УДК 519.683.8

КОНЦЕПЦИЯ ПРИМЕНЕНИЯ MAPREDUCE В ИЕРАРХИЧЕСКОЙ АГЛОМЕРАТИВНОЙ КЛАСТЕРИЗАЦИИ

С.А. Ермоченко

Учреждение образования «Витебский государственный университет имени П.М. Машерова»

В статье рассматриваются особенности иерархической агломеративной кластеризации и проблемы, связанные с необходимостью распараллелить этот процесс при использовании разных мер близости объектов.

Цель работы – выработка концепции применения модели MapReduce для иерархического агломеративного кластерного анализа большого объема данных.

***Материал и методы.** Материалом являются объекты произвольной природы, имеющие набор числовых характеристик и требующие выполнения их иерархической кластеризации. Особенность набора объектов – большое их количество (более 10 000). Использованы описательно-аналитический метод и метод проектирования распределенных вычислительных систем.*

***Результаты и их обсуждение.** Для применения модели MapReduce в рассматриваемой задаче выделены операции, выполнение которых предлагается осуществлять на стадии предварительной обработки (Map), и операции для стадии свертки (Reduce). Достоинством предложенной концепции является возможность обработки большого числа объектов, информация о которых хранится в распределенных хранилищах данных. Результаты могут применяться на практике при проектировании вычислительных систем, ориентированных на конкретные предметные области.*

***Заключение.** Предложена концепция использования модели MapReduce для выполнения иерархической агломеративной кластеризации в распределенной вычислительной системе, позволяющей гибкое горизонтальное масштабирование.*

***Ключевые слова:** иерархическая агломеративная кластеризация, распределенные вычисления, модель MapReduce, обработка большого объема данных, мера близости объектов.*

CONCEPT OF USING MAPREDUCE IN HIERARCHICAL AGGLOMERATIVE CLUSTERING

S.A. Yermochenko

Educational Establishment «Vitebsk State P.M. Masherov University»

Features of hierarchical agglomerative clustering and problems connected with the necessity to parallel this process while using different measures of object proximity are considered in the article.

The purpose is to elaborate the concept of using MapReduce model for hierarchical agglomerative clustering analysis of a large amount of data.

***Material and methods.** The material is objects of arbitrary nature which have a set of numerical characteristics and require their hierarchical clustering. The peculiarity of the set of objects is their large amount (over 10 000). The descriptive and analytical method and the method of designing distributed computing systems were used.*

***Findings and their discussion.** To apply MapReduce model in the considered problem operations are singled out which are suggested to be executed at the stage of preliminary processing (Map) as well as operations for the closing stage (Reduce). The advantage of the offered concept is the possibility to process a large amount of objects, information about which is stored in the distributed data bases. The results can be used in practice in computer system design, which aim at definite object areas.*

***Conclusion.** A concept of using MapReduce model for performing hierarchical agglomerative clustering in the distributed computer system, which allows flexible horizontal scaling, is offered.*

***Key words:** hierarchical agglomerative clustering, distributed computing, MapReduce model, a large amount of data processing, measure of object proximation.*

Модель распределенных вычислений MapReduce была представлена компанией Google Inc. в 2004 году [1]. Она может использоваться для организации параллельных вычислений на кластерах из обычных персональных компьютеров или серверных компьютеров. При этом предложенная модель позволяет обрабатывать большие объемы данных (вплоть до нескольких петабайт), а за счет распределенных вычислений помогает добиться достаточно быстрой обработки даже таких значительных объемов данных, так как эффективно и практически неограниченно горизонтально масштабирует вычислительный кластер. Предоставляемые гибкие возможности по обработке больших объемов информации сделали модель MapReduce одной из основ такого феномена в информационных технологиях, как большие данные (Big Data) [2].

Вычисления по модели MapReduce проводятся в два этапа: выполнение некоторой обработки каждого набора параметров из некоторого набора (стадия Map) и объединение результатов обработки нескольких наборов параметров в некий общий результат (стадия Reduce). Подобные вычисления удобно проводить в задачах анализа некоторого объема данных. Например, обработка текста с целью поиска ключевых слов в этом тексте. Эта задача кажется не столь сложной при обработке одного текста. Но когда речь идет об обработке огромного количества текстов и о составлении семантического ядра каждого из них для поискового робота, то в этом случае модель MapReduce позволяет на порядок увеличить скорость обработки за счет распараллеливания простейших операций.

Тем не менее даже распределенные вычисления не всегда помогают быстро получить результат. В этом случае зачастую массив данных кластеризуют (разбивают на определенные группы), а затем к каждой группе применяют свои методы анализа. Например, при анализе рынка потребления товаров данные о продажах могут кластеризоваться по группам товаров, по стоимости или по временным интервалам. Как правило, подобная кластеризация осуществляется классическим способом, а ее результат уже передается на обработку распределенной вычислительной системе, построенной по модели MapReduce.

Сам по себе кластерный анализ – достаточно хорошо и давно изученная область математики. Существуют самые различные методы кластерного анализа, описанные еще в 70–80-х годах XX века [3]. Под классическим способом здесь понимаются методы, не требующие параллельных вычислений. Для уменьшения объема обрабатываемых данных применяются методы кластеризации, ориентированные на фиксированное число кластеров, и задача кластеризации сводится к тому, что каждый анализируемый объект относится к тому или иному известному кластеру. Существуют также методы кластеризации, в которых количество кластеров заранее не известно и определяется в ходе процедуры кластеризации. К таким методам и относится иерархическая агломеративная кластеризация [4]. Этот вид кластеризации удобно использовать на данных, структура которых еще не известна. Имеется в виду структура взаимосвязей между объектами – какие из них более близки между собой и по каким характеристикам. По сути, такая кластеризация является подвидом анализа данных на предмет скрытых закономерностей. Иерархическая кластеризация, ввиду особенностей алгоритмов, эффективно работает на небольших объемах данных. Существуют готовые инструменты, позволяющие проводить подобную кластеризацию (например, пакет прикладных программ «Statistica»), или библиотеки для специализированных языков программирования (например, Python или R). Но библиотеки языков программирования не имеют встроенной поддержки MapReduce или других механизмов распараллеливания задачи кластеризации. Часто такие библиотеки способны обрабатывать лишь данные, находящиеся в оперативной памяти, что вообще делает невозможным обработку большого объема информации, хранящейся, например, в базе данных. Пакет прикладных программ «Statistica» имеет отдельную корпоративную версию, позволяющую производить анализ больших данных, но стоимость названного пакета очень высока.

Цель статьи – выработка концепции применения модели MapReduce для иерархического агломеративного кластерного анализа большого объема данных.

Для этого поставим следующие задачи:

– проанализировать метод иерархической агломеративной кластеризации на предмет возможности его распараллеливания;

– выработать концепцию реализации метода иерархической агломеративной кластеризации в рамках модели MapReduce;

– оценить потенциал горизонтального масштабирования вычислительной распределенной системы и временную сложность работы системы.

Материал и методы. Основные методы исследования – описательно-аналитический и проектирования, в частности, проектирования распределенных вычислительных систем.

Для проектирования распределенных систем необходимо выделить состав узлов системы (количество и назначение узлов) и способ связи между этими узлами.

Поскольку целью работы является выработка концепции применения модели MapReduce, то предметная область обрабатываемых данных не имеет существенного значения. Будем рассматривать данные об n объектах произвольной природы, где n – достаточно большое число, для определенности будем считать $n > 10\,000$.

Будем считать также, что каждый объект характеризуется k числовыми параметрами p_{ij}, где $i \in [1, n]$, $j \in [1, k]$. Эти параметры составляют данные о рассматриваемых объектах. Будем считать эти параметры нормализованными: $p_{ij} \in [0, 1]$. Нормализация параметров может осуществляться различными методами, в том числе с учетом веса того или иного параметра с точки зрения особенностей предметной области. Для дальнейшего рассмотрения способ нормализации будет несущественным.

Результаты и их обсуждение. Для анализа метода иерархической агломеративной кластеризации на предмет возможности его распараллеливания опишем сначала кратко, в чем состоит этот метод [4].

Иерархические методы в целом базируются на построении иерархии вложенных кластеров. В этой иерархии кластеры образуют дерево, в корне которого один кластер, содержащий в себе все рассматриваемые объекты. Этот кластер состоит из меньших кластеров, те могут дробиться дальше. Самые мелкие кластеры, листья дерева, содержат по одному объекту каждый. Агломеративные методы основаны на объединении кластеров в более крупные кластеры до тех пор, пока все они не будут объединены в один суперкластер. При этом процесс построения иерархической кластерной структуры начинается с того, что создаются n кластеров, каждый из которых содержит по одному исходному объекту. Затем наиболее близкие между собой кластеры объединяются в более крупные кластеры. Этим агломеративные методы отличаются от дивизивных методов, которые начинают обработку данных с того, что сначала объединяют все объекты в один кластер, а затем постепенно дробят кластеры на более мелкие до тех пор, пока каждый объект не окажется единственным в своем отдельном кластере.

Основным положением метода иерархической агломеративной кластеризации является выбор из текущего множества кластеров верхнего уровня некоего подмножества кластеров, наиболее близких между собой.

Рассмотрим способы определения меры близости кластеров между собой. Для этого сначала необходимо установить меру близости отдельных объектов между собой [3].

Один из самых простых способов определения меры близости между объектами – классическое Евклидово расстояние:

$$d(s,t) = \sqrt{\sum_{j=1}^{k} \left(p_{sj} - p_{tj}\right)^2} \,. \tag{1}$$

Или расстояние Чебышева:

$$d(s,t) = \max_{j \in [1,k]} \left| p_{sj} - p_{tj} \right| . \tag{2}$$

Иногда используется и простое линейное расстояние (или Манхеттонское расстояние, также называемое расстоянием городских кварталов):

$$d(s,t) = \sum_{j=1}^{k} \left| p_{sj} - p_{tj} \right| . \tag{3}$$

Можно рассматривать и обобщенное степенное расстояние Минковского:

$$d(s,t) = \sqrt[\nu]{\sum_{j=1}^{k} \left(p_{sj} - p_{tj}\right)^{\nu}} \,. \tag{4}$$

Во всех указанных формулах индексы s и t являются индексами соответствующих объектов: $s \in [1, n]$, $t \in [1, n]$. В формуле (4) параметр $\nu \geq 1$ и выбирается исследователями произвольно, исходя из особенностей предметной области. Легко видеть, что формула (4) является обобщением остальных

трех формул. Так, при v = 1 получаем формулу (3), при v = 2 – формулу (1), а при $v \to \infty$ – формулу (2). Также важно отметить, что любой способ определения меры близости должен отвечать определенным требованиям, в частности, $d(s, t) = 0$, если $s = t$.

Расстояние (1) – самое универсальное, как с точки зрения точности полученного значения, так и с точки зрения простоты его вычисления. Расстояния (2) и (4) могут использоваться в некоторых специальных случаях, в которых оно позволяет улучшить точность анализа, но эти метрики вычисляются несколько сложнее. Самое простое для вычисления, но наименее точное – это расстояние (3). С точки зрения вычислительной сложности данные метрики кардинально друг от друга не отличаются. Однако при обработке большого количества объектов вычисление расстояния между парами объектов необходимо будет производить многократно. В таком случае способы вычисления расстояния уже могут оказывать заметное влияние на скорость кластеризации столь значительного количества объектов.

Теперь рассмотрим способы определения меры близости кластеров между собой. Таких способов несколько больше, чем способов определения расстояния между самими объектами. Но для понимания сути метода иерархической агломеративной кластеризации различия в этих способах несущественны. Хотя они и оказывают решающую роль на качественную структуру кластеризации, это имеет смысл лишь применительно к конкретной предметной области [4]. Поэтому здесь проанализируем лишь некоторые способы определения меры близости кластеров между собой, которые отличаются вычислительной сложностью.

Метод одиночной связи, также известный как метод ближайшего соседа:

$$d(x, y) = \min_{s \in X, t \in Y} d(s, t). \quad (5)$$

Он определяет расстояние между двумя кластерами как расстояние между двумя самыми близкими объектами (или подкластерами) данных кластеров. Здесь множества X и Y – это множества индексов объектов (или подкластеров) соответственно первого кластера (x) и второго кластера (y).

Метод полной связи (или метод дальнего соседа):

$$d(x, y) = \max_{s \in X, t \in Y} d(s, t). \quad (6)$$

Устанавливает расстояние между двумя кластерами как расстояние между двумя самыми дальними объектами (или подкластерами) данных кластеров.

Метод средней связи:

$$d(x, y) = \frac{1}{|X| \cdot |Y|} \sum_{s \in X} \sum_{t \in Y} d(s, t). \quad (7)$$

Определяет расстояние между двумя кластерами как усредненное расстояние между всеми парами объектов (или подкластеров) данных кластеров. Здесь $|X|$ и $|Y|$ – мощности соответствующих кластеров (количество объектов в них, включая объекты, входящие в подкластеры, или, по-другому, сумма мощностей всех подкластеров).

Центроидный метод:

$$d(x, y) = d(s_c, t_c). \quad (8)$$

Он устанавливает расстояние между двумя кластерами как расстояние между центрами масс (центроидами) данных кластеров. Здесь s_c и t_c – индексы объектов (или подкластеров), имеющие усредненные характеристики в пределах всего своего кластера. При этом подобные объекты могут не совпадать с реальными объектами соответствующих кластеров.

Оценку сложности определения таких метрик будем производить вместе с анализом самого алгоритма метода иерархической агломеративной кластеризации.

Рассмотрим теперь сам алгоритм метода и проанализируем возможности и трудности его распараллеливания. При описании метода удобнее использовать матричный подход. Если у нас есть n некоторых объектов, то, определяя с помощью некоторой функции расстояния между всевозможными парами таких объектов, составим матрицу D, элементы которой вычисляются по формуле:

$$d_{ij} = d(i, j), \forall i \in [1, n], j \in [1, n]. \quad (9)$$

Количество элементов в матрице будет равно n^2, что делает вычисление элементов матрицы процессом достаточно трудоемким, но хорошо распараллеливаемым, так как каждый элемент вычисляется независимо. При этом стоит отметить, что данная матрица является симметричной, а также элементы главной диагонали этой матрицы должны быть нулевыми (ввиду свойства функции расстояния). Таким

образом, необходимо вычислить n (n – 1) / 2 элементов. Однако асимптотическая сложность данного шага алгоритма все еще остается равной O(n^2).

После вычисления данной матрицы начинается процесс объединения кластеров, состоящих на первом этапе из одного объекта, в более крупные кластеры. Для этого в матрице находят минимальное расстояние (не считая нулей на главной диагонали). Далее в кластер попадают такие объекты, расстояние между которыми равно найденному минимальному расстоянию. При этом возможны ситуации, когда минимальное расстояние встречается в матрице несколько раз. Проиллюстрируем это примером. В роли объектов рассмотрим набор точек на плоскости. В качестве их характеристик рассмотрим их декартовы координаты. Таким образом, в нашем примере n = 7, k = 2. Расположение объектов приведено на рис. 1:

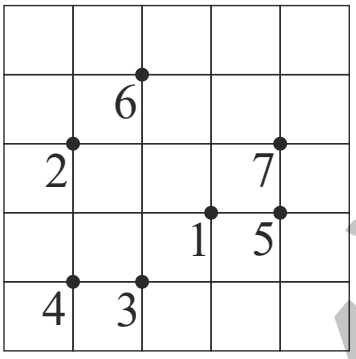


Рис. 1. **Начальный набор объектов**

Характеристики (координаты) объектов приведем в следующей табл. 1 (числа подобраны условно, лишь с демонстративной целью):

Таблица 1

Характеристики объектов

Номер объекта	Координата *x*	Координата *y*
1	3	2
2	1	3
3	2	1
4	1	1
5	4	2
6	2	4
7	4	3

Вычислим элементы матрицы расстояний, применяя Евклидову метрику (1), с точностью до 2 знаков после запятой и получим следующую матрицу расстояний (табл. 2):

Таблица 2

Матрица расстояний

	1	2	3	4	5	6	7
1	0	2,24	1,41	2,24	1,00	2,24	1,41
2	2,24	0	2,24	2,00	3,16	1,41	3,00
3	1,41	2,24	0	1,00	2,24	3,00	2,83
4	2,24	2,00	1,00	0	3,16	3,16	3,61
5	1,00	3,16	2,24	3,16	0	2,83	1,00
6	2,24	1,41	3,00	3,16	2,83	0	2,24
7	1,41	3,00	2,83	3,61	1,00	2,24	0

Минимальное расстояние в матрице из табл. 2 равно 1,00. Но так как это расстояние встречается не единожды (этому расстоянию равны расстояния между парами объектов {1, 5}, {3, 4} и {5, 7}), то встает вопрос о правилах формирования кластера (кластеров) из этих объектов. В данном случае образуется два кластера: {1, 5, 7} и {3, 4} (рис. 2).

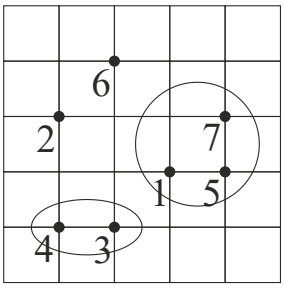


Рис. 2. **Результат первого этапа кластеризации**

При распараллеливании поиска минимального значения в матрице расстояний особых трудностей не возникает. Для каждого вычислительного узла будет выделена часть матрицы расстояний, в которой будет производиться минимум. Затем результат работы каждого узла будет обрабатываться централизованно (то есть из всех найденных минимумов необходимо будет найти общий минимум). Однако здесь встает вопрос синхронизации операций.

Во-первых, узел, выполняющий поиск минимума в назначенной ему части матрицы расстояний, должен начать свою работу, когда эта часть матрицы уже заполнена. Конечно, это можно делать и одновременно, с помощью одного из подходов в решении задачи «читателей и писателей» из области параллельного программирования.

Во-вторых, нахождение общего минимума также должно синхронизироваться (начинаться после операций поиска промежуточных минимумов).

Но поиск объектов, которые можно объединить в один кластер, уже будет плохо распараллеливаться, так как искать эти соответствующие расстояния необходимо по всей матрице, потому что их расположение не локализовано. Однако поиск расстояния по сложности не сравним со сложностью вычисления данного расстояния, поэтому эта проблема не сказывается существенным образом на распараллеливании всего алгоритма.

После выделения кластеров следует из матрицы расстояния исключить те объекты, которые попали во вновь созданные кластеры, а вместо этих объектов ввести в матрицу расстояния эти кластеры. Для этого необходимо пересчитать расстояние от этих кластеров до каждого из объектов (кластеров), которые остались в матрице расстояний без изменений.

Возможность и способ распараллеливания данного этапа работы алгоритма зависят от способа вычисления расстояния между кластерами. При вычислении расстояния методом одиночной (5) или полной (6) связи можно воспользоваться уже подсчитанными расстояниями. Здесь возникает вопрос: нужно ли постоянно хранить матрицу расстояний или требуемые расстояния можно при необходимости пересчитывать? Ответ на него зависит от количества анализируемых объектов и доступных вычислительных ресурсов.

При организации распределенной вычислительной системы для решения задачи кластерного анализа мы имеем достаточно большой объем обрабатываемых данных. Передавать полную копию этих данных каждому вычислительному узлу будет не только не рационально, но и невозможно, потому что даже для хранения информации обо всех имеющихся объектах, как правило, используют распределенные хранилища. Поэтому такая распределенная система должна относиться к классу систем с общей памятью. И прежде всего при проектировании подобных систем необходимо определиться с тем, сколько узлов будет задействовано под хранение информации (будет формировать общую память), а также сколько будет выделено вычислительных узлов для обработки этой информации. Кроме того, эти узлы могут иметь различные технические характеристики, позволяющие говорить об общем

объеме доступной памяти в распределенной вычислительной системе, о скорости доступа к этой памяти для операций чтения и записи, а также о вычислительной мощности данной системы. Но в целом, учитывая тенденцию, что стоимость хранения данных ниже стоимости вычислительных ресурсов, чаще промежуточные данные хранятся для обеспечения большей производительности.

Проиллюстрируем теперь пересчет расстояний между кластерами на примере одиночной связи. Обозначим новый кластер {1, 5, 7} индексом 8, а кластер {3, 4} индексом 9. Тогда расстояние от кластера 8 до кластера 2 определяется расстоянием между объектом 2 и объектом 1, так как оно минимально из расстояний 2–1, 2–5, 2–7. Остальные расстояния пересчитываются аналогичным образом. Тогда получим новую матрицу расстояний, приведенную в табл. 3.

Таблица 3

Пересчитанная матрица расстояний

	2	6	8	9
2	0	1,41	2,24	2,00
6	1,41	0	2,24	3,00
8	2,24	2,24	0	1,41
9	2,00	3,00	1,41	0

Если же рассмотреть другие способы вычисления расстояния между кластерами, то для процесса распараллеливания кардинальных изменений они не внесут. Так, например, метод средней связи (7) основан на вычислении среднего арифметического между всеми возможными расстояниями пар объектов двух кластеров. Данная задача достаточно сложна вычислительно, особенно при большом количестве анализируемых объектов. В свою очередь, центроидный метод (8) дает схожие результаты, но имеет меньшую вычислительную сложность. К тому же этот метод позволяет хранить меньшее количество промежуточных данных. При формировании нового кластера создается некий искусственный объект, имеющий усредненные характеристики кластера. Далее для расчета расстояний от этого кластера до других объектов достаточно использовать только характеристики обозначенного искусственного объекта. А за счет того, что на каждом шаге количество кластеров уменьшается, то и объем промежуточных данных с течением времени сокращается.

Для рассмотренного выше примера применение метода средней связи на первом шаге указано в табл. 4.

Таблица 4

Матрица расстояний для метода средней связи

	2	6	8	9
2	0	1,41	2,80	2,12
6	1,41	0	2,44	3,08
8	2,80	2,44	0	2,58
9	2,12	3,08	2,58	0

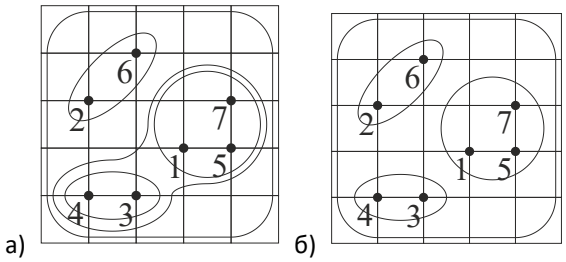
Центроидный метод дает результат на первом шаге рассматриваемого алгоритма, показанный в табл. 5.

Таблица 5

Матрица расстояний для центроидного метода

	2	6	8	9
2	0	1,41	2,75	2,06
6	1,41	0	2,36	3,04
8	2,75	2,36	0	2,55
9	2,06	3,04	2,55	0

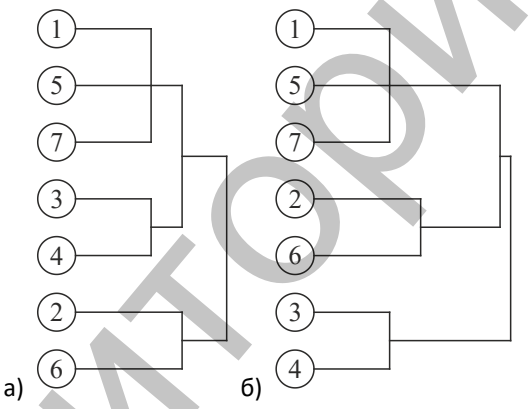
Вне зависимости от метода нахождения расстояния между двумя кластерами данный процесс продолжается до тех пор, пока не останется один кластер, содержащий в себе все объекты. Для нашего примера получим следующие способы разбиения на кластеры (рис. 3).



а) односвязный метод, б) центроидный метод

Рис. 3. **Результат кластеризации**

Стоит отметить, что приведенный на рис. 3 результат кластеризации применим только к этому конкретному примеру. При работе с большим количеством характеристик ($k > 2$) такие построения уже невозможны. Поэтому для представления результата кластеризации используют, как правило, диаграммы специального вида – дендрограммы. Дендрограммы для рассмотренного примера приведены на рис. 4.



а) односвязный метод, б) центроидный метод

Рис. 4. **Дендрограммы**

Таким образом, мы познакомились с алгоритмом построения иерархической структуры кластеров агломеративным методом. Эту иерархию удобно представить в постоянном хранилище, так как строить графическую дендрограмму для большого объема данных бессмысленно – подобная дендрограмма будет слишком перегруженной и не информативной. Но тем не менее построить иерархическую структуру кластеров для большого числа объектов в постоянном хранилище с помощью распределенной вычислительной системы допустимо. Большинство операций алгоритма кластеризации могут быть распараллелены. Проблема, которая может возникать при организации данной системы, – это управление вычислительными узлами с целью синхронизации их работы. Такую работу как раз и следует поручить инструментам, реализующим вычислительные системы по модели MapReduce.

Основная идея MapReduce модели заключается в том, что все операции, которые выполняются в распределенной вычислительной системе, делятся на два вида: операции Map и операции Reduce [1]. Эти операции определенным образом распределяются между вычислительными узлами системы. Как правило, способ распределения и функции, которые должен выполнять каждый узел, настраиваются в конфигурационных файлах и с помощью разрабатываемых модулей расширения. Операция Map – простая функция, которой передается один набор данных. Эта операция позволяет сопоставить некоторому входному набору значений одно выходное значение. Для агломеративной иерархической

кластеризации операция Map поможет вычислять расстояния между простыми объектами. Операция Reduce – это функция, которой передается не один, а несколько наборов значений, которые объединяются в некоторое простое значение или некоторый простой набор. Это операция уменьшения размерности. Для рассматриваемого алгоритма кластеризации подобная операция позволит вычислять расстояния между кластерами в системе. Данная операция объединяет кластеры в более крупные кластеры, уменьшая размерность обрабатываемых данных.

Оценим теперь сложность отдельных шагов алгоритма при реализации их на распределенной вычислительной сети. Будем ориентироваться на систему, в которой предусматривается хранение промежуточных данных, таких как вычисленные расстояния между объектами и кластерами, характеристики искусственно созданных объектов.

В худшем случае (при объединении кластеров) на каждой итерации количество кластеров будет уменьшаться на 1. Таким образом, для n объектов, для которых заданы k характеристик, имеем следующее. Количество расстояний, вычисляемых между всевозможными парами объектов, составляет $n(n-1)/2$. Количество итераций, на которых происходит объединение одного кластера: $n-1$. На m-ной итерации размерность матрицы расстояний будет составлять $n-m+1$, где $m \in [1, n-1]$. Для поиска минимума необходимо на m-ой итерации выполнить $(n-m)(n-m+1)/2$ операций сравнения. Также на каждой итерации следует как минимум вычислить $n-m-1$ расстояний от вновь образованного кластера до каждого из оставшихся кластеров.

В общем итоге получаем:

1. Количество самых трудоемких операций (сверток) по вычислению расстояний между кластерами и объединению кластеров имеет асимптотический порядок $O(n \ln n)$.

2. Количество операций по вычислению расстояний между первичными объектами со средней трудоемкостью имеет асимптотический порядок $O(n^2)$.

3. Количество операций по поиску минимума, что по трудоемкости является самой простой операцией, но имеет сложности в синхронизации этого шага с другими шагами алгоритма, имеет асимптотический порядок $O(n^2)$.

Полученная оценка временной сложности является приемлемой, особенно учитывая, что две из трех операций распараллеливаются очень эффективно, и лишь одна операция требует дополнительной синхронизации, но при этом позволяет распараллеливание. Это за счет простоты реализации и низкой трудоемкости операций сравнения может быть реализовано на сравнительно небольшом числе вычислительных узлов. При наличии же в вычислительной системе узла с мощным многоядерным процессором подобную задачу вполне может решить один вычислительный узел.

Оценим необходимую емкость памяти для хранения входных, промежуточных и выходных данных. Для хранения входных данных понадобится $n \cdot k$ вещественных чисел. Выходными данными будут данные о номерах кластеров ($2n-1$ целых числа), о том, к какому кластеру относится каждый из рассматриваемых объектов (n целых чисел) и для каждого кластера номер его родительского кластера и максимальное расстояние, на котором располагаются объекты внутри данного кластера ($2n-1$ целых числа и столько же вещественных чисел). Стоит также обратить внимание, что для хранения расстояния нет смысла хранить вещественное число с двойной точностью, так как эти числа лишь показывают меру близости объектов, но не характеризуют сами объекты. Поэтому можно считать, что расстояние – это тоже целое число (с точки зрения выделяемой ему памяти). Таким образом, получаем для выходных данных $7n-3$ целых числа. Учитывая, что характеристики объекта – вещественные числа (обычно двойной точности), а их количество, как правило, от 5 до 20, то объем выходных данных может быть в 2–3 раза меньше объема входных данных. Что же касается промежуточных данных, то для хранения матрицы расстояний необходимо $n(n-1)/2$ вещественных чисел одиночной точности. Также для хранения расстояний между кластерами требуется еще порядка $n(n-1)/2$ вещественных чисел одиночной точности. Хотя на самом деле при добавлении в систему информации о расстоянии между кластерами информация о расстоянии между простыми объектами уже не нужна, и ее можно не хранить (в большинстве методов иерархической кластеризации), здесь мы этого учитывать не будем. В худшем случае, при использовании центроидных методов вычисления меры сходства между кластерами,

для промежуточных данных понадобится еще $n \cdot k$ вещественных чисел для хранения данных об искусственно вводимых объектах.

Как видно, при обработке большого объема данных основной проблемой может стать не скорость их обработки, а проблема хранения промежуточных данных, поэтому выбор метода оценивания близости кластеров лучше подбирать с учетом начального объема данных.

В ряде случаев применение иерархических методов вообще может оказаться невозможным, при этом можно использовать методы кластеризации на фиксированное, заранее заданное число кластеров.

Заключение. В предложенной работе была рассмотрена концепция организации иерархической агломеративной кластеризации больших объемов данных. В качестве метода кластеризации выбрана именно иерархическая кластеризация как дающая больше всего информации для анализа данных, структура и особенности которых еще не известны. В исследовании рекомендовано организовать процесс кластеризации в распределенной вычислительной системе, архитектура которой строится в соответствии с моделью MapReduce. Также были предложены те шаги алгоритма иерархической кластеризации, которые могут быть выполнены на этапе предварительной обработки данных (стадия Map) и на этапе уменьшения размерности данных (стадия Reduce).

Выработанная концепция проанализирована на предмет возможности и эффективности распараллеливания. Также приведены оценки сложности различных этапов вычислений и оценки объема хранимых данных в зависимости от объема обрабатываемых данных.

К основным результатам допустимо отнести полученную концепцию применения распределенных вычислений к иерархической агломеративной кластеризации, а также описанное влияние особенностей методов расчета меры близости объектов и кластеров между собой на возможность и потенциальные сложности для распараллеливания.

В целом исследование носит более теоретический характер. Интересным может быть применение данной концепции к обработке некоторого большого массива реальных данных. Однако на практике такая обработка будет иметь исследовательский интерес, если обрабатывать одни и те же данные разными методами и сравнивать результаты по скорости и используемому объему памяти. Но для анализа особенностей распараллеливания метода иерархической кластеризации необходимо нивелировать аппаратные характеристики вычислительной системы. Для этого одни и те же данные нужно обрабатывать на нескольких различных системах, что требует значительных временных и финансовых затрат.

Данная работа может иметь практическое применение при изучении методов обработки данных в учебном процессе. Существует несколько различных инструментов, реализующих модель MapReduce, а также языков программирования, позволяющих реализовать методы кластеризации. Такой подход поможет усилить межпредметные связи и продемонстрировать студентам различные способы анализа данных.

ЛИТЕРАТУРА

1. Dean, J. MapReduce: Simplifed Data Processing on Large Clusters [Electronic resource] / J. Dean, S. Ghemawat. – Google Inc., 2004. – Mode of access: https://storage.googleapis.com/pub-tools-public-publication-data/pdf/16cb30b4b92fd4989b8619a61752a2387c6dd474.pdf. – Date of access: 20.08.2019.
2. Chen, M. Big Data. Related Technologies, Challenges, and Future Prospects / M. Chen, Sh. Mao, Y. Zhang, V.C.M. Leung. – Springer, 2014. – 100 p.
3. Мандель, И.Д. Кластерный анализ / И.Д. Мандель. – М.: Финансы и статистика, 1988. – 176 с.
4. Жамбю, М. Иерархический кластер-анализ и соответствия / М. Жамбю. – М.: Финансы и статистика, 1988. – 345 с.

REFERENCES

1. Dean, J. MapReduce: Simplifed Data Processing on Large Clusters [Electronic resource] / J. Dean, S. Ghemawat. – Google Inc., 2004. – Mode of access: https://storage.googleapis.com/pub-tools-public-publication-data/pdf/16cb30b4b92fd4989b8619a61752a2387c6dd474.pdf. – Date of access: 20.08.2019.
2. Chen, M. Big Data. Related Technologies, Challenges, and Future Prospects / M. Chen, Sh. Mao, Y. Zhang, V.C. M. Leung. – Springer, 2014. – 100 p.
3. Mandel I.D. *Klasternyi analiz* [Cluster Analysis], Moscow: Finansy i statistika, 1988, 176 p.
4. Jambue M. *Iyerarkhicheski klaster-analiz i sootvetstviya* [Hierarchical Cluster Analysis and Correspondences], Moscow: Finansy i statistika, 1988, 345 p.

Поступила в редакцию 26.08.2019
Адрес для корреспонденции: e-mail: yermochenko@gmail.com – Ермоченко С.А.