

A Quantitative Study of Two Matrix Clustering Algorithms

Александр Слесарев
студент СПбГУ, 2-й курс

Вячеслав Галактионов, Никита Бобров, Георгий Чернышев
СПбГУ, JetBrains Research

SEIM 2019
13 апреля 2019

Типы фрагментирования

- ▶ Горизонтальное
- ▶ Вертикальное
- ▶ Гибридное

Число возможных фрагментов

Число Белла – число всех неупорядоченных разбиений n -элементного множества

При больших n выполняется $B(n) \approx n^n$,
например, $B(30) \approx 10^{23}$

n	$B(n)$
1	1
2	1
3	2
4	5
5	15
6	52
7	203
8	877
9	4140
10	21147
11	115975

Виды вертикального фрагментирования

- ▶ стоимостное
- ▶ эвристическое
 - ▶ методы матричной кластеризации
 - ▶ графовый подход
 - ▶ data mining

О проекте

Цель: экспериментальная проверка алгоритмов ВФ на основе подхода матричной кластеризации

Современные работы по ВФ на матричной кластеризации:

- ▶ C. Cheng “Algorithms for vertical partitioning in database physical design”, **1993**
- ▶ C.-H. Cheng “A branch and bound clustering algorithm”, **1995**
- ▶ C.-H. Cheng and J. Motwani “An examination of cluster identification-based algorithms for vertical partitions”, **2009**
- ▶ C.-H. Cheng et al. “An improved branch-and-bound clustering approach for data partitioning”, **2011**

Наши предыдущие работы:

- ▶ V. Galaktionov et al. “Matrix clustering algorithms for vertical partitioning problem: an initial performance study”, **2016**
- ▶ V. Galaktionov “Parallelization of matrix clustering algorithms”, **2016**
- ▶ V. Galaktionov et al. “A study of several matrix-clustering vertical partitioning algorithms in a disk-based environment”, **2017**

Матрица запросов

q1: SELECT a FROM TABLE WHERE a > 10;

q2: SELECT b, f FROM TABLE;

q3: SELECT a, c FROM TABLE WHERE a = c;

q4: SELECT a FROM TABLE WHERE a < 10;

q5: SELECT e FROM TABLE;

q6: SELECT d, e FROM TABLE WHERE d + e > 0;

	a	b	c	d	e	f
q ₁	1	0	0	0	0	0
q ₂	0	1	0	0	0	1
q ₃	1	0	1	0	0	0
q ₄	1	0	0	0	0	0
q ₅	0	0	0	0	1	0
q ₆	0	0	0	1	1	0



	a	c	b	f	d	e
q ₁	1	0	0	0	0	0
q ₃	1	1	0	0	0	0
q ₄	1	0	0	0	0	0
q ₂	0	0	1	1	0	0
q ₆	0	0	0	0	1	1
q ₅	0	0	0	0	0	1

Cluster identification

$$\begin{array}{c}
 \begin{array}{cccccc}
 & 2 & 3 & 4 & 6 & 1 & 5 \\
 1 & \left[\begin{array}{cccccc}
 0 & 1 & 1 & 1 & 0 & 0 \\
 1 & 0 & 0 & 0 & 1 & 0 \\
 1 & 0 & 0 & 0 & 1 & 0 \\
 0 & 0 & 1 & 1 & 0 & 1 \\
 0 & 1 & 1 & 1 & 0 & 1
 \end{array} \right]
 \end{array}
 \end{array}$$

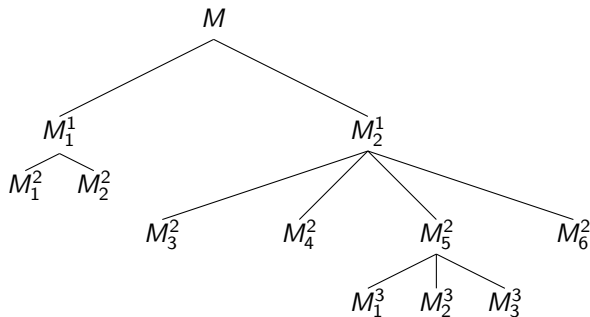
$$\begin{array}{c}
 \begin{array}{cccccc}
 1 & 2 & 3 & 4 & 5 & 6 \\
 \left[\begin{array}{cccccc}
 0 & 1 & 1 & 1 & 0 & 0 \\
 1 & 0 & 0 & 0 & 1 & 0 \\
 1 & 0 & 0 & 0 & 1 & 0 \\
 0 & 0 & 1 & 1 & 0 & 1 \\
 0 & 1 & 1 & 1 & 0 & 1
 \end{array} \right]
 \end{array}
 \longrightarrow
 \begin{array}{c}
 \begin{array}{cccccc}
 1 & 2 & 3 & 4 & 5 & 6 \\
 \left[\begin{array}{cccccc}
 0 & 1 & 1 & 1 & 0 & 0 \\
 1 & 0 & 0 & 0 & 1 & 0 \\
 1 & 0 & 0 & 0 & 1 & 0 \\
 0 & 0 & 1 & 1 & 0 & 1 \\
 0 & 1 & 1 & 1 & 0 & 1
 \end{array} \right]
 \end{array}
 \longrightarrow
 \begin{array}{c}
 \begin{array}{cccccc}
 1 & 2 & 3 & 4 & 5 & 6 \\
 \left[\begin{array}{cccccc}
 0 & 1 & 1 & 1 & 0 & 0 \\
 1 & 0 & 0 & 0 & 1 & 0 \\
 1 & 0 & 0 & 0 & 1 & 0 \\
 0 & 0 & 1 & 1 & 0 & 1 \\
 0 & 1 & 1 & 1 & 0 & 1
 \end{array} \right]
 \end{array}
 \end{array}$$

$$\begin{array}{c}
 \begin{array}{cccccc}
 & 2 & 3 & 4 & 6 & 1 & 5 \\
 1 & \left[\begin{array}{cccccc}
 1 & 1 & 1 & 0 & 0 & 0 \\
 0 & 1 & 1 & 1 & 0 & 0 \\
 1 & 1 & 1 & 1 & 0 & 0 \\
 0 & 0 & 0 & 0 & 1 & 1 \\
 0 & 0 & 0 & 0 & 1 & 1
 \end{array} \right]
 \end{array}
 \end{array}$$

Поиск решения – 1

M – матрица запросов

M_i^j – i -й узел на j -ом уровне фрагментирования



Поиск решения – 2

R - множество индексов
запросов M

C - множество индексов
атрибутов M

$$cohesion(M) = \frac{|\{a_{ij} = 1, i \in R \wedge j \in C\}|}{|R| * |C|}$$

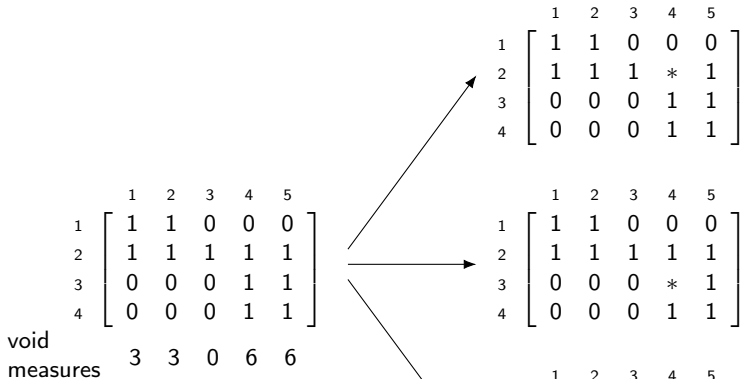
Набор кластеров - решение, если каждый кластер S удовлетворяет условиям:

- ▶ $cohesion(S) < threshold$
- ▶ В S отсутствуют нулевые строки или столбцы

Метод ветвей и границ 2009 – 1

нижняя граница: Z_L – число единиц, удаленных из матрицы запросов в ходе ветвления

верхняя граница: Z_U – минимальный Z_L среди найденных решений



Метод ветвей и границ 2009 – 2

Распределение межкластерных элементов по матрицам в решении:

- ▶ (separate) составить отдельный фрагмент из межкластерных элементов
- ▶ (nearest) добавить в каждый фрагмент только необходимые межкластерные элементы
- ▶ (replicate) добавить в каждый фрагмент все необходимые межкластерные элементы

Метод ветвей и границ 2011 – 1

нижняя граница: Z_L – глубина узла в дереве

верхняя граница: Z_U – минимальный Z_L среди найденных решений

$$\begin{array}{c}
 1 \\
 2 \\
 3 \\
 4
 \end{array}
 \begin{bmatrix}
 1 & 2 & 3 & 4 & 5 \\
 1 & 1 & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 1 \\
 0 & 0 & 0 & 1 & 1 \\
 0 & 0 & 0 & 1 & 1
 \end{bmatrix}$$

void
measures 3 3 0 6 6

$$\begin{array}{c}
 1 \\
 2 \\
 3 \\
 4
 \end{array}
 \begin{bmatrix}
 1 & 2 & 3 & 4 & 4' & 4'' & 5 \\
 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
 0 & 0 & 0 & 0 & 0 & 1 & 1
 \end{bmatrix}$$

$$\begin{array}{c}
 1 \\
 2 \\
 3 \\
 4
 \end{array}
 \begin{bmatrix}
 1 & 2 & 3 & 4 & 5 & 5' & 5'' \\
 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
 0 & 0 & 0 & 1 & 0 & 0 & 1
 \end{bmatrix}$$

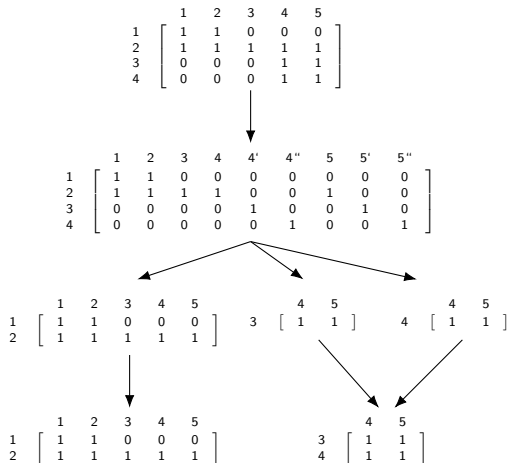
$$\begin{array}{c}
 1 \\
 2 \\
 3 \\
 4
 \end{array}
 \begin{bmatrix}
 1 & 1' & 2 & 3 & 4 & 5 \\
 1 & 0 & 1 & 0 & 0 & 0 \\
 0 & 1 & 1 & 1 & 1 & 1 \\
 0 & 0 & 0 & 0 & 1 & 1 \\
 0 & 0 & 0 & 0 & 1 & 1
 \end{bmatrix}$$

$$\begin{array}{c}
 1 \\
 2 \\
 3 \\
 4
 \end{array}
 \begin{bmatrix}
 1 & 2 & 2' & 3 & 4 & 5 \\
 1 & 1 & 0 & 0 & 0 & 0 \\
 1 & 0 & 1 & 1 & 1 & 1 \\
 0 & 0 & 0 & 0 & 1 & 1 \\
 0 & 0 & 0 & 0 & 1 & 1
 \end{bmatrix}$$

$$\begin{array}{c}
 1 \\
 2 \\
 3 \\
 4
 \end{array}
 \begin{bmatrix}
 1 & 2 & 3 & 4 & 5 \\
 1 & 1 & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 1 \\
 0 & 0 & 0 & 1 & 1 \\
 0 & 0 & 0 & 1 & 1
 \end{bmatrix}$$

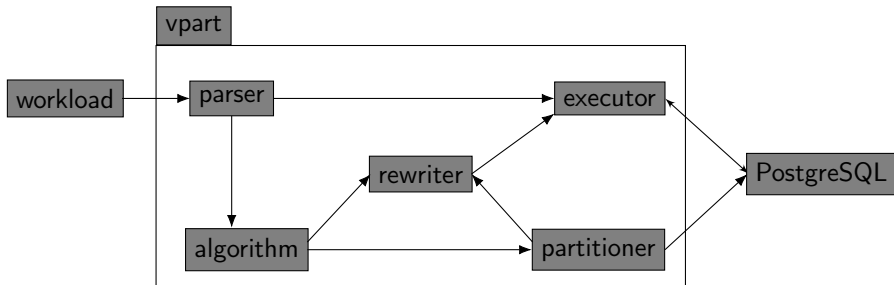
Метод ветвей и границ 2011 – 2

Постобработка решения



Тестовый стенд

Структура программы



Критерии оценки алгоритмов :

- ▶ скорость матричной кластеризации
- ▶ скорость выполнения запросов после применения алгоритма
- ▶ затраты памяти на хранение фрагментов

Реализация

Аппаратные средства :

- ▶ Inspiron 15 7000 Gaming (0798)
- ▶ 8GiB RAM
- ▶ Intel(R) Core(TM) i5-7300HQ
CPU @ 2.50GHz
- ▶ TOSHIBA 1TB MQ02ABD1

Программное обеспечение :

- ▶ Ubuntu 18.10
- ▶ PostgreSQL 11.1
- ▶ gcc 8.2.0

Датасет :

- ▶ Sloan Digital Sky Survey, "Star" table

Обеспечение чистоты экспериментов

$$\sum_{q_i \in \text{Queries}} \frac{\text{size}(T)}{\text{time}(q_i, T)} = \sum_{q_i \in \text{Queries}} \frac{\text{size}(\text{table}(q_i))}{\text{time}(q_i, \text{table}(q_i))}$$

$\text{size}(T)$ – размер исходной таблицы в байтах

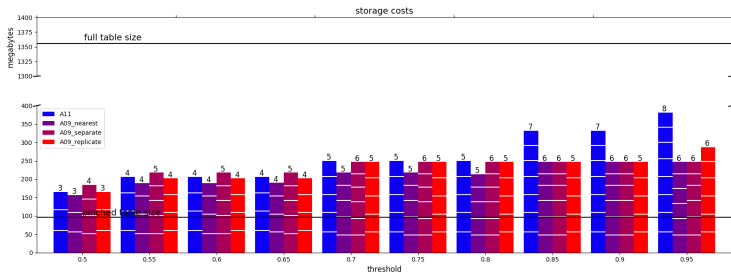
$\text{size}(\text{table}(q_i))$ – размер фрагмента, соответствующего запросу

$\text{time}(q_i, \text{table})$ – время выполнения запроса к соответствующей таблице

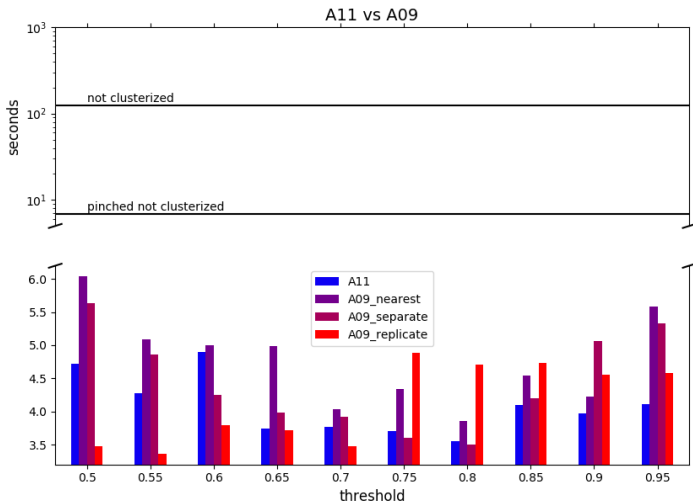
- ▶ сбросить системный кеш: установить флаг 3 в `/proc/sys/vm/drop_caches`
- ▶ запретить параллельное выполнение запросов:
`set max_parallel_workers_per_gather to 0 ;`

Эксперимент 1 – затраты памяти на фрагменты

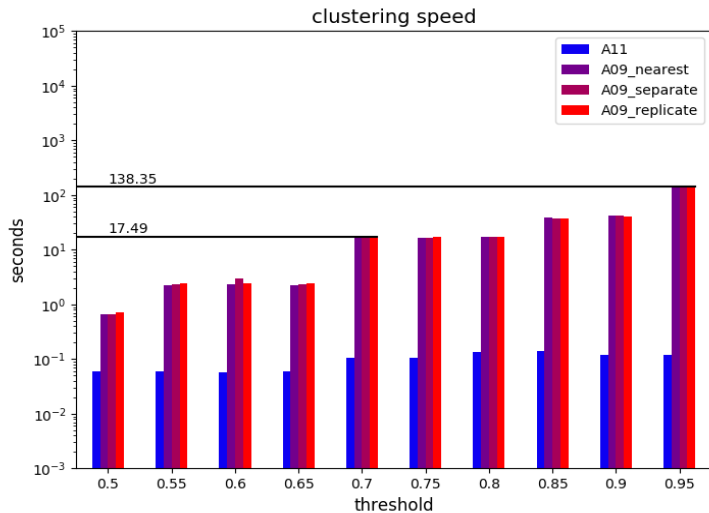
- ▶ база данных SDSS-IV Data Release 14, 2016
- ▶ 8 запросов
- ▶ таблица "Star"
- ▶ 509 атрибутов, 492515 записей
- ▶ каждое показание является средним значением из 10 итераций



Эксперимент 1 – скорость выполнения запросов

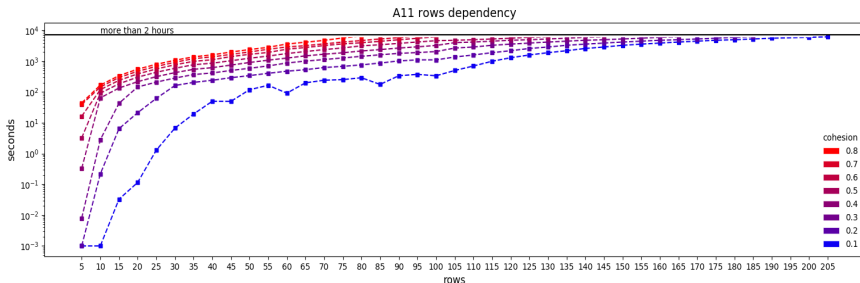


Эксперимент 1 – скорость получения фрагментов

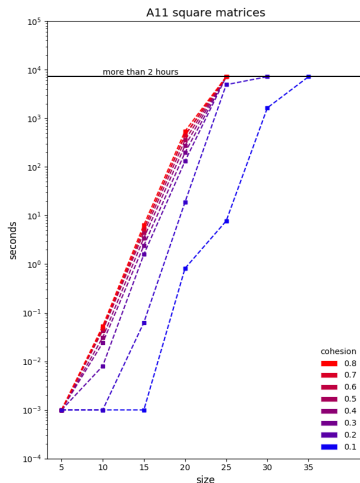
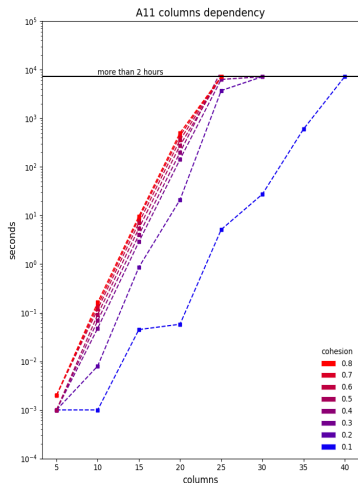


Эксперимент 2 – матрицы с фиксированным числом столбцов

- ▶ синтетические матрицы
- ▶ варьирование плотности и размера
- ▶ $threshold = 0.9$
- ▶ предел времени выполнения 2 часа



Эксперимент 2 – матрицы с фиксированным числом строк и квадратные матрицы



Выводы – 1

- ▶ Фрагментирование повышает производительность
- ▶ Качество фрагментов зависит от threshold. С этой точки зрения лучше всего работает A09 с replication.
- ▶ Топ-5 по скорости запросов к фрагментам:

время	алгоритм	стратегия	threshold
3.358	A09	replicate	0.55
3.472	A09	replicate	0.55
3.479	A09	replicate	0.70
3.500	A09	separate	0.80
3.553	A11		0.80

Выводы – 2

- ▶ При $\text{threshold} = 0.7$ суммарное время работы всех алгоритмов наименьшее.
- ▶ С данным датасетом A11 сработал почти в 10 раз быстрее, чем A09 вне зависимости от стратегии.
- ▶ Возрастание числа атрибутов замедляет работу алгоритмов быстрее, чем возрастание числа запросов.
- ▶ Наборы фрагментов, производимые алгоритмами, требуют в 1,5 – 2 раза больше памяти, чем таблица, состоящая только из используемых атрибутов.