УДК 004.67 + 004.93'14
ББК 32.96

О КЛАСТЕРИЗАЦИИ ГЕОРАСПРЕДЕЛЕННЫХ ДАННЫХ БОЛЬШОГО ОБЪЕМА[1]

Чечеткин И.А.[2], **Щербаков М.В.**[3], **Садовникова Н.П.**[4], **Парыгин Д.С.**[5] **Голубев А.В.**[6]

(ФГБОУ ВО «Волгоградский государственный технический университет», Волгоград)

В статье рассматривается проблема обработки георасределенных данных большого объема с целью автоматической группировки данных в соответствии с географической близостью (кластеризация). Рассмотрены различные алгоритмы кластеризации и проанализированы возможности их применения к данным рассматриваемого типа. Предложен подход к кластеризации георасределенных данных, основанный на базовом алгоритме k-средних и вычислении реального расстояния между точками с учетом городского рельефа. Рассмотрены различные способы реализации, в том числе и распараллеливание выполнения. Представлены различные тестовые примеры, объясняющие суть предлагаемого подхода.

Ключевые слова: георасределенные данные, построение маршрутов, кластеризация.

---

[1] *Работа выполнена при финансовой поддержке РФФИ, гранты № 16-37-60066_мол_а_дк, 15-47-02613_р_поволжье_а, проекта МД-6964.2016.9*

[2] *Илья Александрович Чечеткин, студент, (illaech@gmail.com).*

[3] *Максим Владимирович Щербаков, доктор технических наук (Волгоград, пр. Ленина, д. 28, тел. (8442) 24-88-00, maxim.shcherbakov@vstu.ru)*

[4] *Наталья Петровна Садовникова, доктор технических наук, профессор (Волгоград, пр. Ленина, д. 28, тел. (8442) 24-88-00).*

[5] *Данила Сергеевич Парыгин, кандидат технических наук, старший преподаватель (dparygin@gmail.com).*

[6] *Алексей Владимирович Голубев, студент, (ax.golubev@gmail.com)*

Введение

В настоящее время повсеместный сбор данных и технологии обработки данных, которые становятся все более доступными, открывают новые возможности для разработки систем и сервисов, использующих георасределенные данные. Георасределенные данные — это данные о событиях, которые в качестве основных атрибутов и атрибутов временной метки имеют две географические координаты [20].

Большая часть населения, живущего в городах, имеют мобильные телефоны или смартфоны. Операторы мобильной связи могут получать большой объем данных о перемещении владельцев телефонов по городу. Кроме этого, пользователи смартфонов добровольно делятся информацией о своем перемещении, размещая информацию в социальных сетях и геоинформационных сервисах. Исходя из предположения, что каждый человек имеет свои определенные «шаблоны перемещения» — набор часто посещаемых мест, которые можно рассматривать как личные предпочтения по перемещению в городе, эти данные можно использовать для оценки текущей инфраструктуры города и модификации на основе реальных перемещений жителей, предоставления услуг, реализации новых геоинформационных сервисов. Можно сформулировать два принципа построения подобных сервисов:

- формирование сервисов по принципу стандартной цифоровой услуги: обращение к сервису гарантирует получение (цифровой) услуги в гарантированное время вне зависимости от инструмента выполнения (человек, человек и системы поддержки принятия решений, автоматические системы);
- принцип проактивных систем: перенос человека принимающего решения на верхний (супервизорный) уровень и исключение его из контура оперативного управления без потери эффективности управления [43].

Построение новых геоинформационных сервисов, как правило, реализуется в соответствии со следующей схемой: сбор

данных, предварительная обработка данных, создание алгоритма обработки гео-распределенных данных и предоставление информации в соответствии с (i) предпочтениями человека и (ii) с заданными критериями качества.

Поскольку получение данных происходит от каждого отдельного человека, то результирующие выборки получаются больших размеров. Например, даже при указании одной пары точек отправления - назначения, число собранных данных в два раза превышает число респондентов. В этом случае, возникает проблема обнаружения геометрических (географических) скоплений точек. Например, для выявления остановочных пунктов общественного транспорта необходимо выявить центры скоплений пунктов отправления и назначения. Это выделяется в отдельную задачу – задачу кластеризации геораспределенных данных. Поскольку элементы данных являются географическими объектами, то при кластеризации необходимо учитывать естественные и искусственные препятствия — дороги, магистрали, парки, реки, железные дороги т.д.

В статье представлено решение задачи кластеризации геораспределенных данных, основанное на попытках решения задачи анализа и модификации сети общественного транспорта [2, 4]. Для упрощения восприятия информации, задачу кластеризации геораспределенных объектов рассмотрим в рамках решения задачи совершенствования транспортной сети небольшого города (300 000 жителей) [32, 41].

1. Постановка задачи

Задача кластеризации заключается в разбиении множества объектов на непересекающиеся подмножества — кластеры — таким образом, чтобы каждое подмножество состояло из схожих объектов, а объекты разных кластеров существенно отличались, при этом структура кластеров изначально неизвестна. Для определения схожести объектов необходимо задавать функцию расстояния на множестве объектов.

Сформулируем постановку задачи кластеризации геораспре-

деленных данных.

Имеется выборка элементов $X = \{x_1, \ldots, x_n\}$, полученная из картографических сервисов, функция расстояния между элементами $\rho(x, x')$, и k — ожидаемое число узлов (остановочных пунктов) во входной выборке. Требуется разбить выборку на $C = \{c_1, \ldots, c_k\} - k$ непересекающихся подмножеств элементов, называемых кластерами, таким образом, чтобы каждый кластер состоял из объектов, близких по метрике ρ, а объекты разных кластеров существенно различались. При этом метрика ρ должна оценивать географическую близость элементов, учитывая рельеф городской местности.

Задача является типичной задачей машинного обучения без учителя, но имеет некоторые специфические особенности. Во-первых, стандартные применяемые метрики для решения данной задачи не подходят: необходимо учитывать особенности местности, такие как реки, овраги и балки, автомобильные и железные дороги, здания и промышленные зоны. Существуют алгоритмы, учитывающие препятствий [16, 17, 21], но они не опираются на карту местности, а хранят информацию о препятствиях в некотором абстрактном виде. Во-вторых, поскольку на выходе необходимо получить сеть остановочных пунктов, то центры рассчитываемых кластеров должны находиться на участках дорожной сети, а не в произвольной географической точке.

2. Обзор литературы

Для кластеризации георасределенных данных было проанализировано множество алгоритмов: алгоритмы среднего сдвига (Mean Shift), k-средних (k-means), иерархический метод BIRCH, алгоритмы DBSCAN и OPTICS. В качестве метрик расстояний между точками были рассмотрены евклидова метрика, решение обратной геодезической задачи и расчет расстояний по городским дорогам с при помощи сервиса построения маршрутов Open Source Routing Machine (OSRM) [34].

Решение задачи кластеризации является неоднозначным. Во-первых, не существует всеми принятого критерия качества кла-

стеризации. Существует множество логически или эмпирически выведенных критериев качества, также существует множество алгоритмов, которые не имеют четко выраженного критерия качества, но осуществляющих достаточно разумную кластеризацию. Во-вторых, число кластеров, как правило, является неизвестным заранее и устанавливается определенным критерием, отличающегося у различных конкретных алгоритмов. В-третьих, результат кластеризации существенно зависит от выбора метрики ρ, выбор которой также субъективен и определяется экспертом [3, 18].

Поскольку выбор метрики зачастую является абсолютно субъективным, то для решения задачи кластеризации создаются новые, всё более сложные, алгоритмы и улучшаются старые. Одним из широко применяемых алгоритмов для кластеризации является алгоритм k-средних [5, 8, 15, 24, 27, 29, 45].

Алгоритм k-средних производит разбиение входной выборки на k кластеров. Действие алгоритма таково, что он стремится минимизировать среднеквадратичное отклонение точек кластера от его центра. Основная мысль алгоритма заключается в том, что на каждом шаге перевычисляется центр масс каждого кластера, полученного на предыдущем шаге, а затем выборка переразбивается на кластеры вновь в соответствии с тем, какой из новый центров кластеров оказался ближе по выбранной метрике. Алгоритм завершается, когда на каком-либо шаге не происходит изменения кластеров.

Достоинствами метода являются его простота и быстрота использования (вычислительная сложность равна $O(nki)$, где n — число объектов выборки, k — число кластеров, i — число итераций алгоритма), а также его понятность и прозрачность. Недостатками алгоритма являются: (i) чувствительность к выбросам, (ii) необходимость задавать количество кластеров, (iii) необходимость сканировать всю выборку для определения положения каждого кластера.

Из-за медленной работы алгоритма на больших выборках данных, а также простоты идеи и неплохих результатов для большинства выборок, предлагается множество способов ускорить

или улучшить алгоритм k-средних.

Так, в работе [15] для ускорения работы алгоритма k-средних, предложено использование неравенства треугольника. Ускоренный алгоритм избегает излишних вычислений расстояния с помощью сохранения «верхних» и «нижних» границ для дистанций между точками и центрами, для расчета которых двумя различными методами применяется неравенство треугольника. Ускоренный алгоритм выдает те же результаты, что и обычный, но при этом количество вычислений дистанций сокращается в десятки раз.

Еще один вариант ускорения алгоритма путем сокращения лишних вычислений предлагают авторы статьи [28]: после каждой итерации алгоритма необходимо разделять кластеры на те, которые изменили своё положение, так называемые «активные», и «статические» — оставшиеся на месте; и продолжать дальнейшие расчеты для активных кластеров. При этом для корректировки списков сохраняется минимальное значение расстояний от центров кластера до точек, при достижении которых кластер снова становится «активным».

В работе [36] алгоритм ускоряется при помощи хранения результатов в так называемых «мульти-размерных kd-деревьях». В узлах дерева хранятся центры кластеров, точки ищутся в «листьях» — особых гиперпрямоугольниках, что позволяет сократить количество расчетов.

В этой же статье рассматривается еще один алгоритм — так называемый алгоритм «Черного списка». Идея заключается в идентификации тех центров, которым точно не будут принадлежать точки из листьев, и исключить их из расчетов.

В статье [24] предлагается «алгоритм фильтрации». Он также основан на хранении многомерных данных в структуре kd-деревьев, а отличается тем, что выбор кластера, которому должен принадлежать лист, осуществляется построением гиперплоскости между претендентами.

Авторы работы [45] оценивают отношение расстояний между точкой и двумя центрами кластеров, и приходят к выводу,

что большая часть «активных», то есть сдвигающих центр, точек находится в определенном диапазоне значений этого отношения. Используя этот факт на шаге присвоения точек, они создают ускоренную версию алгоритма k-средних, однако результат в данном случае оказывается не точным, а приближенным.

В диссертации [23] был описан «быстрый жадный алгоритм k-средних». Предположение, используемое в алгоритме, такое же, как в обычном жадном (глобальном) алгоритме [29]; оно заключается в том, что глобальный оптимум может быть достигнут при запуске алгоритма с $(k-1)$ центрами, и k-тым центром, достраиваемым автоматически на подходящую позицию. Автор объединил варианты ускорения, предложенные в [36] и [24], с наивным алгоритмом.

В работе [8] предложен усовершенствованный метод k-средних, который путем последовательного запуска алгоритма для центров от 1 до k приближает решение к глобальному минимуму искажения. Это достигается за счет процедуры определения векторов-кандидатов на выбор позиции для вставки нового центра, при этом общее время работы изменяется незначительно.

Кришна и Мёрти в своей работе [27] представляют «генетический k-средних» — генетический алгоритм, в котором в качестве операции кроссовера выступает операция k-средних. Сам генетический алгоритм был изменен для специфической работы — кластеризации.

Результат работы алгоритма k-средних очень сильно зависит от начального распределения кластеров. В работе [11] рассматривается метод k-means++, в котором начальное распределение центров кластеров задается особым образом, опираясь на вероятности, рассчитанные по кратчайшим расстояниям от точек до выбранных центров.

В работе [5] представлен алгоритм квантования цветов, основанный на методе k-средних и дополненный алгоритмом гравитационного поиска [14]. После стандартного расчета расстояний по метрике, рассчитываются «массы» кластеров и «силы», действующие на них. Таким образом, в работе достигается улуч-

шение значений заданной целевой функции.

Работы [33] и [38] описывают построение словарного дерева с использованием двух различных вариаций алгоритма k-средних: иерархического и приближенного соответственно. Иерархический алгоритм строит дерево результатов, на каждом следующем уровне разбивая имеющиеся кластеры на количество, называемое «коэффициентом разветвления», при этом каждый из полученных кластеров рассчитывается независимо от других. В приближенном алгоритме изначально каждое дерево развернуто до листьев, затем итерационно создается очередь приоритетных узлов до достижения конкретного числа открытых путей по дереву. При этом увеличение количества деревьев не приводит к значительному увеличению затрачиваемого времени.

В работе [7] алгоритм k-средних контролируется введением и комбинированием эвристических критериев: критерия оценки ошибки кластеризации и информационного критерия функциональной эффективности.

В работе [37] алгоритм k-средних дополнен оценкой количества кластеров с помощью алгоритма X-means, запускаемого после каждой итерации основного алгоритма. X-means основан на двух идеях для определения оптимального количества кластеров: создание нового кластера рядом с одним из существующих и разделение некоторого количества кластеров на равноотстоящие друг от друга кластеры. Решение о принятии созданной структуры принимается на основе Байесовского информационного критерия.

Определение оптимального количества кластеров во входной выборке является одной из сложнейших проблем в кластерном анализе [10, 11, 37, 40, 42].

Авторы работы [42] предлагают метод определения количества кластеров, основанный на искажениях: после очередной кластеризации рассчитываются соответствующие искажения, затем искажения трансформируются, рассчитываются значения «скачков» и среди них выбирается значение количества кластеров.

В работах [40] и [10] для определения количества класте-

ров используется метод «силуэтов». Каждый кластер представляется силуэтом, основанном на его плотности и разреженности. В [40] автор предлагает метод построения и применения силуэтов для оценки результатов кластерного анализа, а в [10] описывается применение различных методов (индекс Данна, индекс Калинского-Харабаза и др.) для определения количества кластеров в зашумленной выборке.

В статье [30] для определения кластерной структуры предварительно используется генетический алгоритм и метод силуэтов, которые в совокупности определяют оптимальные параметры для кластеризации методом k-средних.

Тханг, Пантюхин и Галушкин в работе [9] представляют гибридный алгоритм кластеризации на основе алгоритма k-средних и DBSCAN – плотностного алгоритма для кластеризации зашумленных пространственных данных. При существенном сокращении времени кластеризации точность при некоторых параметрах достигает точности алгоритма DBSCAN.

Также существует множество других алгоритмов кластеризации, основанных на различных идеях о поиске внутренней структуры данных. Можно выделить несколько больших групп алгоритмов по способу кластеризации [21]: разделяющие методы, иерархические методы, плотностные методы и сеточные методы.

Для кластеризации при обработке изображений и данных высокой размерности зачастую применяют алгоритм среднего сдвига, или Mean Shift. В статье [13] автор использует этот алгоритм для определения границ регионов и фона на изображениях, а также описывает принцип работы алгоритма и критерии подбора его параметров.

Для кластеризации динамических наборов точек авторы работы [12] разработали модель, названную инкрементной кластеризацией. Она основана на тщательном анализе требований поисковых систем, целью модели является сохранить кластеры малого диаметра при добавлении новых точек.

Для кластеризации категориальных данных в работе [22] применяется обобщение парадигмы метода k-средних — метод

k-modes. В нем используется особая метрика, вместо среднего значения набора данных находится их мода и используется частотный метод обновления мод.

В работе [46] описывается алгоритм BIRCH — иерархичесий алгоритм кластеризации, особенно эффективный при обработке очень больших выборок. Алгоритм является локальным — принятие решений относительно одного кластера производится без перебора всех данных; алгоритм используют на данных с четко выраженными кластерами; алгоритм является иерархическим, благодаря чему может находить скрытые субкластеры, а время работы линейно маштабируемо.

Проблема кластеризации георасределенных данных с учетом препятствий была описана в работах [21, 44]. В качестве наглядного примера был приведен пример расстановки банкоматов в области, через которую протекает река.

Для решения этой задачи в [44] предлагалось использовать измененный алгоритм CLARANS (более подробно рассмотрен в работе [21]). В [21] перед основной кластеризацией алгоритмом CLARANS предлагается сделать предварительную кластеризацию алгоритмом BIRCH. Сам алгоритм CLARANS был немного изменен для применения к решению задачи кластеризации георасределенных данных. В работе [26] так же рекомендуется использовать алгоритм CLARANS, поскольку он эффективно обрабатывает большие пространственные базы данных. В качестве альтернативы рассматривается алгоритм BIRCH из-за более быстрой работы. В работах [16] и [17] данные о препятствиях рассчитываются из диаграмм Воронова и Делоне с помощью алгоритмов AUTOCLUST и AUTOCLUST+ соответственно.

Использование подходов накладывают дополнительные требования к запуску алгоритмов, такие как задание значений параметров алгоритмов (например, *maxneighbor* и *numlocal* в [26]).

Таким образом, если рассматривать задачу создания остановочных пунктов методом кластеризации георасределенных данных с учетом рельефа, то рассмотренные способы и подходы не решают сформулированные задачи и имеют следующие недо-

статки: (а) не учитывают предпочтения жителей по перемещениям в городской среде; (б) не адаптированы для работы с реальными георасределенными данными, получаемых из картографических сервисов и (в) требуют проведения дополнительного исследования для подбора допустимых значений параметров, задаваемых априорно.

3. Метод кластеризации георасределенных данных

3.1. ОБЩЕЕ ОПИСАНИЕ

Алгоритм k-средних строит k кластеров, расположенных таким образом, чтобы минимизировать среднеквадратичное отклонение объектов выборки от центров кластеров. Выбор числа k может базироваться на результатах теоретических соображений, предшествующих исследований, визуального наблюдения или иных источниках информации [6].

Идеальным кластером алгоритм k-средних считает сферу с центром кластера в центре сферы [1].

При известном числе кластеров алгоритм k-средних начинает работу с некоторого начального расположения кластеров, к которым присваиваются объекты выборки.

Начальное расположение кластеров очень сильно влияет на работу алгоритма. Из-за неправильного выбора начальных значений алгоритм может попасть в некоторый локальный минимум целевой функции (рис. 1).

Целевой функцией алгоритма является среднеквадратичное расстояние (при использовании евклидовой метрики) между объектами выборки и центрами их кластеров: $f(X, C) = \sum_{j=1}^{k} \sum_{i=1}^{n} \|x_i - \mu_j\|^2$, где μ_j — центр кластера C_j, вычисляемый по формуле:

$$\mu_j = \frac{1}{|C_j|} \sum_{i=1}^{n} x_i. \tag{1}$$

[7] *http://www.math.le.ac.uk/people/ag153/homepage/KmeansKmedoids/Kmeans_Kmedoids.html*

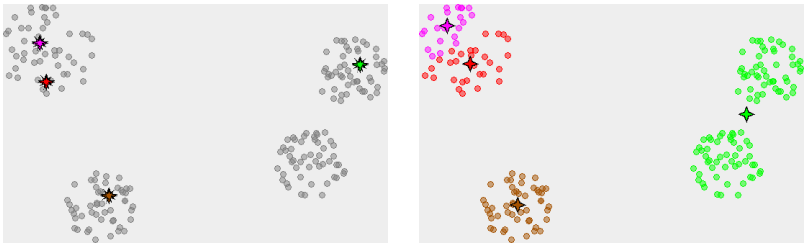


Рис. 1. Пример сходимости алгоритма к локальному минимуму. Кружками обозначены объекты выборки, звездами — центры кластеров. Рисунок получен с помощью апплета Миркеса[7]

Начальное расположение кластеров может задаваться как вручную, так и автоматически. В изначальном варианте алгоритма начальное расположение кластеров было случайным [31]. На практике в качестве начальных центров кластеров зачастую либо выбирают первые k объектов выборки, либо выбирают k самых отдаленных друг от друга объектов выборки.

Алгоритм k-средних относительно масштабируем и эффективен при обработке не слишком больших баз данных поскольку его вычислительная сложность равна $O(nki)$, где n — полное число объектов выборки, k — число кластеров, i — число итераций алгоритма; обычно $k \ll n$ и $i \ll n$ [21].

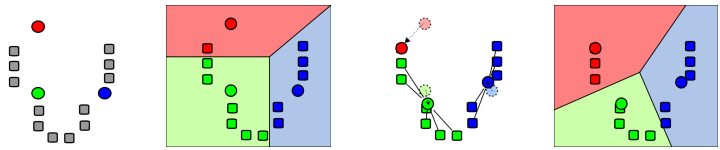
Недостатками алгоритма являются:

- чувствительность к выбросам — точки, далеко находящиеся от всех кластеров, сильно искажают среднее;
- необходимость задавать количество кластеров — при работе с алгоритмом выбор числа кластеров является самым сложным вопросом. Если нет предположений о кластерной структуре данных, то рекомендуют постепенно наращивать число k, сравнивая полученные результаты [6];
- необходимость сканировать всю выборку для определения положения каждого кластера — при работе с большими базами данных это сильно замедляет работу.

Достоинствами метода являются его простота и быстрота использования, а также понятность и прозрачность алгоритма.

3.2. ПОЯСНЕНИЯ К РАБОТЕ

Работу алгоритма можно разбить на три этапа: начальный этап, этап присвоения и этап сдвига.



а) б) в) г)

Рис. 2. Демонстрация работы алгоритма. Иллюстрация взята из статьи об алгоритме k-средних[8]

На начальном этапе происходит выбор числа k, расстановка начальных центров кластеров C_0 (рис. 2а).

На этапе присвоения происходит расчет расстояния от каждого объекта выборки X до каждого объекта кластера C_j. После чего считается, что объект принадлежит тому кластеру, до которого у него минимальное расстояние (рис. 2б).

На этапе сдвига происходит расчет нового положения центров кластеров по формуле (1) (рис. 2в).

Этапы присвоения и сдвига повторяются, пока кластерные центры не стабилизируются, то есть все объекты, принадлежащие кластеру на прошлой итерации, принадлежат и на этой, либо пока не достигнется заданное максимальное число итераций I. На выходе алгоритм выдает множество центров кластеров C и метки принадлежности объектов выборки кластерам L (рис. 2г).

Псевдокод алгоритма представлен на схеме 1.

Алгоритм 1 (k-средних)**.**

1. $C = C_0$, $L_0 = \varnothing$, $i = 0$
2. Пока $L! = L_0$ и $i < I$
   1. $L_0 = L$, $A = \varnothing$, $\mu = \varnothing$
   2. Для каждого $x \in X$
      1. $r = \varnothing$
      2. Для каждого $c \in C$
         1. Рассчитать $r_{xc} = \rho(x, c)$

[8] *https://en.wikipedia.org/wiki/k-means_clustering*

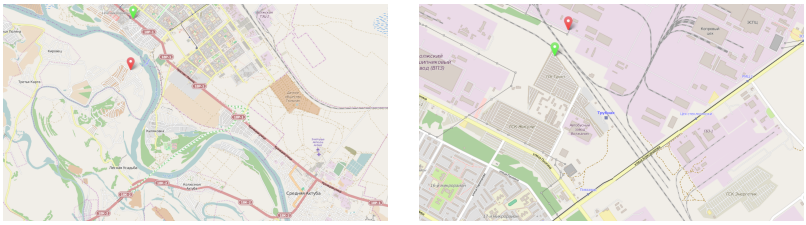


Рис. 3. Близкие по евклидовой метрике пары объектов

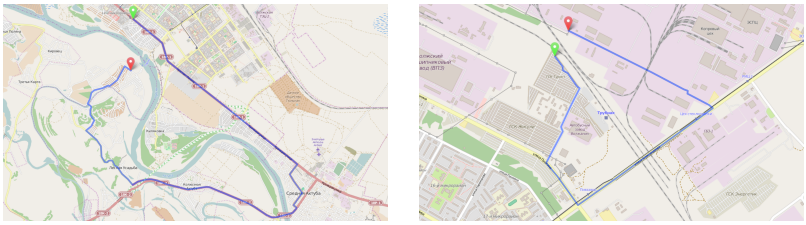


Рис. 4. Расстояние между объектами, которое должно учитываться при расчете

2. $r = r + \{r_{xc}\}$
3. $L[x] = C[\min(r)]$
4. $A[C[\min(r)]] = A[C[\min(r)]] + \{x\}$

3. Для каждого $c \in C$
   1. $c = \mu[c] = \sum(A[c]) / |A[c]|$
4. $i = i + 1$

3. Закончить.

3.3. Расчет расстояний между точками

Особая метрика при работе алгоритма необходима по той причине, что два близких по стандартным метрикам георазпределенных объекта в реальности могут преграждаться препятствиями. Примеры таких объектов приведены на рисунке 3. На рисунке 4 приведено реальное расстояние между объектами, которое должно учитываться при транспортных расчетах.

Для расчета расстояния между точками в работе используются две метрики, названные «Surface» и «Route».

Метрика *Surface* рассчитывает расстояние путем решения обратной геодезической задачи [25, с. 48-50]. Метрика *Route* рас-

считывает расстояние между объектами по графу городских дорог.

Для решения обратной геодезической задачи в метрике *Surface* используется библиотека *GeographicLib* [19]. Метрика *Route* использует сервис построения маршрутов *Open Source Routing Machine* (OSRM) [34] для расчета расстояния между точками по городским дорогам.

Сервис OSRM предоставляет открытый код своего движка маршрутизации. Для работе с собранным движком на пользовательском компьютере используются HTTP-запросы определенной структуры: так называемое HTTP-API сервиса (поддерживается две версии API: v4 и v5). Результат обработки запроса выводится в json-формате, содержащем всю информацию по запросу.

В метрике используются два типа запросов: *route* (*viaroute* в версии HTTP-API v4) для расчета расстояний и *nearest* (*locate* в версии HTTP-API v4) для сдвига кластеров к дорожной сети. Ответ на запрос *route* содержит список точек, через которые проложен маршрут, и список маршрутов; каждый маршрут содержит свою длину. Ответ на запрос *nearest* содержит список точек на графе дорог с указанием расстояний до них.

Для работы движок маршрутизации должен распаковать карту сервиса *Open Street Map* [35] для выделения на ней графа дорог. Главным параметром при работе с движком маршрутизации является используемый профиль (англ. *profile*). При сборке OSRM из исходных кодов предоставляется несколько профилей, среди который профили автомобиля, велосипеда и пешехода. Используемый профиль влияет на способ построения маршрута и выбор городских дород для его прокладки. При работе с метрикой *Route* использовался профиль автомобиля. Пример работы сервиса Open Source Routing Machine представлен на рисунке 5.

Рис. 5. Пример работы сервиса Open Source Routing Machine [34]

4. Испытание и обоснование эффективности предлагаемых подходов

Предложенные алгоритмы были реализованы с использованием языка программирования Python и сервиса построения маршрутов *Open Source Routing Machine* для расчета расстояния между узлами графа по городским дорогам. Программный код опубликован на хостинге Github.

4.1. Методика проведения эксперимента

Для оценки эффективности работы алгоритма и изучения специфики метрики *Route* были проведены эксперименты, в ходе которых менялись выборки данных, количество получаемых кластеров и их начальное местоположение, а так же реализация метода.

Для генерации данных о предпочтениях жителей города по перемещению был создан инструмент *Scatter* [39], позволяющий удобным образом генерировать большие объемы геоданных.

Для оценки работы алгоритма на данных, приближенных к реальным, были сгенерированы данные о предпочтениях по перемещению жителей среднего по размерам города с примерным числом жителей около 300 000. В качестве результата было полу-

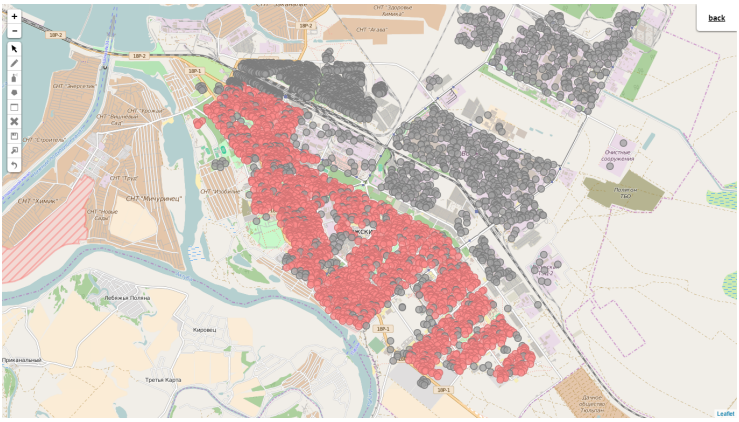


Рис. 6. Сгенерированная выборка из 12 000 точек. Красным отмечены точки отправления, серым — назначения

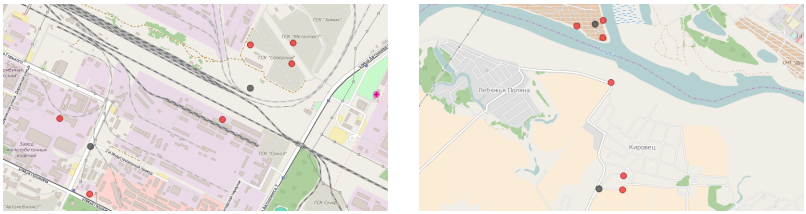
чено 6 000 пар точек отправления–назначения (рис. 6), или 12 000 точек в общей сложности (выборка *Main*). Эту выборку было решено разбить на 125 кластеров, начальное положение которых было случайным[9]. В последствие это случайное расположение кластеров было взято за основу для тестирования различных версий алгоритма.

Так же были сгенерированы другие выборки для проверки специфичных случаев: обхода препятствий в виде железной дороги (выборка *Railway*, рис. 7а) и реки (выборка *River*, рис. 7б). Каждая из них представляет собой набор из шести точек, которые кластеризуются в два кластера. Точки и начальные центры расположены таким образом, что если алгоритм не учитывает препятствия между объектами выборки, то в одном кластере окажутся точки, разделенные препятствием.

Критериями для оценки эффективности разработанных алгоритмов и метрик являются:

- время, затраченное на расчет расстояния между объектами выборки;

[9] *Логика разбиения основывается на числе остановочных пунктов в городе*



а) б)

Рис. 7. Выборки для проверки обхода препятствий. На рисунке а) между объектами выборки находится железная дорога, на рисунке б) — река. Красными кругами отмечены элементы выборки, черными — начальные центры кластеров

- учет препятствий;
- визуальная оценка результатов кластеризации.

5. Описание результатов

Результаты работы алгоритма k-средних доступны по следующей ссылке: https://vstu-cad-stuff.github.io/clustering/kmeans/.

Результаты работы алгоритма k-средних со сдвигом центров кластеров к дорожной сети доступны по следующей ссылке: https://vstu-cad-stuff.github.io/clustering/terminals/.

Для тестирования работы алгоритма было использовано 4 выборки:

- выборка *Main*: $|X| = 12000$ точек, $k = 125$ (рис. 6); используется для общего тестирования работы алгоритмов;
- выборка *Railway*: $|X| = 6$ точек, $k = 2$ (рис. 7а); используется для проверки обхода препятствий;
- выборка *River*: $|X| = 6$ точек, $k = 2$ (рис. 7б); используется для проверки обхода препятствий;
- выборка *Common*: $|X| = 180$ точек, $k = 10$ (рис. 8); используется для тестов алгоритмов на скорость.

На рисунках, показывающих результаты кластеризации линиями очерчены *кластерные области* — выпуклые оболочки множества точек, принадлежащих кластеру.

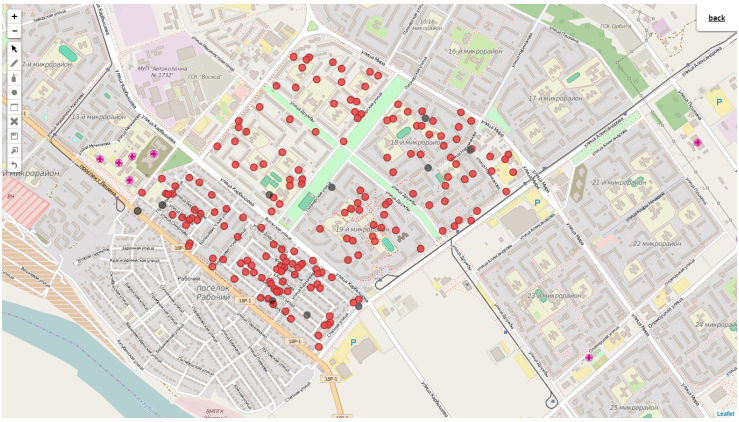


Рис. 8. Выборка Common. Красными кругами отмечены элементы выборки, черными — начальные центры кластеров

5.1. Последовательная реализация

Данная реализация является наиболее простой из всех с точки зрения расчета расстояния. В данном случае применяется метрика *Surface*. Эта версия может быть интересна в качестве ориентира в тесте на производительность. Данный вариант не позволяет учитывать препятствия, результаты кластеризации выборок *Railway* и *River* представлены на рис. 9.

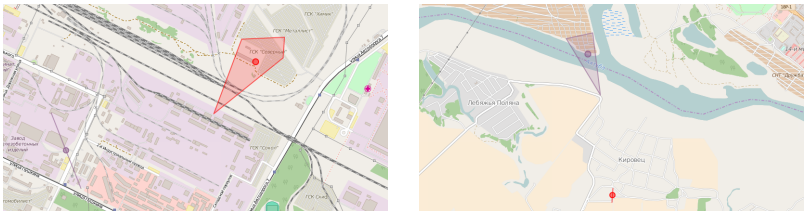


Рис. 9. Результаты кластеризации выборок Railway и River с метрикой Surface

5.2. Последовательная с использованием OSRM

Использование метрики *Route* существенно увеличивает время расчета расстояния между объектами. Использование OSRM

является узким местом в реализации, однако эта метрика позволяет учитывать препятствия, результаты кластеризации выборок *Railway* и *River* представлены на рис. 10

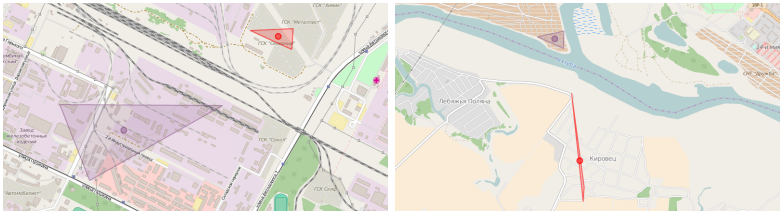


Рис. 10. Результаты кластеризации выборок Railway и River с метрикой Route

5.3. Параллельная с использованием OSRM

Для ускорения расчетов при использовании метрики *Route* предлагается распараллеливание внутреннего цикла по X (шаги 2.2.1–2.2.4) алгоритма 1. Распараллеливание применимо, так как результаты обработки одного объекта выборки не влияют на обработку других объектов.

5.4. Результаты

Результаты обработки выборки *Main* представлены на рисунках 11, 12.

Таблица 1. Среднее время выполнения одного расчета расстояния при различных метриках и реализациях алгоритма, мс

Реализация	Euclid	Surface	Route
Последовательная	0,0665	0,709	4,785
Параллельная (2)	0,0659	0,692	4,069
Параллельная (4)	0,0654	0,663	3,596

В таблице 1 приведены средние длительности выполнения одного расчета дистанции между геопространственными объектами для трех реализаций алгоритма: последовательной, параллельной на два потока и параллельной на 4 потока, и для трех

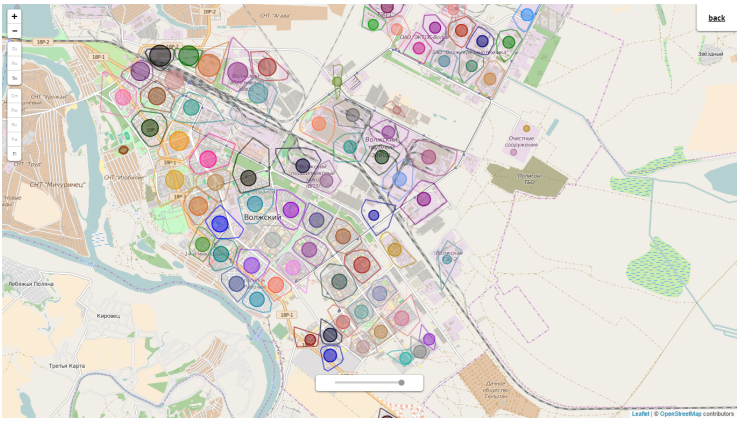


Рис. 11. Результаты обработки выборки Main метрикой Surface

метрик: *Euclid*, *Surface* и *Route*. Метрика *Euclid* является обычной евклидовой метрикой: $\rho(a,b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$ и приводится для сравнения. Величины приведены в миллисекундах.

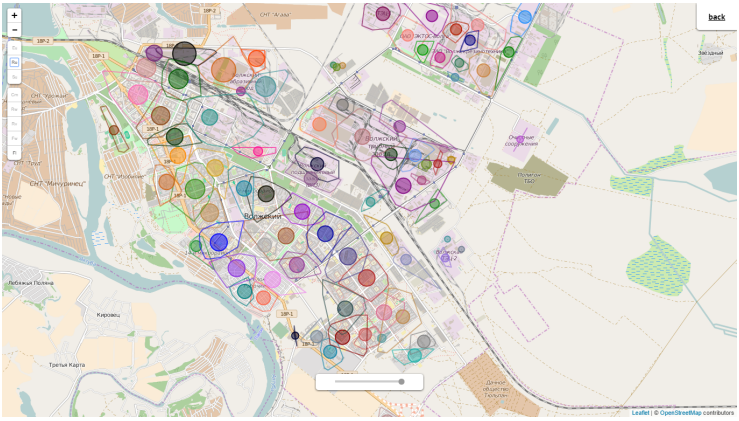


Рис. 12. Результаты обработки выборки Main метрикой Route

Результат работы алгоритма со сдвигом центров кластеров на ближайший узел графа дорожной сети приведен на рис. 13:

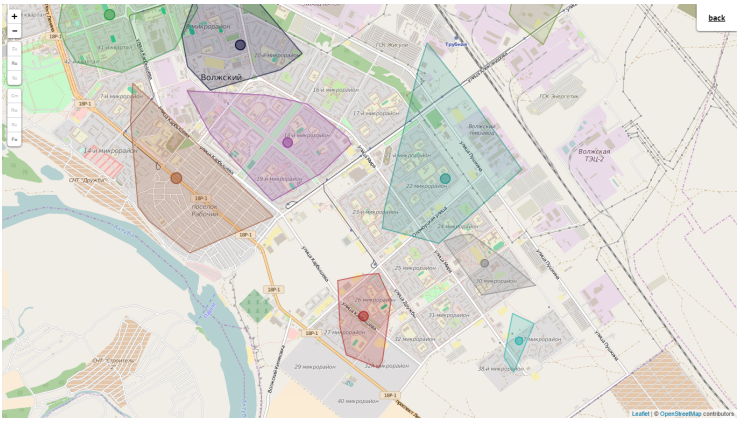


Рис. 13. Результат работы алгоритма со сдвигом центров кластеров к дорожной сети

6. Обсуждение результатов

Из рис. 9 видно, что метрика *Surface* не способна учитывать препятствия при расчетах, в то время как метрика *Route* учитывает их (рис. 10). То, что метрика *Surface* не учитывает препятствия, видно и на рисунке 11.

На рисунке 12 видно, что кластерные области пересекаются. Это происходит из-за того, что при расчете расстояния сервис OSRM выбирает ближайшие к элементам выборки участки дорожной сети, а затем строит маршрут по графу дорог между ними. Из-за этого элементы, расположенные, например, по разные стороны домов, могут попасть на различные участки дорог, и расстояние между ними и центром кластера получается различным. Это также происходит и из-за обхода метрикой препятствий, из-за чего близкие точки оказываются в разных кластерах и создается наложение областей кластеров. Поскольку элементы, принадлежащие одному кластеру, на рисунках показаны в виде выпуклых оболочек, то из-за нескольких элементов кластера, лежащих в стороне от других, создается видимое наложение областей кластеров.

Одна итерация алгоритма при кластеризации выборки *Main* требует $12\,000 \times 125 = 1{,}5$ млн рассчетов дистанций между объектами выборки и центрами кластеров. При использовании метрики *Surface* кластеризация была выполнена за 28 итераций алгоритма, то есть потребовалось 42 млн расчетов. При использовании метрики *Route* кластеризация была выполнена за 21 итерацию алгоритма, то есть потребовала 31,5 млн расчетов расстояний. Таким образом, соотнеся эти сведения с результатами из таблицы 1, на одну итерацию с метрикой *Surface* понадобилось примерно 16,5 минут, на всю кластеризацию — около 8 часов. На одну итерацию с метрикой *Route* понадобилось примерно 1,5 часа, а на всю кластеризацию — 31,5 часа или почти полтора дня. Для сравнения, кластеризация с евклидовой метрикой потребовала 19 итераций, на выполнение которых было потрачено чуть больше получаса. Результат кластеризации выборки *Main* евклидовой метрикой приведен на рис. 14.

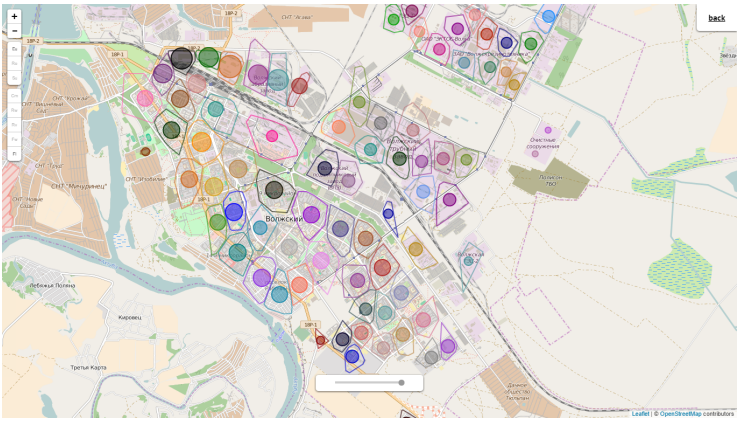


Рис. 14. Результаты обработки выборки Main евклидовой метрикой

На рисунке 13 видно, что некоторые кластерные центры сдвинулись к магистралям, а некоторые остались в жилых кварталах. Так происходит из-за отсутствия возможности детального рассмотрения кандидатов на расположение центра кластера —

участков дорожной сети — средствами OSRM.

7. Выводы и перспективы

Подводя итог, следует отметить, что для кластеризации географически распределенных данных предлагаемый подход эффективен по нескольким причинам. Во-первых, ключевой особенностью подхода является вычисление расстояний между объектами по графу городских дорог, что позволяет учитывать все препятствия, расположенные между ними. Это позволяет учитывать при кластеризации ситуации, изображенные на рисунке 3. Во-вторых, это также позволяет выполнять основную функцию системы — создание остановочных пунктов, расположенных на участках дорог. В-третьих, используется алгоритм k-средних, обладающий невысокой вычислительной сложностью и отсутствием привязки к плотности данных.

Недостатки данного подхода в большей степени связаны со временем, которое требуется сервису OSRM на расчет дистанции. Этот недостаток, вероятно, можно частично сгладить использованием одного из ускоряющих методов, описанных в разделе 2. Также недостатком подхода можно считать определение пользователем числа k кластеров, однако такой вариант задавания числа кластеров гораздо нагляднее и удобнее для пользователя, чем определение, например, параметра ширины окна в алгоритме Mean Shift. Способом устранить этот недостаток может быть использование предполагающих методов, например, использование метода силуэтов. Недостаток, связанный с привязкой остановочных пунктов к дорожным узлам внутри жилых кварталов, связан с отсутствием описания точек в ответе сервиса OSRM на запрос. Этот недостаток может быть устранен поиском предложенных OSRM узлов дорожной сети в структуре карты OpenStreetMap, и добавления условия по тэгу (англ. *tag*) с ключом *highway*.

В целом, данный подход может быть использован для кластеризации географически распределенных данных и для построения сети остановочных пунктов общественного транспорта, особенно в тех

случаях, если на рассматриваемой местности находятся какие-либо естественные или искусственные препятствия, или время кластеризации не столь важно, как получаемый результат.

Литература

1. БОЛЬШАКОВА, Е. И. *Автоматическая обработка текстов на естественном языке и компьютерная лингвистика* : учеб. пособие / Е. И. Большакова, Э. С. Клышинский, Д. В. Ландэ, А. А. Носков, О. В. Пескова, Е. В. Ягунова — М.: МИЭМ, 2011. — 272 с.
2. БУРЕ, В. М., МАЗАЛОВ, В. В., ПЛАКСИНА, Н. В. Вычисление характеристик пассажиропотоков в транспортных системах / Управление большими системами. Выпуск 47. М.: ИПУ РАН, 2014. С.77-91.
3. ВОРОНЦОВ, К. В. *Машинное обучение. Курс лекций* [Электронный ресурс]. — Режим доступа: http://www.machinelearning.ru/wiki/images/6/6d/Voron-ML-1.pdf (дата обращения: 25.03.2016).
4. ЗАХАРОВ, В. В., КРЫЛАТОВ А. Ю. Моделирование конкурентной маршрутизации экологически безопасных транспортных потоков на городской транспортной сети / Управление большими системами. Выпуск 55. М.: ИПУ РАН, 2015. С.185-223.
5. ЛИСИН А. В., ФАЙЗУЛЛИН Р. Т. *Применение метаэвристических алгоритмов к решению задач кластеризации методом k-средних* // Компьютерная оптика. — 2015. — Т. 39, №. 3. — С. 406–412.
6. НЕЙСКИЙ, И. М. *Классификация и сравнение методов кластеризации* // Интеллектуальные технологии и системы. Сборник учебно-методических работ и статей аспирантов и студентов. — М.: НОК «CLAIM», 2006. — Выпуск 8. — С. 130–142.
7. ПЕТРОВ, С. А. И ДР. *Гібридний алгоритм кластер-аналізу для формування апріорного розбиття простору ознак на класи знань в системах дистанційного нав-*

чання // Вісник Вінницького політехнічного інституту. — 2015. — №. 4. — С. 80–87.

8. ТКАЧЕНКО, О. М. И ДР. *Метод кластеризации на основе последовательного запуска k-средних с усовершенствованным выбором кандидата на новую позицию вставки* // Научные труды Винницкого национального технического университета. — 2012. — №. 2.
9. ТХАНГ, В. В., ПАНТЮХИН, Д. В., ГАЛУШКИН, А. И. *Гибридный алгоритм кластеризации FastDBSCAN* // Труды Московского Физико-Технического Института. — 2015. — Т. 7, №. 3. — С. 77–81.
10. AMORIM DE, R. C., HENNIG, C. *Recovering the number of clusters in data sets with noise features using feature rescaling factors* // Information Sciences. — 2015. — Vol. 324. — P. 126–145.
11. ARTHUR, D., VASSILVITSKII, S. *k-means++: The advantages of careful seeding* // Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. — Society for Industrial and Applied Mathematics, 2007. — P. 1027–1035.
12. CHARIKAR, M. ET AL. *Incremental clustering and dynamic information retrieval* // SIAM Journal on Computing. — 2004. — Vol. 33, №. 6. — P. 1417–1440.
13. COMANICIU, D., MEER, P. *Mean shift: A robust approach toward feature space analysis* // Pattern Analysis and Machine Intelligence, IEEE Transactions on. — 2002. — Vol. 24, №. 5. — P. 603–619.
14. DUMAN, S., GÜVENÇ, U., YÖRÜKEREN, N. *Gravitational Search Algorithm for Economic Dispatch with Valve-Point Effects* // International Review of Electrical Engineering 2010. — 2010. — Vol. 5. — P. 2890–2895.
15. ELKAN, C. *Using the triangle inequality to accelerate k-means* // ICML. — 2003. — Vol. 3. — P. 147–153.
16. ESTIVILL-CASTRO, V., LEE, I. *Argument free clustering for large spatial point-data sets via boundary extraction from*

Delaunay Diagram // Computers, Environment and urban systems. — 2002. — Vol. 26, №. 4. — P. 315–334.

17. ESTIVILL-CASTRO, V., LEE, I. *Clustering with obstacles for geographical data mining* // ISPRS Journal of Photogrammetry and Remote Sensing. — 2004. — Vol. 59, №. 1. — P. 21–34.
18. FRALEY, C., RAFTERY, A. E. *Model-based clustering, discriminant analysis, and density estimation* // Journal of the American statistical Association. — 2002. — Vol. 97, №. 458. — P. 611–631.
19. *GeographicLib — a small set of C++ classes for converting between geographic, UTM, UPS, MGRS, and geocentric coordinates* [Электронный ресурс]. — Режим доступа: http://geographiclib.sourceforge.net/ (дата обращения: 14.11.2015).
20. GOLUBEV, A., CHECHETKIN, I., SOLNUSHKIN, K.S., SADOVNIKOVA, N., PARYGIN, D., SHCHERBAKOV, M. *Strategway: web solutions for building public transportation routes using big geodata analysis* // Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services. — ACM, 2015. — P. 91–94.
21. HAN, J., KAMBER, M., TUNG, A. K. H. *Spatial Clustering Methods in Data Mining: A Survey* // Geographic Data Mining and Knowledge Discovery, Research Monographs in GIS. — 2001. — P. 201–231.
22. HUANG, Z. *A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining* // DMKD. — 1997. — 8 p.
23. HUSSEIN, N. *A Fast Greedy k-means Algorithm* : Master's Thesis. // University of Amsterdam, Amsterdam. — 2002.
24. KANUNGO, T. ET AL. *An efficient k-means clustering algorithm: Analysis and implementation* // Pattern Analysis and Machine Intelligence, IEEE Transactions on. — 2002. — Vol. 24, №. 7. — P. 881–892.
25. KARNEY, C. F. F. *Algorithms for geodesics* // Journal of

Geodesy. — 2013. — Vol. 87, №. 1. — P. 43–55.

26. KOPERSKI, K., HAN, J., ADHIKARY, J. *Mining knowledge in geographical data* // Communications of ACM. — 1998. — Vol. 26.
27. KRISHNA, K., MURTY, M. N. *Genetic k-means algorithm* //Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on. — 1999. — Vol. 29, №. 3. — P. 433–439.
28. LAI, J. Z. C., HUANG, T. J., LIAW, Y. C. *A fast k-means clustering algorithm using cluster center displacement* // Pattern Recognition. — 2009. — Vol. 42, №. 11. — P. 2551–2556.
29. LIKAS, A., VLASSIS, N., VERBEEK, J. J. *The global k-means clustering algorithm* // Pattern recognition. — 2003. — Vol. 36, №. 2. — P. 451–461.
30. LLETÍ, R. ET AL. *Selecting variables for k-means cluster analysis by using a genetic algorithm that optimises the silhouettes* // Analytica Chimica Acta. — 2004. — Vol. 515, №. 1. — P. 87–100.
31. MACQUEEN, J. ET AL. *Some methods for classification and analysis of multivariate observations* // Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. — 1967. — Vol. 1, №. 14. — P. 281–297.
32. NIELSEN, G., LANGE, T. *Network Design for Public Transport Success — theory and examples* // Norwegian Ministry of Transport and Communications, Oslo. — 2008.
33. NISTER, D., STEWENIUS, H. *Scalable recognition with a vocabulary tree* // Computer vision and pattern recognition, 2006 IEEE computer society conference on. — IEEE, 2006. — Vol. 2. — P. 2161–2168.
34. *Open Source Routing Machine* [Электронный ресурс]. — Режим доступа: http://project-osrm.org/ (дата обращения: 25.11.2014).
35. *OpenStreetMap — The Free Wiki World Map — An openly licensed map of the world being created by volunteers using local knowledge, GPS tracks and donated sources* [Электрон-

ный ресурс]. — Режим доступа: https://www.openstreetmap.org (Дата обращения: 25.11.2014).

36. PELLEG, D., MOORE, A. W. *Accelerating exact k-means algorithms with geometric reasoning* // Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. — ACM, 1999. — P. 277–281.
37. PELLEG, D., MOORE, A. W. *X-means: Extending k-means with Efficient Estimation of the Number of Clusters* // Proceeding ICML '00 Proceedings of the Seventeenth International Conference on Machine Learning. — 2000. — Vol. 1. — P. 727–734.
38. PHILBIN, J. ET AL. *Object retrieval with large vocabularies and fast spatial matching* // Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on. — IEEE, 2007. — P. 1–8.
39. *Project "Scatter" — generate geospatial data* [Электронный ресурс]. — Режим доступа: https://vstu-cad-stuff.github.io/scatter/ (Дата обращения: 15.03.2015).
40. ROUSSEEUW, P. J. *Silhouettes: a graphical aid to the interpretation and validation of cluster analysis* // Journal of computational and applied mathematics. — 1987. — Vol. 20. — P. 53–65.
41. SADOVNIKOVA, N. *Evaluating the sustainability of Volgograd* / N. Sadovnikova, D. Parygin, E. Gnedkova, B. Sanzhapov, and N. Gidkova. // In The Sustainable City VIII. — WIT Press, 2013.
42. SUGAR, C. A., JAMES, G. M. *Finding the number of clusters in a dataset* // Journal of the American Statistical Association. — Vol. 98, №. 463. — 2003. — P. 750–763.
43. TENNENHOUSE, D. *Proactive computing* // Communications of the ACM. — 2000. — Vol. 43. — P. 43–50.
44. TUNG, A. K. H., HOU, J., HAN, J. *Spatial clustering in the presence of obstacles* // Data Engineering, 2001. Proceedings 17th International Conference on. — IEEE, 2001. — P. 359–

367.
45. WANG, J. ET AL. *Fast approximate k-means via cluster closures* // Multimedia Data Mining and Analytics. — Springer International Publishing, 2015. — P. 373–395.
46. ZHANG, T., RAMAKRISHNAN, R., LIVNY, M. *BIRCH: an efficient data clustering method for very large databases* // ACM Sigmod Record. — ACM, 1996. — Vol. 25, №. 2. — P. 103–114.

HOW TO CLUSTER BIG GEOSPATIAL DATA

Ilya Chechetkin, Volgograd State Technical University, Volgograd, student (illaech@gmail.com).
Maxim Shcherbakov, Volgograd State Technical University, Volgograd, Doctor of Science, professor (Volgograd, Lenina av., 28, (8442) 24-88-00).
Natalia Sadovnikova, Volgograd State Technical University, Volgograd, Doctor of Science, professor (Volgograd, Lenina av., 28, (8442) 24-88-00).
Danila Parygin, Volgograd State Technical University, Volgograd, Cand.Sc., senior lecturer (dparygin@gmail.com).
Alexey Golubev, Volgograd State Technical University, Volgograd, Cand.Sc., student (ax.golubev@gmail.com).

Abstract: This paper has deal with a problem of clustering big geospatial data and identifies it's features. The proposed approach to geospatial clustering is based on k-means algorithm and real distance calculations, taking into account the relief of city.

Keywords: geospatial data, route planning, clustering.

Статья представлена к публикации
членом редакционной коллегии ...

Поступила в редакцию ...
Дата опубликования ...