УДК 004.021

АНАЛИЗ ЭФФЕКТИВНОСТИ АЛГОРИТМА MAPREDUCE ПРИ РЕШЕНИИ ЗАДАЧИ ВЫЧИСЛЕНИЯ СУММАРНОГО ВРЕМЕНИ, ЗАТРАЧЕННОГО СИСТЕМОЙ TRACKOR ONEVIZON, НА РАБОТУ С ПОЛЬЗОВАТЕЛЕМ ЗА МЕСЯЦ С ПОМОЩЬЮ APACHE HADOOP

Диденко А.А., магистрант гр. ПИм-171, II курс
Научный руководитель: Пимонов А.Г., д.т.н., профессор
Кузбасский государственный технический университет имени Т.Ф. Горбачева
г. Кемерово

Введение. В современном обществе все чаще возникают проблемы, связанные с анализом большого объема данных, таких как логи транзакций, геоданных, статистики, а также использования интернет-сайтов и много другого. Для того чтобы решить данные проблемы компания Google разработала и выложила в открытый доступ алгоритм распределенных вычислений MapReduce [1], который лег в основу свободно распространяемого проекта Apache Software Foundation, который состоит из набора библиотек, утилит, а также фреймворка для разработки и выполнения распределённых программ, которые могут работать на кластерах, состоящих из сотен и тысяч узлов [2]. В данной работе представлены анализ и демонстрация работы данного алгоритма на примере решения задачи вычисления суммарного времени, затраченного системой Trackor Onevizon, на работу с пользователем за месяц с помощью Apache Hadoop на кластере, поднятом в виртуальной машине.

Trackor Onevizon. Платформа Trackor OneVizion представляет из себя веб-приложение, которое решает проблему управления сложноорганизованной информацией, состоящей из различного типа данных. Данная платформа интегрирует управление информацией, управление документами, прогнозирование, управление задачами, а также картографирование Google с целью предоставления полнофункционального программного решения для управления бизнесом [3]. Все изменения данной системы логируются в одном из восьми логов: ошибки помещаются в error_log, действия пользователей системы в usage_log, http запросы в и из системы в rule_http_call_log, уведомления в notif_processing_log, интеграции с другими системами в integration_log, изменения конфигурации системы в comp_object_log, информация о импорте и экспорте компонентов системы в bpl_comp_log, изменения данных в audit_log. В ходе данной статьи будут рассмотрен только usage_log, потому что только он потребуются для решения задачи, описанной в следующем пункте. Структура данного лога представлена на рис. 1. Далее будут перечислены колонки, представляющие интерес для решения задачи:

1. USAGE_LOG_ID – первичный ключ;

2. ACTION_ID – ид действия, совершённого пользователем, например загрузка грида, определенного тракор типа, вход и выход из системы, открытие формы редактирования тракора;
3. BROWSER_RUNTIME – время обработки запроса браузером пользователя в секундах;
4. DB_RUNTIME – время обработки запроса базой данных в секундах;
5. IIS_FINISH – время завершения действия, используется для определения даты и времени событий;
6. IIS_RUNTIME – время генерации грида в секундах;
7. PAGE_NAME – имя открываемой страницы;
8. RESPONSE_DELIVERY_TIME – время на отправку данных клиенту из базы данных в секундах;
9. UN – имя пользователя;
10. USER_ID – ид пользователя;
11. USER_RUNTIME – время с того момента, как данные были отправлены пользователю, и до момента, когда они были отображены в браузере в секундах.

USAGE_LOG

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
ACTION_ID	NUMBER	Yes	null	3	Action, possible values are "grid", "logon"
ACTION_STATUS_ID	NUMBER	Yes	null	32	0 - SUCCESS, 1 - FAIL (user request fails in case of DB or web server error)
BROWSER_RUNTIME	NUMBER	Yes	null	28	Time spent on client side (executing of js code and parsing data)
CLIENT_IP	VARCHAR2(39)	Yes	null	22	IP address of a client
COLS_COUNT	NUMBER	Yes	null	35	
DB_RUNTIME	NUMBER	Yes	null	15	Time DB spent for executing query and return result to IIS
ERROR_MESSAGE	VARCHAR2(1000)	Yes	null	33	
FILTER_NAME	VARCHAR2(200)	Yes	null	17	Filter name
FILTER_USER_ID	NUMBER	Yes	null	16	Filter owner
GATHER_SQL_STATS	NUMBER(1,0)	Yes	null	38	When 1 sql exec plan will be collected for user grid load action based on usage_log_id comment inside the SQL
GRID_PAGE_ID	NUMBER	Yes	null	31	
HIGHLIGHT_MODE	NUMBER	Yes	null	25	View Options Highlight Mode
IIS_FINISH	DATE	Yes	null	6	Grid generation finish time
IIS_RUNTIME	NUMBER	Yes	null	7	Grid generation time
IIS_START	DATE	Yes	current_date	5	Grid generation start time
IS_EDIT_MODE	NUMBER	Yes	null	26	Is Edit Mode On
LOGIN_AS_BY_USER_ID	NUMBER	Yes	null	37	
NOT_LIMITED_SQL	CLOB	Yes	null	30	SQL without grid paging limit
PAGE_NAME	VARCHAR2(255)	Yes	null	4	Name VisionEPM page
PROCESS_ID	NUMBER	Yes	null	36	Process id from Excel Report
PROGRAM_ID	NUMBER	No	null	14	Foreign key to "V_PROGRAM" table
RESPONSE_DELIVERY_TIME	NUMBER	Yes	null	27	Time spent on grid data transfer to client
ROWS_COUNT	NUMBER	Yes	null	34	Total Number of Rows in a Dataset
SERVER_IP	VARCHAR2(39)	Yes	null	21	IP address of web server
SQL	CLOB	Yes	null	11	SQL query used to generate data grid
TMP_REQUEST_TOKEN	VARCHAR2(16)	Yes	null	29	Token for tracking status of request on client
UN	VARCHAR2(50)	Yes	null	2	User name
URL	VARCHAR2(2000)	Yes	null	10	Page URL
USAGE_LOG_BROWSER_ID	NUMBER	Yes	null	20	Foreign key to "usage_log_new3_BROWSER"
USAGE_LOG_ID	NUMBER	Yes	null	1	Primary key
USER_ID	NUMBER	Yes	null	9	Foreign key to "USERS" table
USER_RUNTIME	NUMBER	Yes	null	8	Time since data sent to client and shown in browser
VIEW_NAME	VARCHAR2(200)	Yes	null	19	View Options name
VIEW_USER_ID	NUMBER	Yes	null	18	View Options owner
WS_REQUEST	CLOB	Yes	null	23	Body of WebService request for "Web Service" action
XITOR_TYPE_ID	NUMBER	Yes	null	12	Foreign key to "XITOR_TYPE" table
XML_SIZE	NUMBER	Yes	null	24	Size of XML/CSV file sent back to a client
ZONE_ID	NUMBER	Yes	null	13	Foreign key to "V_ZONE" table

Рис. 1. Структура USAGE_LOG

Постановка задачи. Есть лог из 3.1 млн. транзакций, выгруженный из системы и представленный в формате: `user_id \t un \t action_type \t page_name \t iis_finish \t iis_runtime \t user_runtime \t db_runtime \t response_delivery_time \t browser_runtime \n`. Пример записей из лога представлен на рис. 2.

Необходимо вычислить суммарное время, затраченное системой Trackor Onevizon, на работу с пользователем за месяц.

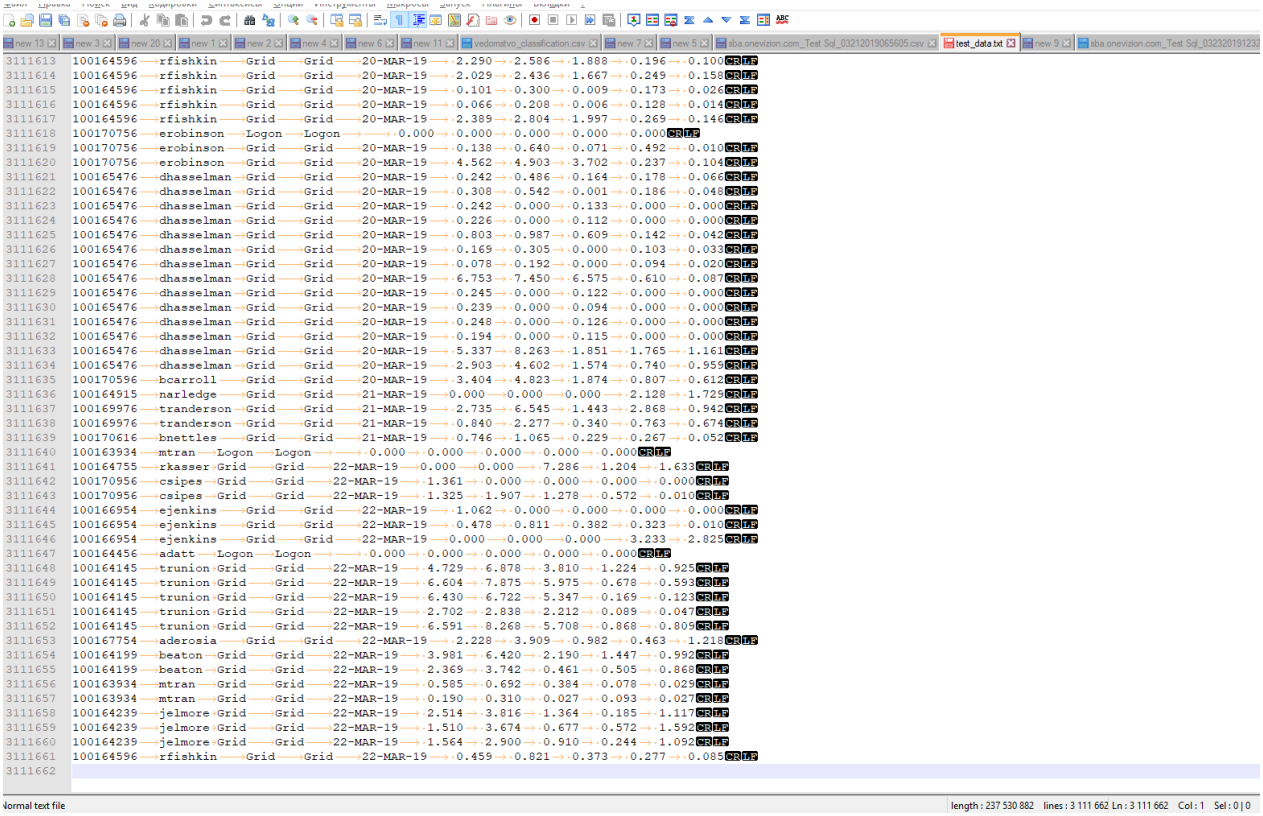

Рис. 2. Лог действий пользователя

Решение данной задачи стандартными методами, предусмотренными в системе Trackor Onevizion, а именно SQL запросом к базе данных, показанном на рис. 3, заняло 7 минут и 54 секунд, это приемлемый результат для такого объема записей, однако его можно улучшить, используя MapReduce.

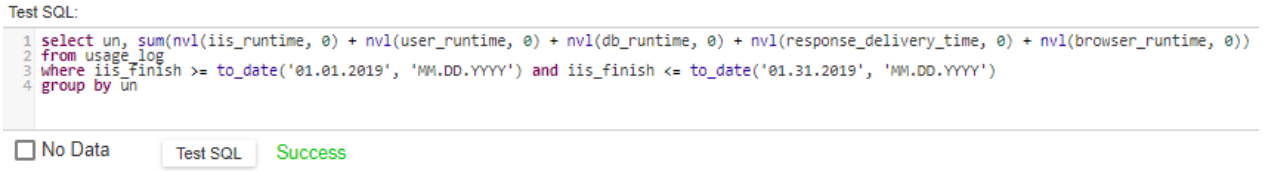

Рис. 3. Решение задачи стандартными методами

Теперь данную задачу, но уже используя алгоритм MapReduce. Данный алгоритм состоит из нескольких шагов [4]. Первый шаг – это применение функции `Map` к каждому элементу исходного лога. Функция возвращает либо ничего, либо создает объект, состоящий из пары значений `Key/Value`, в нашем случае `Key – un / Value – sum(iis_runtime + user_runtime + db_runtime + response_delivery_time + browser_runtime)`. Примеры таких объектов: `[adidenko/2.483]`, `[adidenko/12.238]`, `[rfishkin/12.493]` и так далее. Следующим шагом алгоритм сортирует все объекты, полученные на шаге map, с помощью метода `shuffle and sort`, а также создает новые экземпляры объектов, где все значения (`Value`) будут сгруппированы по ключу. Примерами таких

объектов служат: [adidenko: 2.483, 12.238, …], [rfishkin: 12.493, …] и так далее. Заключительным шагом будет выполнение функции Reduce для каждого из сгенерированных объектов. С ее помощью будут просуммированы все элементы Value объекта, и результаты ее работы возвращены в результирующую коллекцию в формате Key: Result. Все три шага изображены на рис. 4. Примеры результирующих объектов: [adidenko: 14.721], [rfishkin: 12.493] и так далее.

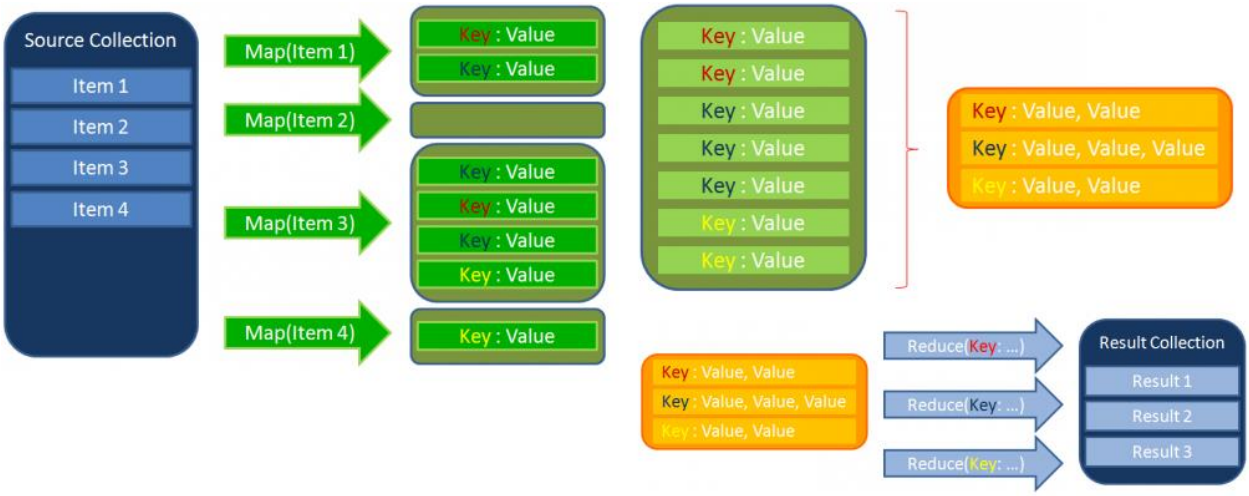

Рис. 4. Алгоритм MapReduce

Для реализации данного алгоритма был выбран язык программирования Python, а также был создан виртуальный кластер в программе Oracle VM VirtualBox.

Для решения данной задачи был написан код Reduce и 3 различных вариантов функции Map, а именно:

1) Mapper_datetime, где происходит вычленение необходимых строк по дате iis_finish и суммирование 5 значений iis_runtime, user_runtime, db_runtime, response_delivery_time, browser_runtime;

2) Mapper_alt_string, где определение строк проходит по части строки iis_finish, а также суммирование 5 значений iis_runtime, user_runtime, db_runtime, response_delivery_time, browser_runtime;

3) Mapper_2_phases_1 и Mapper_2_phases_2, где процесс разбит на 2 фазы а) поиска строк с нужными датами iis_finish и б) суммирования 5 значений iis_runtime, user_runtime, db_runtime, response_delivery_time, browser_runtime.

Код данных функций представлен на рис. 5, 6, 7, 8. Время выполнения и результаты решения задачи всеми 3 способами представлены на рис. 9.

reducer.py | test_data(1).txt | mapper_total_runtime.py | mapper_iis_runtime_date.py

```
#!/usr/bin/python

import sys

valuesTotal = 0
oldKey = None

# Loop around the data
# It will be in the format key\tval
# Where key is user name or category of user log action, val is the time in seconds

for line in sys.stdin:
    data_mapped = line.strip().split("\t")
    if len(data_mapped) != 2:
        # Something has gone wrong. Skip this line.
        continue

    thisKey, thisValue = data_mapped

    if oldKey and oldKey != thisKey:
        print oldKey, "\t", valuesTotal
        oldKey = thisKey;
        valuesTotal = 0

    oldKey = thisKey
    valuesTotal += float(thisValue)

if oldKey != None:
    print oldKey, "\t", valuesTotal
```

Рис. 5. Reducer.py

```
#!/usr/bin/python

# Format of each line is:
# user_id, un, action_type, page_name, date, iis_runtime, user_runtime, db_runtime, response_delivery_time,
browser_runtime
#
# We want elements 2 (user name) and 6, 7, 8, 9, 10 (iis_runtime, user_runtime, db_runtime,
response_delivery_time, browser_runtime) and 5 (date)
# We need to write them out to standard output, separated by a tab

import sys
from datetime import datetime

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 10:
        user_id, un, action_type, page_name, date, iis_runtime, user_runtime, db_runtime,
response_delivery_time, browser_runtime = data
        if len(date) > 0:
                datec = datetime.strptime(date, '%d-%b-%y')
                datemin = datetime.strptime('01-JAN-19', '%d-%b-%y')
                datemax = datetime.strptime('01-FEB-19', '%d-%b-%y')
                total_runtime = float(iis_runtime) + float(user_runtime) + float(db_runtime) + float
(response_delivery_time) + float(browser_runtime)
                if datemin <= datec < datemax:
                        print "{0}\t{1}".format(un, total_runtime)
```

Рис. 6. Mapper_datetime.py

```
#!/usr/bin/python

# Format of each line is:
# user_id, un, action_type, page_name, date, iis_runtime, user_runtime, db_runtime, response_delivery_time,
browser_runtime
#
# We want elements 2 (user name) and 6, 7, 8, 9, 10 (iis_runtime, user_runtime, db_runtime,
response_delivery_time, browser_runtime) and 5 (date)
# We need to write them out to standard output, separated by a tab

import sys

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 10:
        user_id, un, action_type, page_name, date, iis_runtime, user_runtime, db_runtime,
response_delivery_time, browser_runtime = data
        if len(date) > 0 and date[3:] == 'JAN-19':
                total_runtime = float(iis_runtime) + float(user_runtime) + float(db_runtime) + float
(response_delivery_time) + float(browser_runtime)
                print "{0}\t{1}".format(un, total_runtime)
```

Рис. 7. Mapper_alt_string.py

```
#!/usr/bin/python

# Format of each line is:
# user_id, un, action_type, page_name, date, iis_runtime, user_runtime, db_runtime, response_delivery_time,
browser_runtime
#
# We want elements 2 (user name) and 6, 7, 8, 9, 10 (iis_runtime, user_runtime, db_runtime,
response_delivery_time, browser_runtime) and 5 (date)
# We need to write them out to standard output, separated by a tab

import sys

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 10:
        user_id, un, action_type, page_name, date, iis_runtime, user_runtime, db_runtime,
response_delivery_time, browser_runtime = data
        if len(date) > 0 and date[3:] == 'JAN-19':
                print "{0}\t{1}\t{2}\t{3}\t{4}\t{5}".format(un, iis_runtime, user_runtime, db_runtime,
response_delivery_time, browser_runtime)
```

```
#!/usr/bin/python

# Format of each line is:
# un, iis_runtime, user_runtime, db_runtime, response_delivery_time, browser_runtime
#
# We want all elements
# We need to write them out to standard output, separated by a tab

import sys

for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 6:
        un, iis_runtime, user_runtime, db_runtime, response_delivery_time, browser_runtime = data
        total_runtime = float(iis_runtime) + float(user_runtime) + float(db_runtime) + float
(response_delivery_time) + float(browser_runtime)
        print "{0}\t{1}".format(un, total_runtime)
```

Рис. 8. Mapper_2_phases_1.py и Mapper_2_phases_2.py

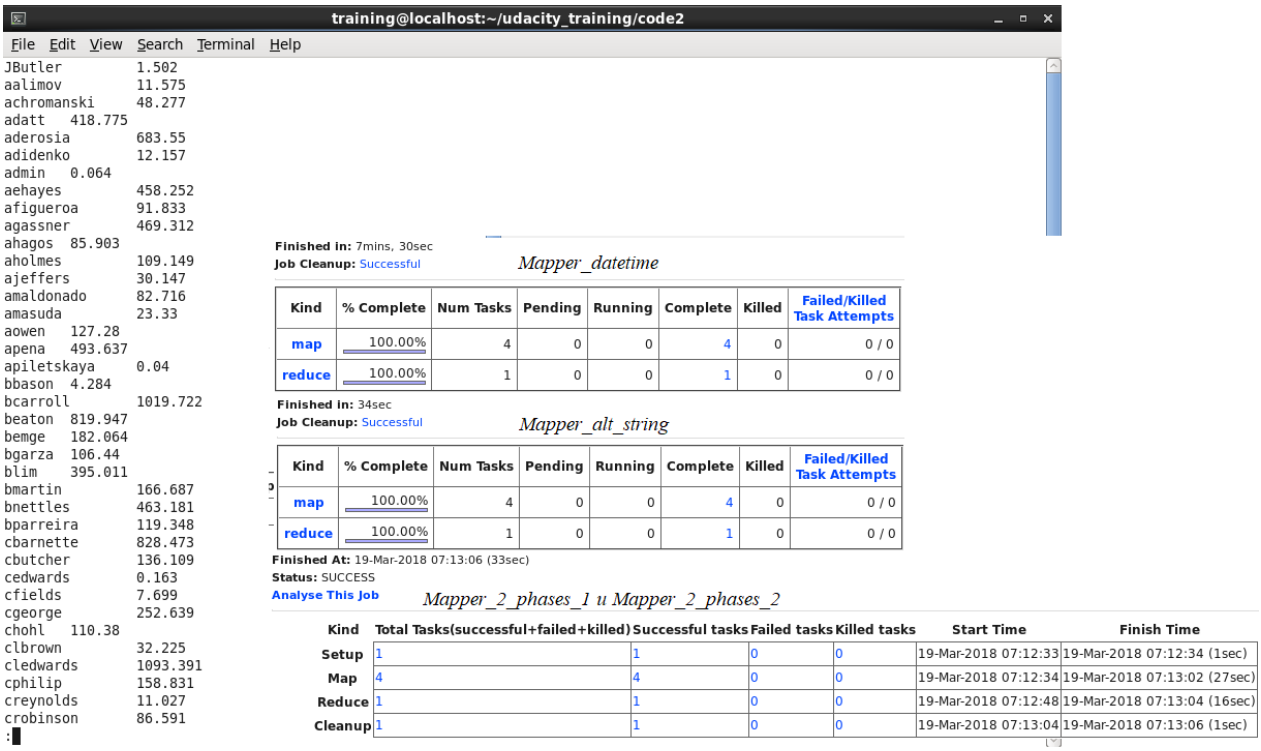

Рис. 9. Результаты решения задачи и время выполнения 3 различных подходов

Из статистики решения видно, что для решения каждого из вариантов задачи было создано четыре параллельные процесса `map` и один – `reduce`. Самым эффективным по времени выполнения маппером себя показал двухфазовый с 33 секундами, затем идет Mapper_alt_string с 34 секундами, и наименее эффективным себя показал Mapper_datetime с 7 минутами и 30 секундами. Такие плохие показатели у последней функции вызваны конвертациями строк в даты, поэтому без острой необходимости такие операции лучше не использовать.

Заключение. В данной статье был рассмотрен алгоритм MapReduce на примере решения задачи вычисления суммарного времени, затраченного системой Trackor Onevizon, на работу с пользователем за месяц с помощью Apache Hadoop на кластере, поднятом в виртуальной машине. В результате было наглядно продемонстрировано, что для данного лога и конкретного компьютера алгоритм MapReduce в самой оптимальной конфигурации в 15 раз эффективнее по времени, чем стандартное решение, написанное на SQL, но при использовании в маппере функций конвертаций строк в даты время выполнения задачи с помощью алгоритма MapReduce сравнимо со стандартным решением.

Список литературы:

1. Википедия MapReduce [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/MapReduce, свободный (дата обращения 27.03.2019).
2. Википедия Hadoop [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Hadoop, свободный (дата обращения: 27.03.2019).
3. Графический компонент создания и редактирования пользовательских форм для платформы Onevizion [Электронный ресурс]. – Режим доступа: http://science.kuzstu.ru/wp-content/Events/Conference/RM/2018/RM18/pages/Articles/31547-.pdf, свободный (дата обращения: 27.03.2019).
4. Что такое MapReduce? // Статья блога о разработке, программировании на C#/.NET, и не только [Электронный ресурс]. – Режим доступа: http://regfordev.blogspot.ru/2015/09/mapreduce.html#.WrQJDOjFKUm, свободный (дата обращения 27.03.2019).