research Part C: Emerging Technologies, 58, 380-394.
[15]. Ahn, J., Ko, E., & Kim, E. Y. (2016). Highway traffic flow prediction using support vector regression and Bayesian classifier. In 2016 International Conference on Big Data and Smart Computing (BigComp) (pp. 239-244). IEEE.
[16]. Chen, X. Y., Pao, H. K., & Lee, Y. J. (2014). Efficient traffic speed forecasting based on massive heterogenous historical data. In Big Data (Big Data), 2014 IEEE International Conference on (pp. 10-17). IEEE.
[17]. Xia, D., Wang, B., Li, H., Li, Y., & Zhang, Z. (2016). A distributed spatial–temporal weighted model on MapReduce for short-term traffic flow forecasting. Neurocomputing, 179, 246-263.
[18]. Nesmachnow S. (2010). Computación científica de alto desempeño en la Facultad de Ingeniería, Universidad de la República. Revista de la Asociación de Ingenieros del Uruguay 61 (1), 12-15.
[19]. Yang H., Sasaki T., Iida Y., Asakura Y. (1992). Estimation of origin-destination matrices from link traffic counts on congested networks, Transportation Research Part B: Methodological, Volume 26, Issue 6, Pages 417-434.
[20]. Trépanier, M., Tranchant, N., & Chapleau, R. (2007). Individual trip destination estimation in a transit smart card automated fare collection system. *Journal of Intelligent Transportation Systems*, *11*(1), 1-14.
[21]. Wang, W., Attanucci, J. P., & Wilson, N. H. (2011). Bus passenger origin-destination estimation and related analyses using automated data collection systems. *Journal of Public Transportation*, *14*(4), 7.
[22]. Munizaga, M. A., & Palma, C. (2012). Estimation of a disaggregate multimodal public transport Origin–Destination matrix from passive smartcard data from Santiago, Chile. *Transportation Research Part C: Emerging Technologies*, *24*, 9-18.
[23]. Peña D., Tchernykh A., Nesmachnow S., Massobrio S., Drozdov A. Y., Garichev S. N. (2016). Multiobjective vehicle type and size scheduling problem in urban public transport using MOCell. IEEE International conference Engineering & Telecommunications, Moscow, Russia.
[24]. R. Massobrio, A. Pías, N. Vázquez, & S. Nesmachnow (2016). Map-Reduce for Processing GPS Data from Public Transport in Montevideo, Uruguay. In 2do Simposio Argentino de Grandes Datos.
[25]. E. Fabbiani, P. Vidal, R. Massobrio, & S. Nesmachnow (2016). Distributed Big Data analysis for mobility estimation in Intelligent Transportation Systems. In Latin American High Performance Computing Conference.

Towards a Cloud Computing Paradigm for Big Data Analysis in Smart Cities

[1] *Renzo Massobrio <renzom@fing.edu.uy>*
[1] *Sergio Nesmachnow <sergion@fing.edu.uy>*
[2] *Andrei Tchernykh <chernykh@cicese.mx>*
[3] *Arutyun Avetisyan <arut@ispras.ru>*
[4] *Gleb Radchenko <gleb.radchenko@susu.ru>*
[1] *Universidad de la República, Montevideo 11300, Uruguay.*
[2] *CICESE Research Center, Ensenada , B.C. 22860, México*
[3] *Institute for System Programming of the RAS, Moscow, 109004, Russia*
[4] *South Ural State University, Chelyabinsk, 454080, Russia.*

Abstract. In this paper, we present a Big Data analysis paradigm related to smart cities using cloud computing infrastructures. The proposed architecture follows the MapReduce parallel model implemented using the Hadoop framework. We analyse two case studies: a quality-of-service assessment of public transportation system using historical bus location data, and a passenger-mobility estimation using ticket sales data from smartcards. Both case studies use real data from the transportation system of Montevideo, Uruguay. The experimental evaluation demonstrates that the proposed model allows processing large volumes of data efficiently.

Keywords: cloud computing; big data; smart cities; intelligent transportation systems.

DOI: 10.15514/ISPRAS-2016-28(6)-9

For citation: Massobrio R., Nesmachnow S., Tchernykh A., Avetisyan A., Radchenko G. Towards a Cloud Computing Paradigm for Big Data Analysis in Smart Cities. *Trudy ISP RAN/Proc.ISP RAS*, vol. 28, issue 6, 2016. pp. 121-140 (in Russian). DOI: 10.15514/ISPRAS-2016-28(6)-9

Acknowledgment. This work is partially supported by Government of the Russian Federation, Act 211, contract № 02.A03.21.0011, and CONACYT (Consejo Nacional de Ciencia y Tecnología, México), grant no. 178415. Datasets used in this paper are from Intendencia de Montevideo.

References

[1]. Deakin, M., & Al Waer, H. (2011). From intelligent to smart cities. Intelligent Buildings International, 3(3), 140-152.
[2]. Grava, S. (2003). Urban transportation systems. Choices for communities.
[3]. Chen, C., Ma, J., Susilo, Y., Liu, Y., Wang, M.: (2016). The promises of big data and small data for travel behavior (aka human mobility) analysis. Transportation Research Part C: Emerging Technologies 68, 285–299.

[4]. Sussman, J. S. (2008). Perspectives on intelligent transportation systems (ITS). Springer Science & Business Media.
[5]. Figueiredo, L., Jesus, I., Machado, J. T., Ferreira, J., & de Carvalho, J. M. (2001). Towards the development of intelligent transportation systems. In Intelligent transportation systems (Vol. 88, pp. 1206-1211).
[6]. Foster I. (1995). *Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
[7]. White T. (2009). *Hadoop: The Definitive Guide* (1st ed.). O'Reilly Media, Inc..
[8]. Attiya H. & Welch J. (2004). *Distributed Computing: Fundamentals, Simulations and Advanced Topics*. John Wiley & Sons.
[9]. Buyya R., Broberg J., & Goscinski. A. M. (2011). *Cloud Computing Principles and Paradigms*. Wiley Publishing.
[10]. Dean J. & Ghemawat S. (2008). MapReduce: simplified data processing on large clusters. *Commun. ACM* 51, 1 (January 2008), 107-113.
[11]. Shafer, J., Rixner, S., & Cox, A. L. (2010). The hadoop distributed filesystem: Balancing portability and performance. In *IEEE International Symposium on Performance Analysis of Systems & Software* (pp. 122-133).
[12]. Zheng, X., Chen, W., Wang, P., Shen, D., Chen, S., Wang, X., ... & Yang, L. (2016). Big data for social transportation. IEEE Transactions on Intelligent Transportation Systems, 17(3), 620-630.
[13]. Oh, S., Byon, Y. J., & Yeo, H. (2016). Improvement of Search Strategy With K-Nearest Neighbors Approach for Traffic State Prediction. IEEE Transactions on Intelligent Transportation Systems, 17(4), 1146-1156.
[14]. Shi, Q., & Abdel-Aty, M. (2015). Big data applications in real-time traffic operation and safety monitoring and improvement on urban expressways. Transportation Research Part C: Emerging Technologies, 58, 380-394.
[15]. Ahn, J., Ko, E., & Kim, E. Y. (2016). Highway traffic flow prediction using support vector regression and Bayesian classifier. In 2016 International Conference on Big Data and Smart Computing (BigComp) (pp. 239-244). IEEE.
[16]. Chen, X. Y., Pao, H. K., & Lee, Y. J. (2014). Efficient traffic speed forecasting based on massive heterogenous historical data. In Big Data (Big Data), 2014 IEEE International Conference on (pp. 10-17). IEEE.
[17]. Xia, D., Wang, B., Li, H., Li, Y., & Zhang, Z. (2016). A distributed spatial–temporal weighted model on MapReduce for short-term traffic flow forecasting. Neurocomputing, 179, 246-263.
[18]. Nesmachnow S. (2010). Computación científica de alto desempeño en la Facultad de Ingeniería, Universidad de la República. Revista de la Asociación de Ingenieros del Uruguay 61 (1), 12-15.
[19]. Yang H., Sasaki T., Iida Y., Asakura Y. (1992). Estimation of origin-destination matrices from link traffic counts on congested networks, Transportation Research Part B: Methodological, Volume 26, Issue 6, Pages 417-434.
[20]. Trépanier, M., Tranchant, N., & Chapleau, R. (2007). Individual trip destination estimation in a transit smart card automated fare collection system. *Journal of Intelligent Transportation Systems*, *11*(1), 1-14.
[21]. Wang, W., Attanucci, J. P., & Wilson, N. H. (2011). Bus passenger origin-destination estimation and related analyses using automated data collection systems. *Journal of Public Transportation*, *14*(4), 7.

[22]. Munizaga, M. A., & Palma, C. (2012). Estimation of a disaggregate multimodal public transport Origin–Destination matrix from passive smartcard data from Santiago, Chile. *Transportation Research Part C: Emerging Technologies*, *24*, 9-18.
[23]. Peña D., Tchernykh A., Nesmachnow S., Massobrio S., Drozdov A. Y., Garichev S. N. (2016). Multiobjective vehicle type and size scheduling problem in urban public transport using MOCell. IEEE International conference Engineering & Telecommunications, Moscow, Russia.
[24]. R. Massobrio, A. Pías, N. Vázquez, & S. Nesmachnow (2016). Map-Reduce for Processing GPS Data from Public Transport in Montevideo, Uruguay. In 2do Simposio Argentino de Grandes Datos.
[25]. E. Fabbiani, P. Vidal, R. Massobrio, & S. Nesmachnow (2016). Distributed Big Data analysis for mobility estimation in Intelligent Transportation Systems. In Latin American High Performance Computing Conference.