ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ ДАННЫХ В СУБД

05.13.11 – математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Диссертация на соискание ученой степени доктора физико-математических наук

Михаил Леонидович Цымблер

Научный консультант:
Соколинский Леонид Борисович,
доктор физ.-мат. наук, профессор

Цель диссертационной работы

Разработка комплекса масштабируемых методов и параллельных алгоритмов для создания программной платформы интеллектуального анализа данных средствами СУБД с открытым кодом

Актуальность исследования

1. Приближение алгоритмов к данным
   - технологии СУБД для больших данных
   - интеграция анализа данных в СУБД
2. Импортозамещение
   - российские СУБД и системы анализа данных
   - решения на основе СУБД с открытым кодом

Основные задачи

1. Разработать методы и алгоритмы для внедрения параллелизма в последовательную СУБД с открытым исходным кодом
2. Разработать, реализовать и исследовать комплекс параллельных алгоритмов интеллектуального анализа данных на основе параллельной СУБД

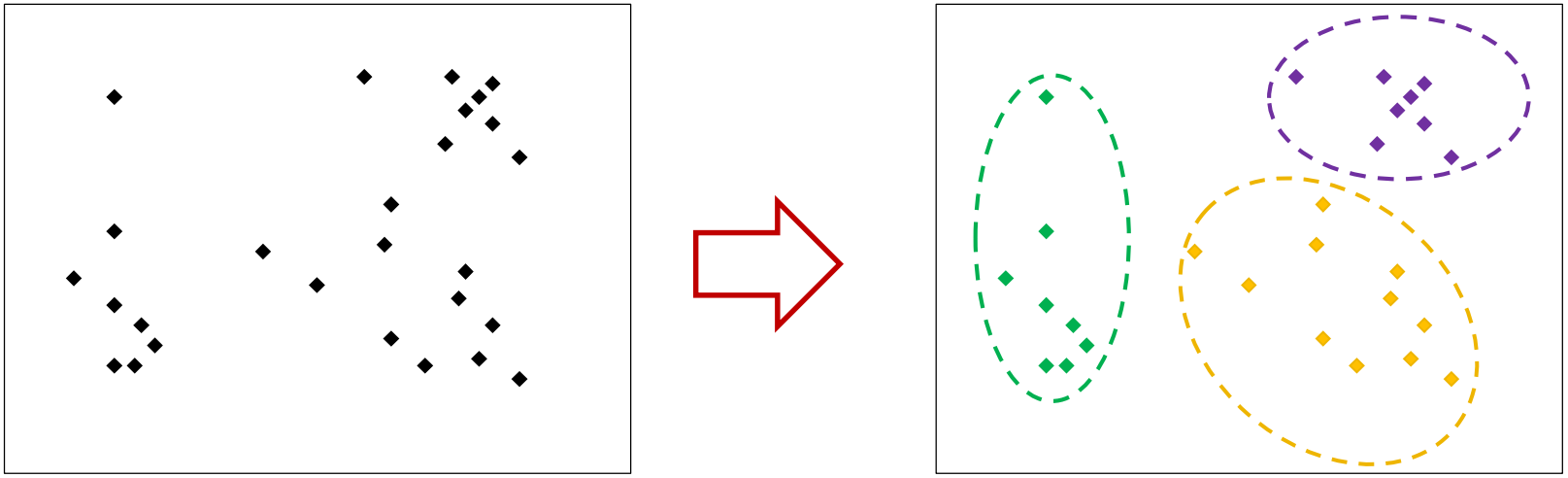
Работы по теме диссертации

Hellerstein J.M., Re C., Schoppmann F., *et al.* The MADlib analytics library or MAD skills, the SQL. PVLDB. 2012. Vol. 5, No. 12. P. 1700–1711.	Пионерская работа по внедрению алгоритмов анализа данных в СУБД
Ordonez C. Can we analyze big data inside a DBMS? Proc. of the 16th Int. Workshop on Data warehousing and OLAP (DOLAP 2013). 2013. P. 85–92.	Анализ больших данных в последовательных СУБД
Abadi D., Agrawal R., Ailamaki A., *et al.* The Beckman report on database research. Commun. ACM. 2016. Vol. 59, No. 2. P. 92–99.	Приближение алгоритмов к данным в СУБД
Mahajan D., Kim J.K., *et al.* In-RDBMS hardware acceleration of advanced analytics. PVLDB. 2018. Vol. 11, No. 11. P. 1317–1331.	Использование FPGA для анализа данных в СУБД
Ghadiri N., Ghaffari M., Nikbakht M.A. BigFCM: Fast, precise and scalable FCM on Hadoop. Future Gen. Comp. Syst. 2017. Vol. 77. P. 29–39.	Параллельная нечеткая кластеризация данных на платформе Hadoop
Zeng Z., Wu B., Wang H. A parallel graph partitioning algorithm to speed up the large-scale distributed graph mining. Proc. of the Int. Workshop on Big Data (BigMine 2012). 2012. P. 61–68.	Параллельная кластеризация больших графов на основе поиска сообществ
Shabib A., Narang A., Niddodi C.P. *et al.* Parallelization of searching and mining time series data using Dynamic Time Warping. Int. Conf. on Advances in Computing, Communications and Informatics (ICACCI 2015). 2015. P. 343–348.	Поиск похожих подпослед. временного ряда на кластере с многоядерными процессорами
Yankov D., Keogh E.J., Rebbapragada U. Disk aware discord discovery: finding unusual time series in terabyte sized datasets. Knowl. Inf. Syst. 2008. Vol. 17, No. 2. P. 241–262.	Параллельный поиск аномалий временного ряда в парадигме MapReduce

Комплекс алгоритмов интеллектуального анализа данных

- Алгоритмы кластеризации данных
  - Кластеризация графов
  - Нечеткая кластеризация
  - Кластеризация данных с шумами и выбросами
- Алгоритмы анализа временных рядов
  - Поиск похожих подпоследовательностей
  - Поиск аномальных подпоследовательностей
- Алгоритмы поиска шаблонов

Задача кластеризации

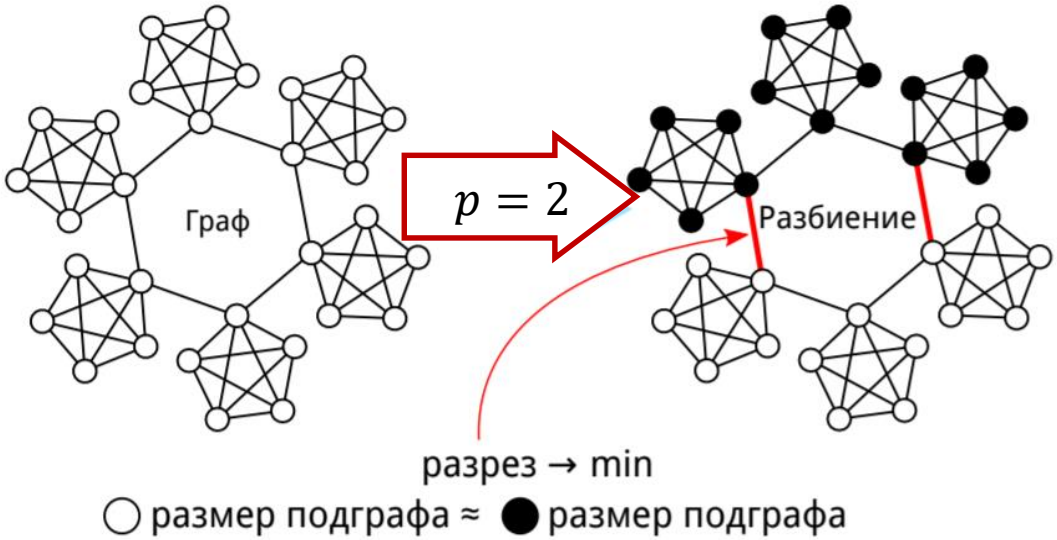


Множество объектов	$X = \{x_i \mid 1 \le i \le n, x_i \in \mathbb{R}^d\}$
Функция расстояния	$\rho(x_i, x_j)$
Количество кластеров	$k, 1 < k \ll n$
Метки кластеров	$C = \{c_i \mid 1 \le i \le k, c_i \in \mathbb{N}\}$
Функция кластеризации	$\alpha: X \to C$

Параллельные алгоритмы кластеризации, разработанные в диссертации

- Кластеризация графа
- Нечеткая кластеризация данных
- Кластеризация на основе техники медоидов

Кластеризация графа



Граф $G(N, E, w)$

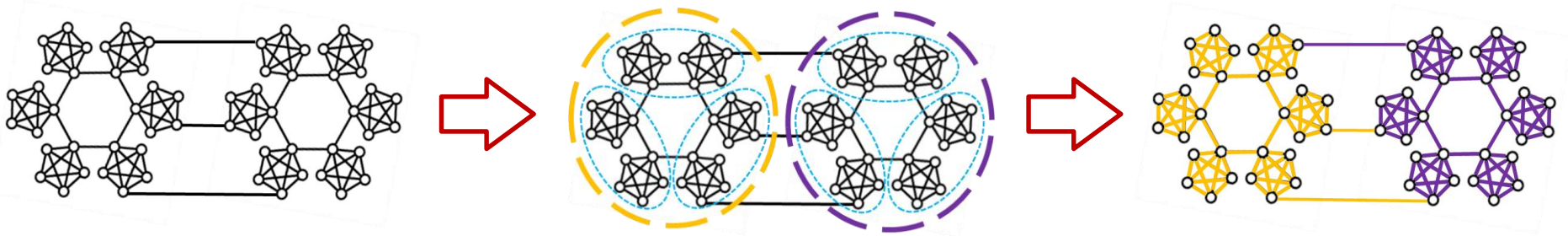
1. $N = \cup_{i=1}^{p} N_i,\ \forall i \neq j\ N_i \cap N_j = \emptyset,\ p > 1$
2. $w(N_i) \approx \frac{w(N)}{p}\ \ \forall i \in \{1, \dots, p\}$
3. *Разрез* $W_{cut} \to \min$, $W_{cut} := \sum_{e \in E_{cut}} w(e)$,
   $E_{cut} := \{(u, v) \in E \mid u \in N_i, v \in N_j, 1 \leq i, j \leq p, i \neq j\}$

Алгоритм dbParGraph

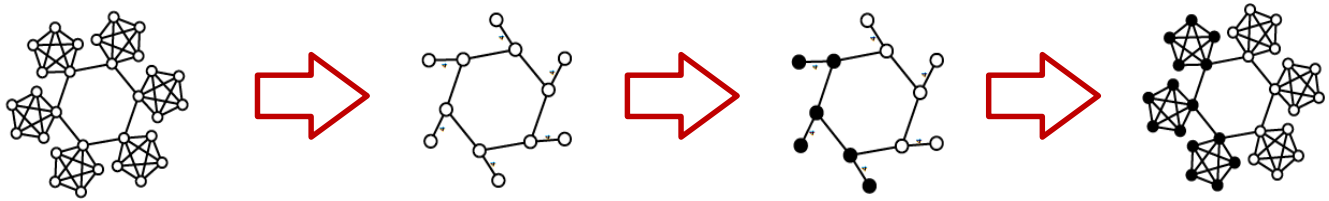
- Область применения
  - Кластеризация сверхбольших социальных графов
- Платформа
  - Параллельная СУБД для кластерной системы

Структура алгоритма dbParGraph

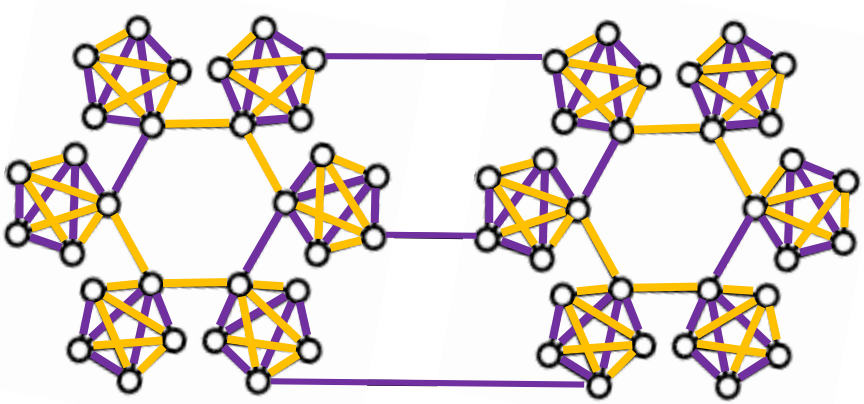
1. Поиск и консолидация сообществ



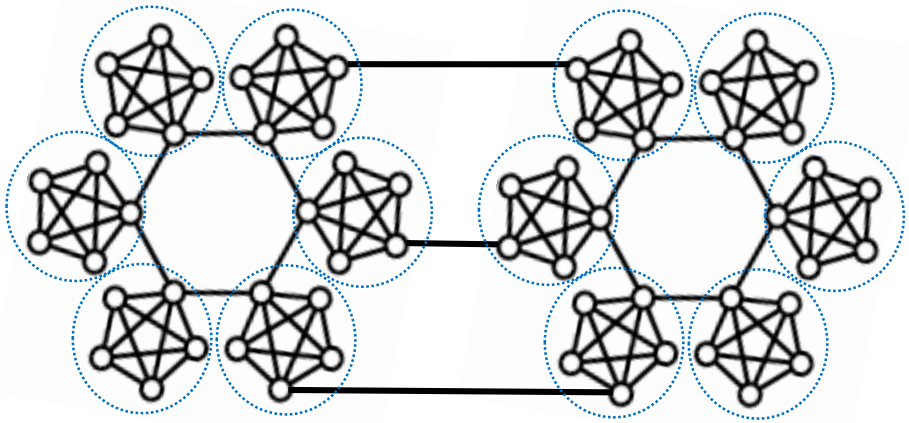
2. Многоуровневое разбиение



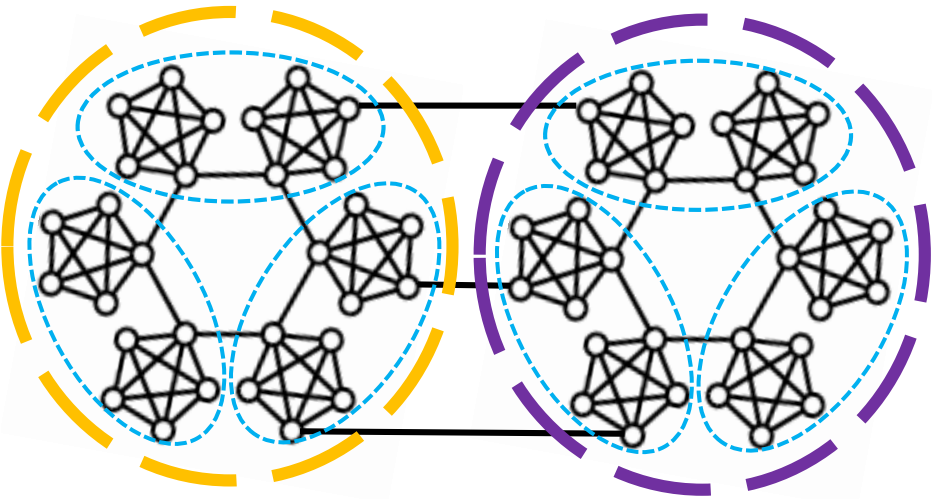
Поиск и консолидация сообществ в графе

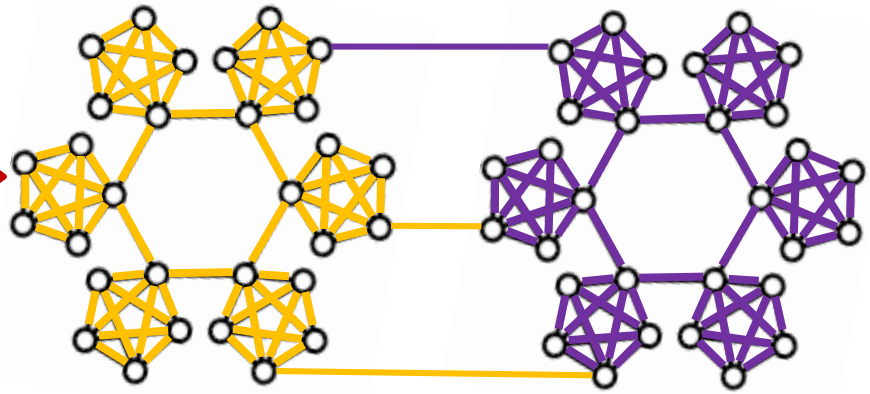


1-й сервер, 2-й сервер



$$\boldsymbol{affinity}(v,u) = \frac{w(v,u)}{\sum_{i \in \mathcal{N}_v} w(v,i)} \quad \boldsymbol{d}(v,C) = \frac{\sum_{u \in \mathcal{N}_v \wedge \mathcal{L}_u = C} affinity(v,u)}{\sum_{u \in \mathcal{N}_v} affinity(v,u)}$$

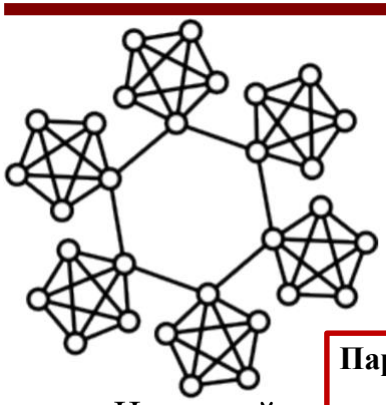




1-й сервер

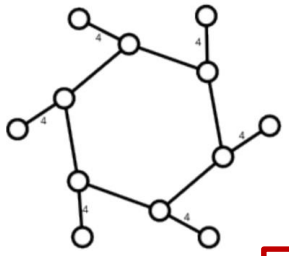
2-й сервер

Многоуровневое разбиение



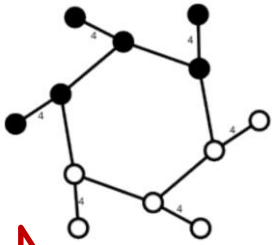
Исходный граф

Параллельная СУБД
Огрубление



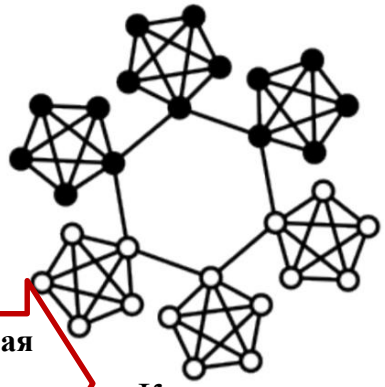
Огрубленный граф

Алгоритм в оперативной памяти
Нач. разбиение



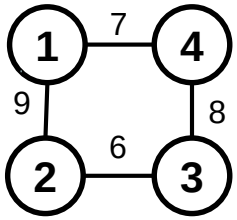
Грубое разбиение

Параллельная СУБД
Уточнение

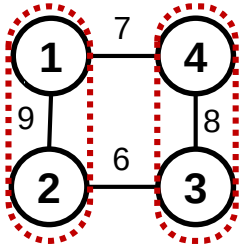


Кластери-зованный граф

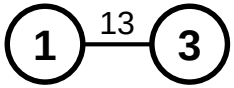
A	B	W
1	2	9
2	3	6
3	4	8
4	1	7



A	B
1	2
3	4



A	B	W
1	3	13

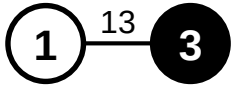


$$g(v) := ext(v) - int(v)$$

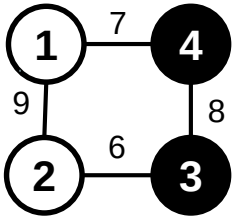
$$ext(v) := \sum_{(v,u)\in E, P(v)\neq P(u)} w(v,u)$$

$$int(v) := \sum_{(v,u)\in E, P(v)=P(u)} w(v,u)$$

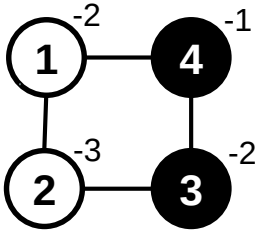
A	P
1	**1**
3	**0**



A	P
1	**1**
2	**1**
3	**0**
4	**0**



A	P	G
1	**1**	-2
2	**1**	-3
3	**0**	-2
4	**0**	-1



Сравнение с аналогами

Алгоритм	Граф		***dbPargraph* (ускорение)**
	$\|N\|$	$\|E\|$	
MSP [1)]	10^7	$5 \cdot 10^7$	**10%**
ParMETIS [2)]	$7.7 \cdot 10^6$	$1.33 \cdot 10^8$	**13%**
PT-Scotch [3)]	$2.3 \cdot 10^7$	$1.75 \cdot 10^8$	**27%**

[1)] Zeng Z., *et al.* A parallel graph partitioning algorithm to speed up the large-scale distributed graph mining. BigMine 2012. pp. 61–68.

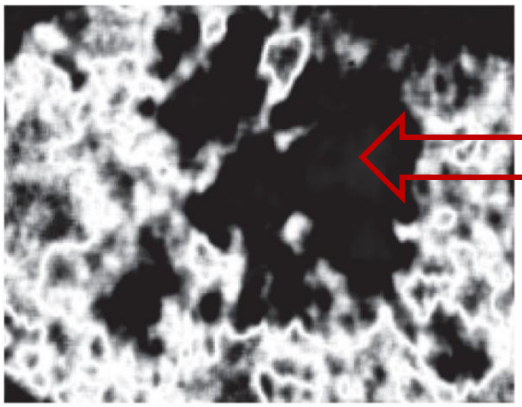
[2)] Karypis G. METIS and ParMETIS. Enc. of Parallel Computing (Ed. by D.A. Padua). Springer, 2011. pp. 1117–1124.

[3)] Chevalier C., Pellegrini F. PT-Scotch: A tool for efficient parallel graph ordering. Parallel Computing. 2008. vol. 34, no. 6-8. pp. 318–331.

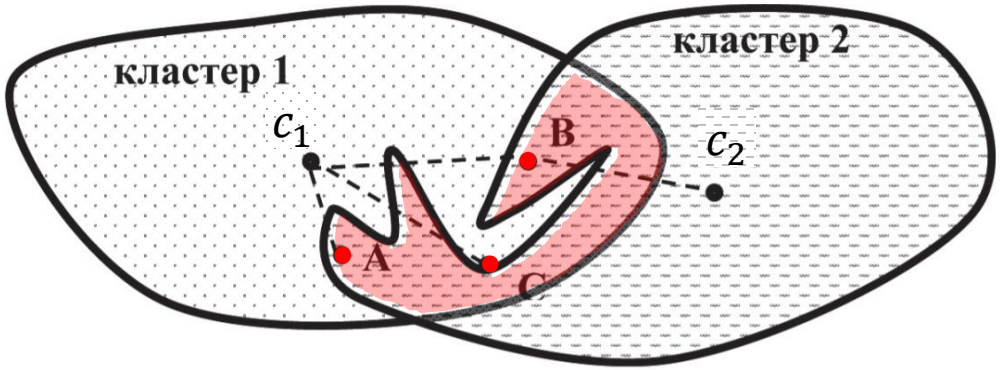
Параллельные алгоритмы кластеризации, разработанные в диссертации

- Кластеризация графа
- **Нечеткая кластеризация данных**
- Кластеризация на основе техники медоидов

Нечеткая кластеризация данных



Радиологическое изображение



Сегментация радиологического изображения

- $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ – матрица объектов
- $k, 1 < k \ll n$ – количество кластеров
- $\boldsymbol{C} \in \mathbb{R}^{k \times d}$ – матрица центроидов
- $\boldsymbol{U} \in \mathbb{R}^{n \times k}$ – матрица принадлежности объектов кластерам:

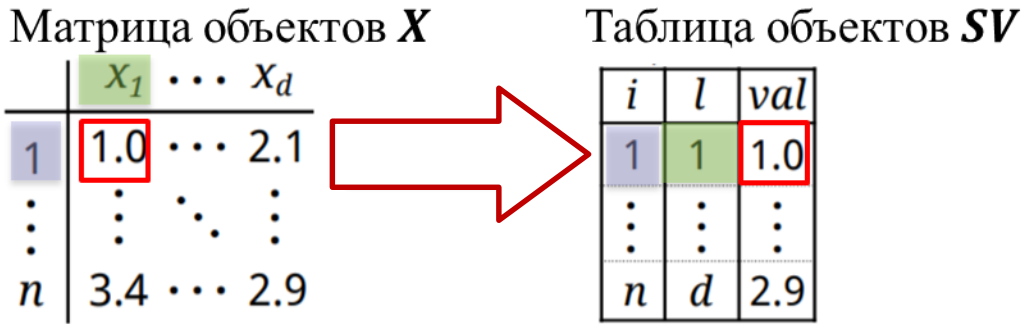
$$0 \leq u_{ij} \leq 1, \sum_{j=1}^{k} u_{ij} = 1$$

Алгоритм pgFCM

- Область применения
  - Нечеткая кластеризация сверхбольших данных
- Платформа
  - Параллельная СУБД для кластерной системы

Структура алгоритма pgFCM

1. «Индексное» представление данных



2. Использование агрегирующих SQL-запросов

```
INSERT INTO C
  SELECT R.j, SV.l, sum(R.s * SV.val) / sum(R.s) AS val
  FROM (
    SELECT i, j, U.val^m AS s
    FROM U) AS R, SV
  WHERE R.i = SV.i
  GROUP BY j, l;
```

Индексное представление данных

Матрица объектов $\boldsymbol{X}$

	x_1	$\cdots$	x_d
1	1.0	$\cdots$	2.1
$\vdots$	$\vdots$	$\ddots$	$\vdots$
n	3.4	$\cdots$	2.9

Таблица объектов $\boldsymbol{SV}$

i	l	val
1	1	1.0
$\vdots$	$\vdots$	$\vdots$
n	d	2.9

Таблица объектов $\boldsymbol{SH}$

i	x_1	$\cdots$	x_d
1	1.0	$\cdots$	2.1
$\vdots$	$\vdots$	$\ddots$	$\vdots$
n	3.4	$\cdots$	2.9

Матрица центроидов $\boldsymbol{C}$

	x_1	$\cdots$	x_d
1	2.2	$\cdots$	8.1
$\vdots$	$\vdots$	$\ddots$	$\vdots$
k	3.4	$\cdots$	6.9

Таблица центроидов $\boldsymbol{C}$

j	l	val
1	1	2.2
$\vdots$	$\vdots$	$\vdots$
k	d	6.9

Матрица принадлежности $\boldsymbol{U}$

	1	$\cdots$	k
1	0.2	$\cdots$	0.1
$\vdots$	$\vdots$	$\ddots$	$\vdots$
n	0.8	$\cdots$	0.1

Таблица $\boldsymbol{U}$ (шаг s)

i	j	val
1	1	0.2
$\vdots$	$\vdots$	$\vdots$
n	k	0.1

Таблица $\boldsymbol{UT}$ (шаг $s+1$)

i	j	val
1	1	0.2
$\vdots$	$\vdots$	$\vdots$
n	k	0.1

Использование агрегирующих SQL-запросов

Алгоритм pgFCM

Создать индексное представление: таблицы $\boldsymbol{U}$,$\boldsymbol{SV}$,$\boldsymbol{C}$,$\boldsymbol{UT}$

Создать таблицу $\boldsymbol{SD}$

повторять

Вычислить центроиды c_j и обновить $\boldsymbol{C}$

```
INSERT INTO C
  SELECT R.j, SV.l, sum(R.s * SV.val) / sum(R.s) AS val
  FROM (
    SELECT i, j, U.val^m AS s
    FROM U) AS R, SV
  WHERE R.i = SV.i
  GROUP BY j, l;
```

Вычислить расстояния $\rho(x_i, c_j)$ и обновить $\boldsymbol{SD}$

```
INSERT INTO SD
  SELECT i, j, sqrt(sum((SV.val - C.val)^2)) as dist
  FROM SV, C
  WHERE SV.l = C.l
  GROUP BY i, j;
```

$\boldsymbol{U} := \boldsymbol{UT}$

Обновить $\boldsymbol{UT}$

```
INSERT INTO UT
  SELECT i, j, SD.dist^(2.0^(1.0 - m)) * SD1.den AS val
  FROM (
    SELECT i, 1.0 / sum(dist^(2.0^(m - 1.0))) AS den
    FROM SD
    GROUP BY i) AS SD1, SD
  WHERE SD.i = SD1.i;
```

пока $\max_{ij}\{|ut_{ij} - u_{ij}|\} > \varepsilon$

```
SELECT max(abs(UT.val - U.val))
FROM U, UT
WHERE U.i = UT.i AND U.j = UT.j;
```

выдать таблицы $\boldsymbol{U}, \boldsymbol{C}$

Сравнение с аналогами

Алгоритм	Данные		***pgFCM* (ускорение)**
	n	d	
WCFC [1)]	$4.9 \cdot 10^6$	41	**2%**
BigFCM [2)]	$1.1 \cdot 10^7$	28	**15%**

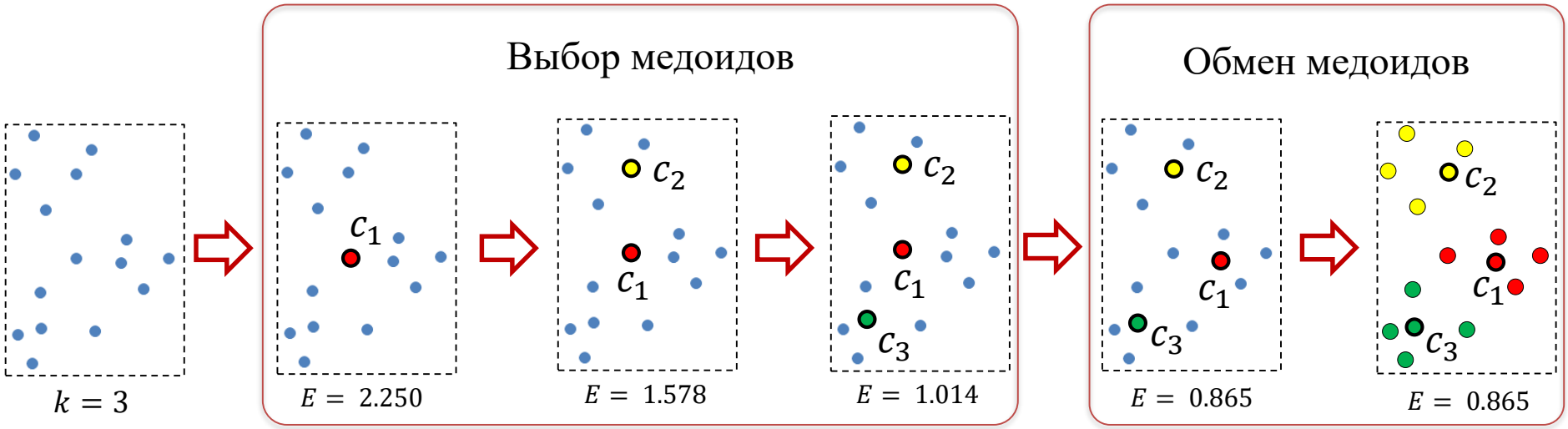
[1)] Hidri M.S., *et al.* Speeding up the large-scale consensus fuzzy clustering for handling Big Data. Fuzzy Sets and Systems. 2018. vol. 348. pp. 50–74.

[2)] Ghadiri N., *et al*. BigFCM: Fast, precise and scalable FCM on Hadoop. Future Generation Comp. Syst. 2017. vol. 77. pp. 29–39.

Параллельные алгоритмы кластеризации, разработанные в диссертации

- Кластеризация графа
- Нечеткая кластеризация данных
- **Кластеризация на основе техники медоидов**

Кластеризация данных на основе техники медоидов



$$c_1 = \arg\min_{1 \le i \le n} \sum_{j=1}^{n} \rho(x_i, x_j)$$

$$c_2 = \arg\min_{1 \le i \le n} \sum_{j=1}^{n} \min\left(\rho(c_1, x_j), \rho(x_i, x_j)\right)$$

$$c_3 = \arg\min_{1 \le i \le n} \sum_{j=1}^{n} \min\left(\min\left(\rho(c_1, x_j), \rho(c_2, x_j)\right), \rho(x_i, x_j)\right)$$

Целевая функция

$E: X \times X \to \mathbb{R}$

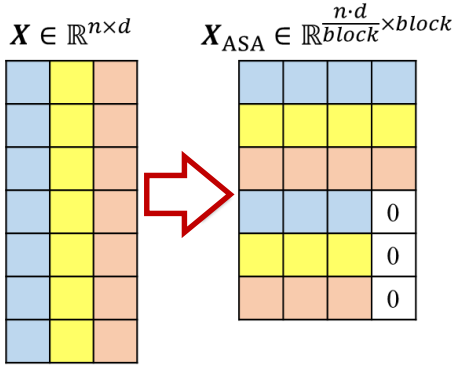
$$E = \sum_{j=1}^{n} \min_{1 \le i \le k} \rho(c_i, x_j)$$

Алгоритм PPAM

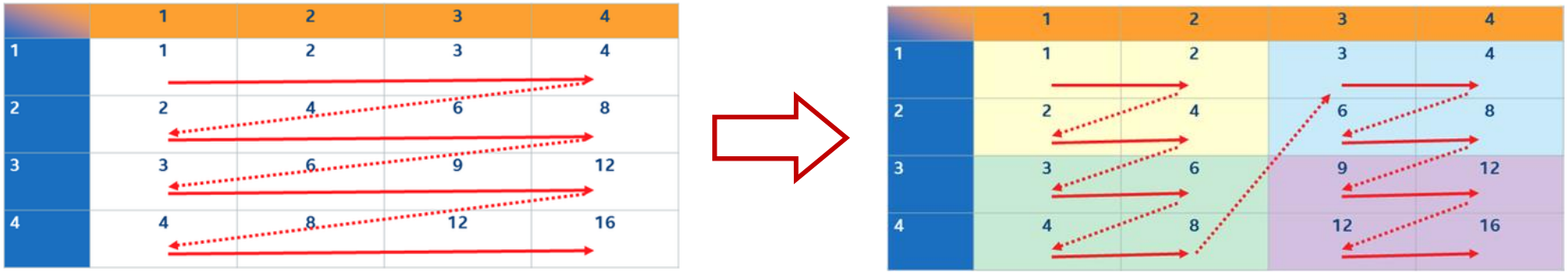
- Область применения
  - Кластеризация данных с выбросами и шумами
- Платформа
  - Многоядерные ускорители
- Отличительные особенности
  - Предвычисление матрицы расстояний
  - Эффективное использование векторизации

Структура алгоритма PPAM

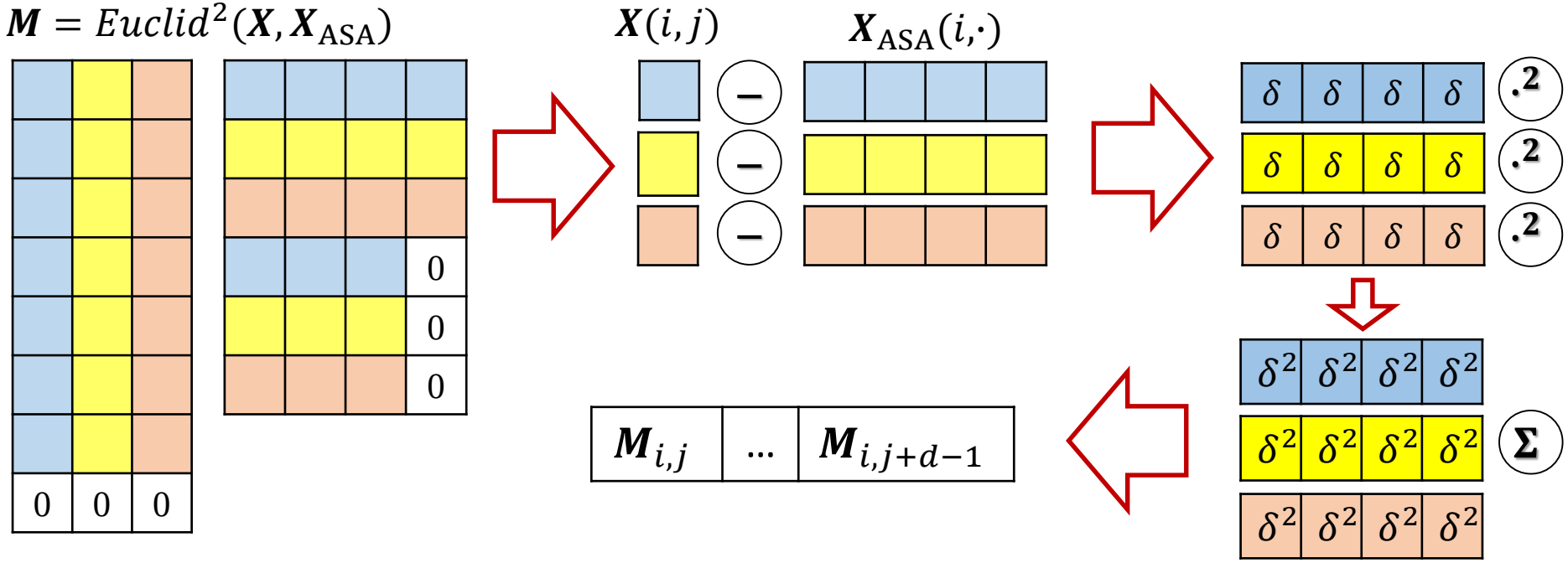
1. Предвычисление матрицы расстояний
   с помощью блочной копии матрицы объектов



2. Выбор и обмен медоидов
   с помощью тайлинга циклов

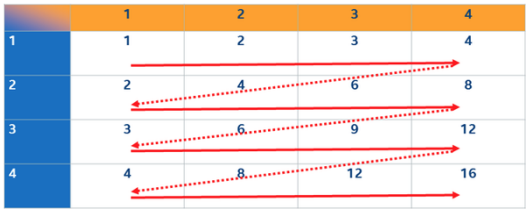


Предвычисление матрицы расстояний. Выбор и обмен медоидов с помощью тайлинга



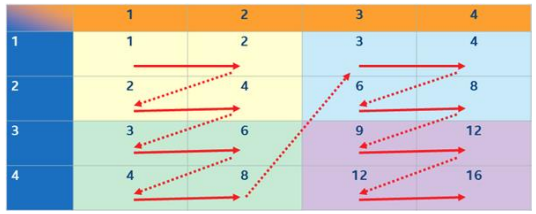
Предвычисление матрицы расстояний. Выбор и обмен медоидов с помощью тайлинга

parallel for $i := 1$ **to** n **do**
$Process(\boldsymbol{M}(i, j))$



Тайлинг

parallel for $i := 1$ **to** n **step** $tile$ **do**
for $i_{tile} := i$ **to** $i + tile$ **do**
$Process(\boldsymbol{M}(i_{tile}, j))$



Сравнение с аналогами

Алгоритм	Время/Качество кластеризации	
	Набор RSSI [1] ($d = 15, n = 6.5 \cdot 10^3, k = 4$)	Набор Avila [2] ($d = 10, n = 2.1 \cdot 10^4, k = 2$)
k-Medians [3]	81.1	163.2
k-Medoids [4]	83.3	52.7
PPAM	**38.3**	**42.2**

[1] Mohammadi M., Al-Fuqaha A.I. Enabling cognitive Smart Cities using Big Data and Machine Learning: approaches and challenges. IEEE Comm. Magazine. 2018. vol. 56, no. 2. pp. 94–101.

[2] De Stefano C., *et al.* Reliable writer identification in medieval manuscripts through page layout features: The "Avila" Bible case. Engineering Applications of Artificial Intelligence. 2018. vol. 72. pp. 99–110.

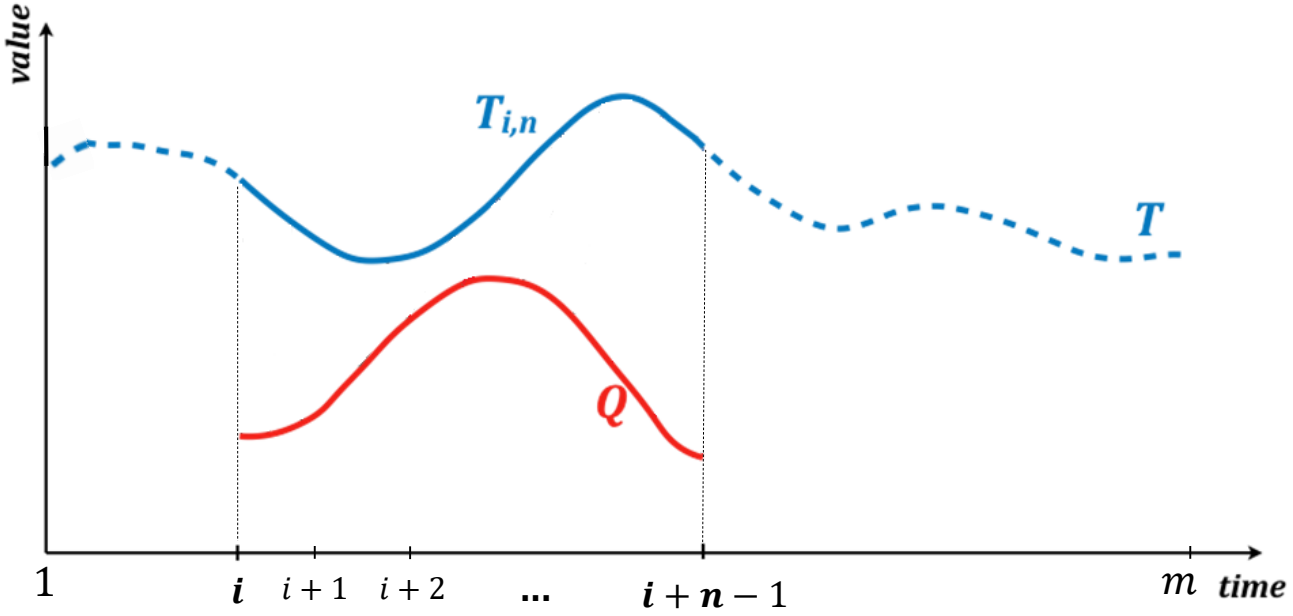
[3] Guha S., *et al.* Clustering data streams. FOCS 2000. pp. 359–366.

[4] Ng R.T., Han J. Efficient and effective clustering methods for spatial Data Mining. VLDB'94. pp. 144–155.

Комплекс алгоритмов интеллектуального анализа данных средствами параллельной СУБД

1. Алгоритмы кластеризации
2. **Алгоритмы анализа временных рядов**
3. Алгоритмы поиска шаблонов

Поиск наиболее похожей подпоследовательности



- Временной ряд: $T = (t_1, \dots, t_m), t_i \in \mathbb{R}$
- Подпоследовательность: $T_{i,n} = (t_i, \dots, t_{i+n-1}), 1 \le i \le m-n+1, n \ll m$
- Образец поиска: $Q = (q_1, \dots, q_n)$
- Мера схожести : DTW
- Подпоследовательность $T_{i,n}$ наиболее похожа на образец поиска Q, если

$$\forall\, T_{j,n} \;\; DTW\left(Q, T_{i,n}\right) \le DTW\left(Q, T_{j,n}\right)$$

Мера схожести
DTW (Dynamic Time Warping)

$$DTW(Q,C) = d(n,n)$$

$$d(i,j) := (q_i - c_j)^2 + \min \begin{cases} d(i-1,j) \\ d(i,j-1) \\ d(i-1,j-1) \end{cases}$$

$$d(0,0) = 0, \qquad d(i,0) = d(0,j) = +\infty$$

$$C = (4,3,3,4,5,4), Q = (1,2,3,3,2,1)$$

$r = 6$

		4	3	3	4	5	4	
1	+∞	28	15	15	20	30	**28**	6
2	+∞	19	11	11	14	20	19	5
3	+∞	15	10	10	11	15	16	4
3	+∞	14	10	10	11	15	16	3
2	+∞	13	10	11	15	24	28	2
1	+∞	9	13	17	26	42	51	1
	0	+∞	+∞	+∞	+∞	+∞	+∞	0
→ *j*	0	1	2	3	4	5	6	↑ *i*

$r = 3$

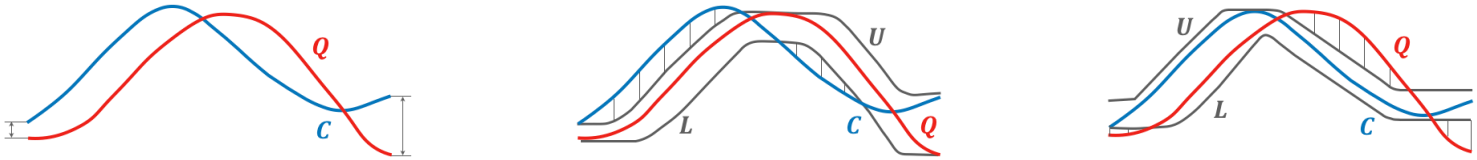
		4	3	3	4	5	4	
1	+∞	0	0	0	9	20	**14**	6
2	+∞	0	0	1	4	10	5	5
3	+∞	0	0	0	1	5	1	4
3	+∞	14	10	10	5	4	0	3
2	+∞	13	10	11	4	0	0	2
1	+∞	9	13	17	0	0	0	1
	0	+∞	+∞	+∞	+∞	+∞	+∞	0
→ *j*	0	1	2	3	4	5	6	↑ *i*

Алгоритм PBM

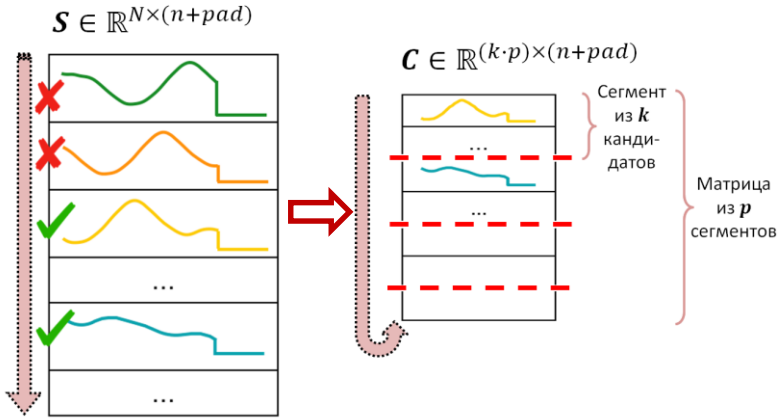
- Область применения
  - Поиск похожих подпоследовательностей в сверхбольших временных рядах
- Платформа
  - Кластерная система с многоядерными ускорителями
- Отличительные особенности
  - Масштабируемость, близкая к линейной
  - Эффективная векторизация вычислений

Структура алгоритма PBM

1. Предвычисление нижних границ схожести

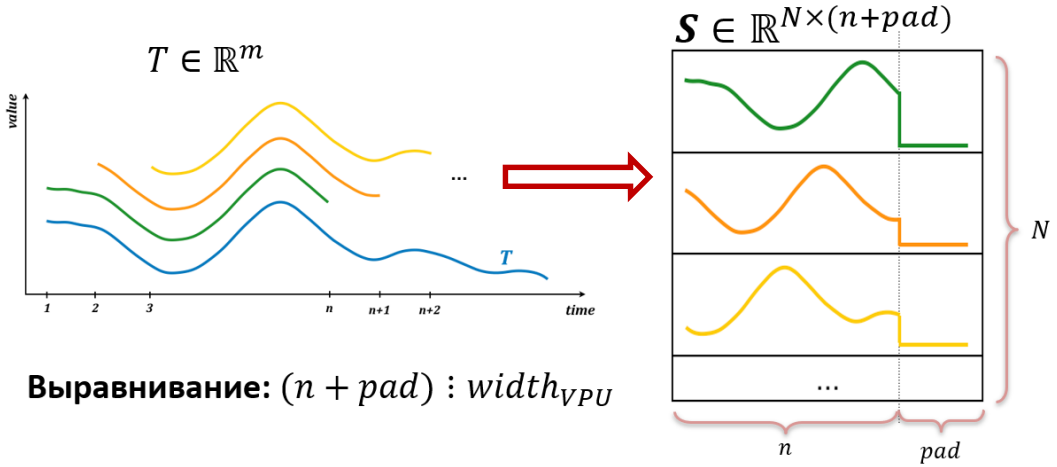


2. Улучшение порога схожести



Предвычисление нижних границ схожести

1. Сформировать S_T^n
2. **pardo** $\forall i \in 1..N$
3. $Normalize(S_T^n(i,\cdot))$
4. **do** $\forall k \in 1..lb_{max}$
5. $L_T^n(i,k) \coloneqq LB_k(S_T^n(i,\cdot))$



Выравнивание: $(n + pad) \vdots width_{VPU}$

Матрица подпоследовательностей

$$S_T^n \in \mathbb{R}^{N\times(n+pad)}$$

$$S_T^n(i,j) = t_{i+j-1}$$

$$N = |T| - n + 1$$

Нормализация

$$\hat{C} = (\hat{c}_1, \dots, \hat{c}_n), \hat{c}_i = \frac{c_i - \mu}{\sigma},$$

$$\mu = \frac{1}{n}\sum_{i=1}^{n} c_i, \sigma^2 = \frac{1}{n}\sum_{i=1}^{n} c^2{}_i - \mu^2$$

Границы схожести

$$LB_1(Q,C) = ED(\hat{q}_1, \hat{c}_1) + ED(\hat{q}_n, \hat{c}_n)$$

$$LB_2(Q,C) = \sum_{i=1}^{n} \begin{cases} (\hat{c}_i - u_i)^2, if\ \hat{c}_i > u_i \\ (\hat{c}_i - \ell_i)^2, if\ \hat{c}_i < \ell_i \\ 0, \text{иначе} \end{cases}$$

$$u_i = \max_{i-r\le k\le i+r} \hat{q}_k, \qquad \ell_i = \min_{i-r\le k\le i+r} \hat{q}_k$$

$$LB_3(Q,C) = LB_2(C,Q)$$

Матрица нижних границ

$$L_T^n \in \mathbb{R}^{N\times lb_{max}}$$

Улучшение порога схожести

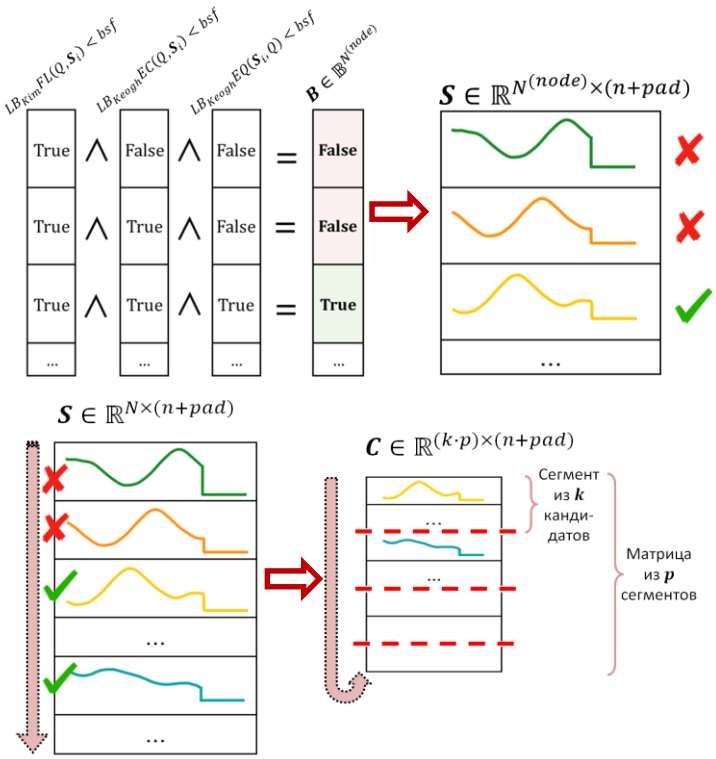
1. **pardo** $\forall i \in 1..N$
2. $B_T^n(i) \coloneqq \bigwedge_{i=0}^{lb_{max}} (L_T^n(i,k) < bsf)$
3. $C_T^n \coloneqq \{S_T^n(i,\cdot) \mid B_T^n(i) = \text{TRUE}\}$
4. **pardo** $\forall c \in C_T^n$
5. $bsf \coloneqq \min_{0 \le i \le thread_{max}} DTW(c,Q)$
6. $position \coloneqq \arg \min_{0 \le i \le thread_{max}} DTW(c,Q)$

Карта схожести

$\boldsymbol{B_T^n} \in \mathbb{B}^N$

Матрица кандидатов

$\boldsymbol{C_T^n} \in \mathbb{R}^{(s \cdot thread_{max}) \times (n+pad)}$

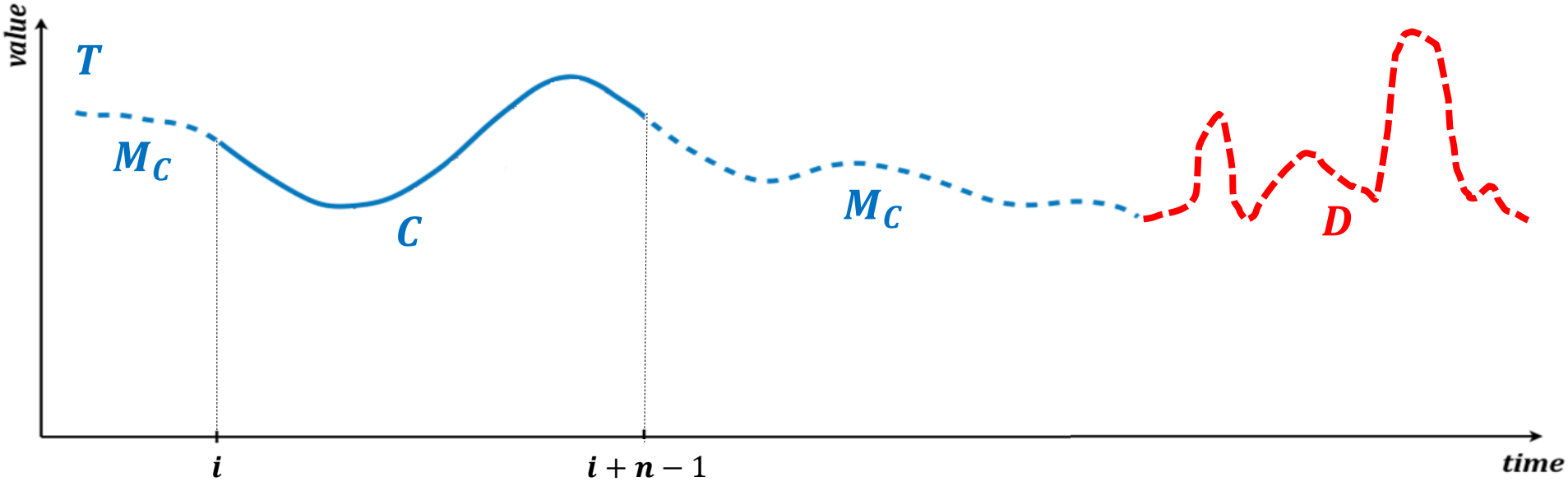


Сравнение с аналогами

Алгоритм	Длина ряда	***PBM*** (ускорение)
Shabib [1)]	$2.2 \cdot 10^8$	**1.3**
Sart [2)]	$1.5 \cdot 10^6$	**2.1**
Huang [3)]	10^6	**2.6**

[1)] Shabib A., *et al.* Parallelization of searching and mining time series data using Dynamic Time Warping. ICACCI 2015. pp. 343–348.
[2)] Sart D., *et al.* Accelerating Dynamic Time Warping subsequence search with GPUs and FPGAs. ICDM 2010. pp. 1001–1006.
[3)] Huang S., *et al.* DTW-based subsequence similarity search on AMD heterogeneous computing platform. HPCC/EUC 2013. pp. 1054–1063.

Поиск аномальной подпоследовательности



- Временной ряд: $T = (t_1, \dots, t_m), t_i \in \mathbb{R}$
- Подпоследовательность: $C = T_{i,n}, n \ll m$
- Непересекающаяся с C подпоследовательность: $M_C = T_{j,n}, |i - j| \geq n$
- Аномальная подпоследовательность D:

$$\forall C, M_C \in T \min(Euclid(D, M_D)) > \min(Euclid(C, M_C))$$

Алгоритм MDD

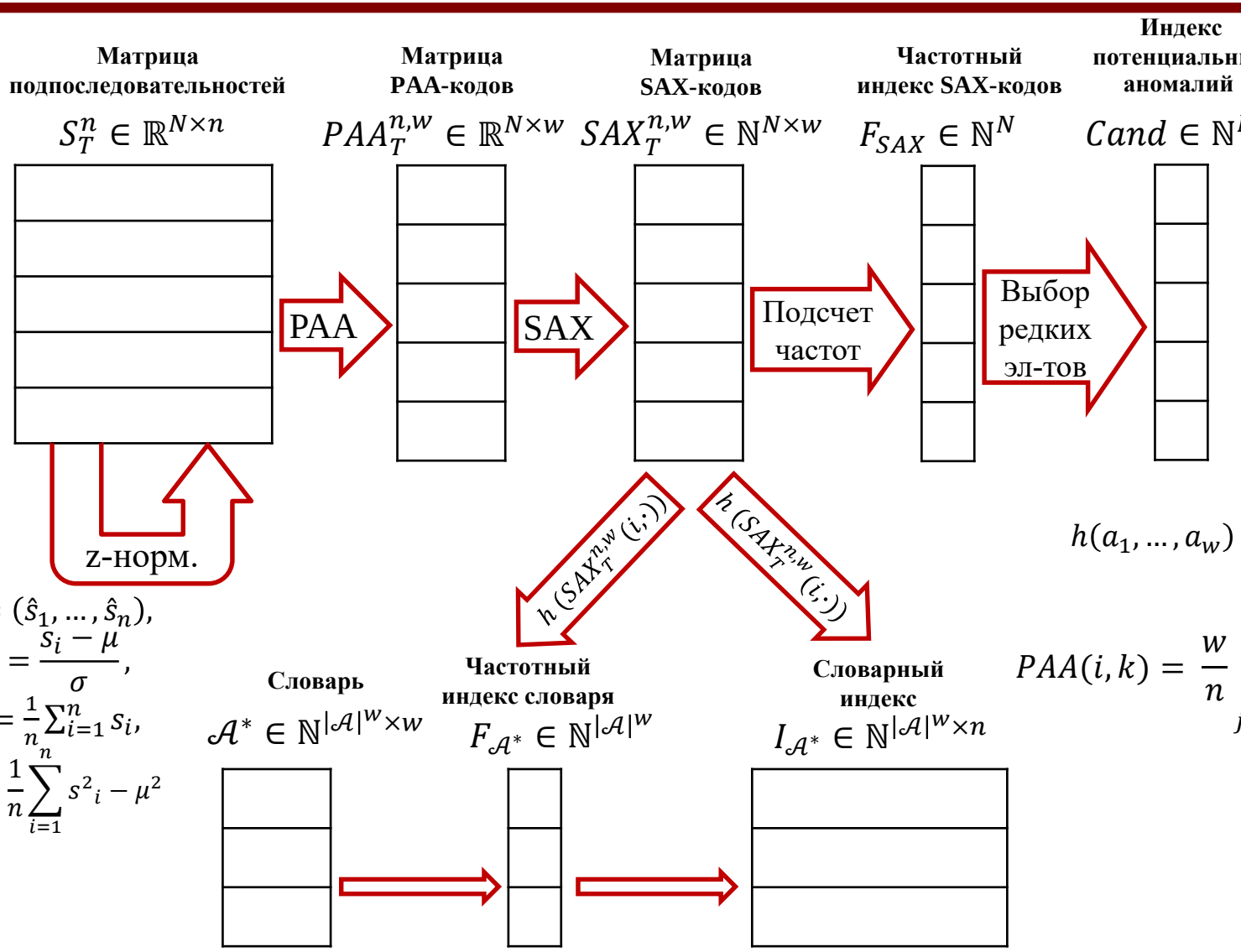
- Область применения
  - Поиск аномальной подпоследовательности временного ряда
- Платформа
  - Многоядерные ускорители
- Отличительные особенности
  - Ускорение, близкое к линейному
  - Эффективная векторизация вычислений

Алгоритм MDD

1. Построение индексных структур

2. Скоринг кандидатов

Построение индексных структур

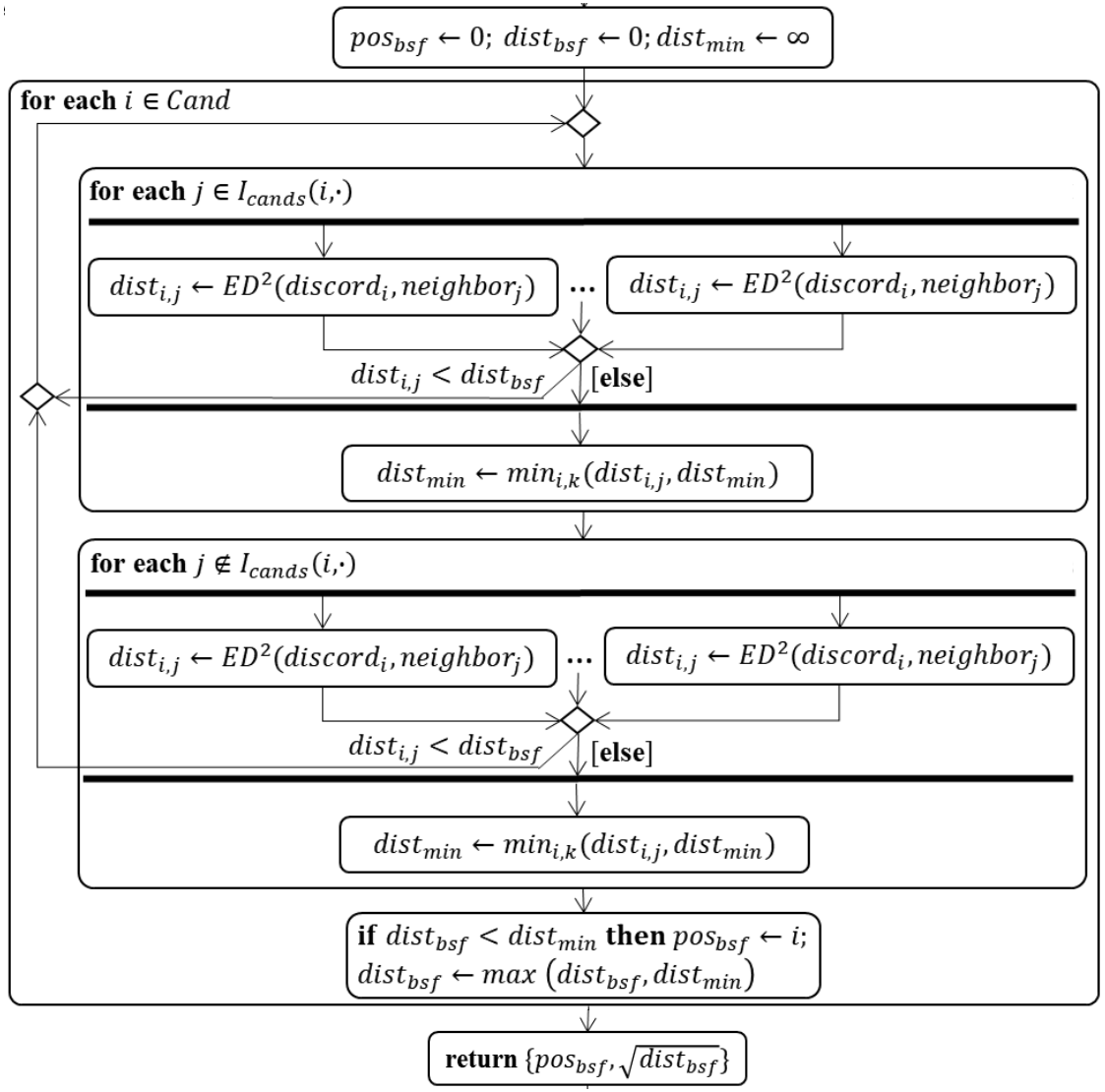


$$\hat{S} = (\hat{s}_1, \dots, \hat{s}_n),$$
$$\hat{s}_i = \frac{s_i - \mu}{\sigma},$$
$$\mu = \frac{1}{n}\sum_{i=1}^{n} s_i,$$
$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} {s^2}_i - \mu^2$$

$$h(a_1, \dots, a_w) = \sum_{j=1}^{w+1} a_j \cdot w^{w-j-1}$$

$$PAA(i,k) = \frac{w}{n} \sum_{j=\left(\frac{n}{w}\right)(i-1)+1}^{\left(\frac{n}{w}\right)i} S(k,j)$$

Скоринг кандидатов



Сравнение с аналогами

Алгоритм	Длина ряда	***MDD*** (ускорение)
DDD [1)]	10^7	**3.1**
MR-DADD [2)]	10^6	**2.4**

[1)] Wu Y., *et al.* Distributed discord discovery: Spark based anomaly detection in time series. HPCC 2015. pp. 154–159.

[2)] Yankov D., *et al.* Disk aware discord discovery: Finding unusual time series in terabyte sized datasets. Knowl. Inf. Syst. 2008. vol. 17, no. 2. pp. 241–262.

Комплекс алгоритмов интеллектуального анализа данных средствами параллельной СУБД

1. Алгоритмы кластеризации
2. Алгоритмы анализа временных рядов
3. **Алгоритмы поиска шаблонов**

Задача поиска шаблонов

T_1: {сыр, хлеб} T_2: {сыр, хлеб, чай} T_3: {кофе, сыр, чай} T_4: {кофе, чай} T_n: {кофе, сахар, сыр, чай}

В супермаркете *найти* **все наборы товаров**, которые **часто** покупают **совместно**

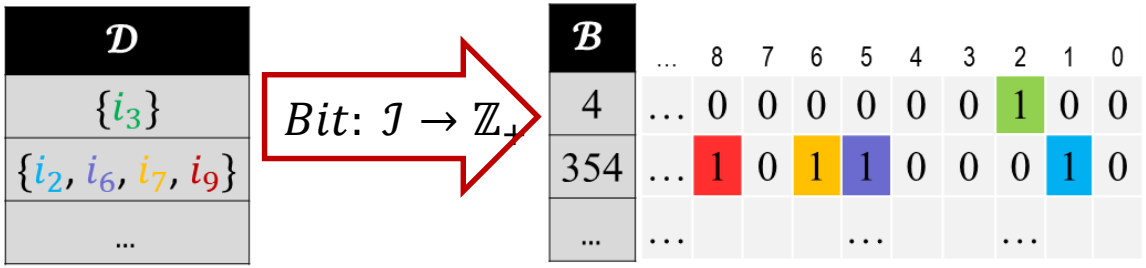
Множество объектов	$\mathcal{I} = \{i_1, i_2, \dots, i_m\}$
k-набор объектов	$I \subseteq \mathcal{I}, \lvert I \rvert = k$
Множество транзакций (корзин)	$\mathcal{D} = \{T_1, \dots, T_n\}, \forall i\ T_i = I, I \subseteq \mathcal{I}$
Поддержка набора	$supp(I) = \frac{\lvert\{T_i \in \mathcal{D} \mid I \subseteq T_i\}\rvert}{n}$
Порог поддержки	$minsup\ (0 < minsup \leq 1)$
Множество частых k-наборов	$\mathcal{L}_k = \{I \mid supp(I) \geq minsup\ \wedge \lvert I \rvert = k\}$
Множество всех частых наборов	$\mathcal{L} = \cup_{k=1}^{k_{max}} \mathcal{L}_k$

Алгоритм PDIC

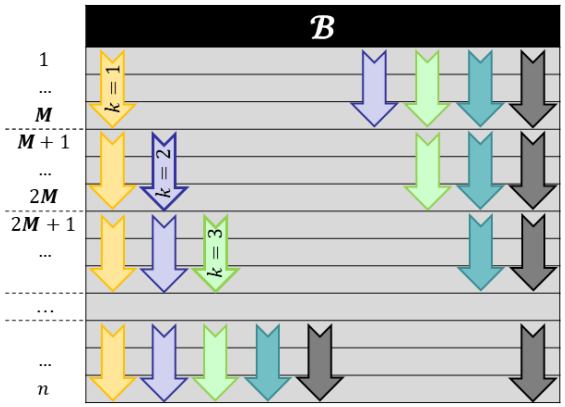
- Область применения
  - Поиск частых наборов в сверхбольших данных
- Платформа
  - Многоядерные ускорители
- Отличительные особенности
  - Использование битовых масок
  - Эффективная векторизация
  - Линейное ускорение

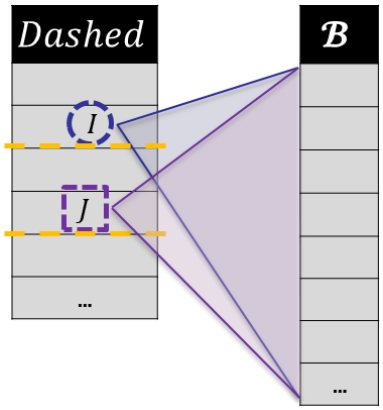
Структура алгоритма PDIC

1. Битовое представление множества корзин



2. Подсчет поддержки наборов



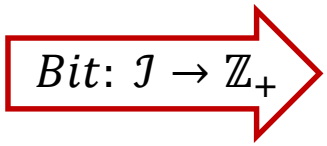


Битовое представление множества корзин

Множество корзин

$\mathcal{D}$
$\{i_3\}$
$\{i_2, i_6, i_7, i_9\}$
…

Функция битовой маски



Битовая карта множества корзин

$\mathcal{B}$	…	8	7	6	5	4	3	2	1	0
4	…	0	0	0	0	0	0	**1**	0	0
354	…	**1**	0	**1**	**1**	0	0	0	**1**	0
…	…				…			…		

Подсчет поддержки набора

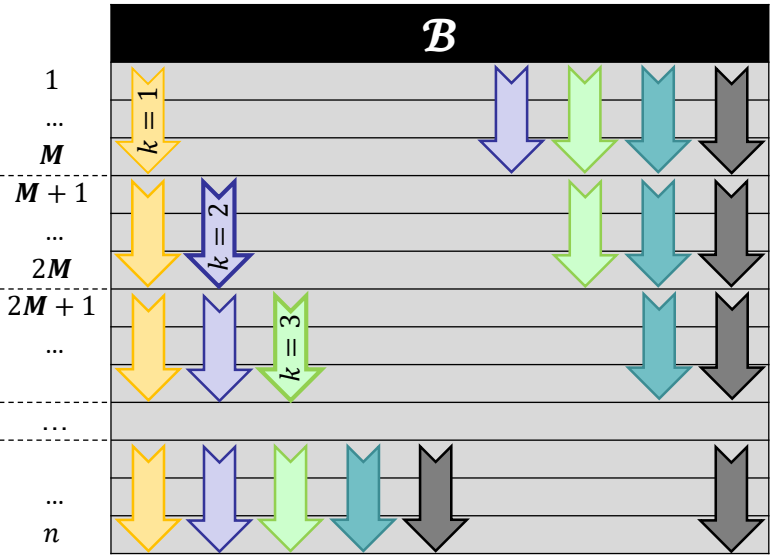
$$I \subseteq T \Leftrightarrow Bit(I) \equiv Bit(I) \wedge Bit(T)$$

Генерация нового k-набора для подсчета поддержки

$$C_k := Bit(I) \vee Bit(J)$$

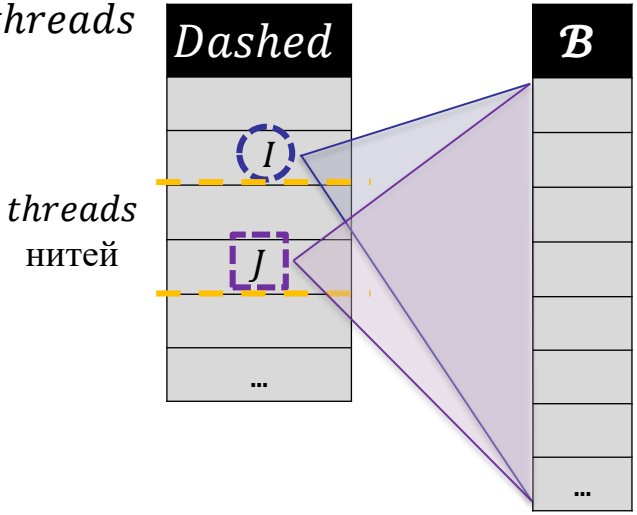
$$\forall\, I, J \in \{I, J \subseteq \mathcal{I} \mid I \neq J, |I| = |J| = k, |I \cap J| = k - 1\}$$

Подсчет поддержки

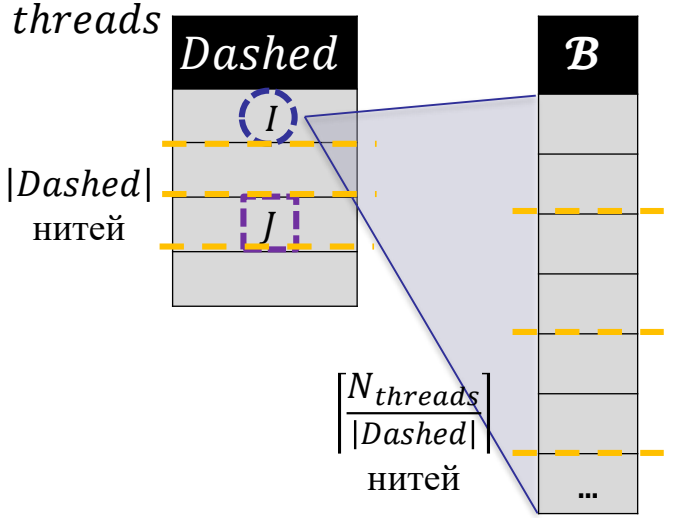


- Функция-счетчик корзин: $Count: \mathcal{I} \times \mathcal{B} \to \mathbb{N} \cup \{0\}$
- Множества наборов
  - I ***Неподтвержденные редкие***
    $DashedCircle = \{I \mid Count(I) < n \ \wedge supp(I) < minsup\}$
  - I ***Неподтвержденные частые***
    $DashedBox = \{I \mid Count(I) < n \ \wedge supp(I) \geq minsup\}$
  - I ***Подтвержденные частые***
    $SolidBox = \{I \mid Count(I) = n \ \wedge supp(I) \geq minsup\}$
  - I ***Подтвержденные редкие***
    $SolidCircle = \{I \mid Count(I) = n \wedge supp(I) < minsup\}$
    Неподтвержденные: $Dashed = DashedCircle \cup DashedBox$
    Подтвержденные: $Solid = SolidCircle \cup SolidBox$

А. $|Dashed| \geq threads$



Б. $|Dashed| < threads$

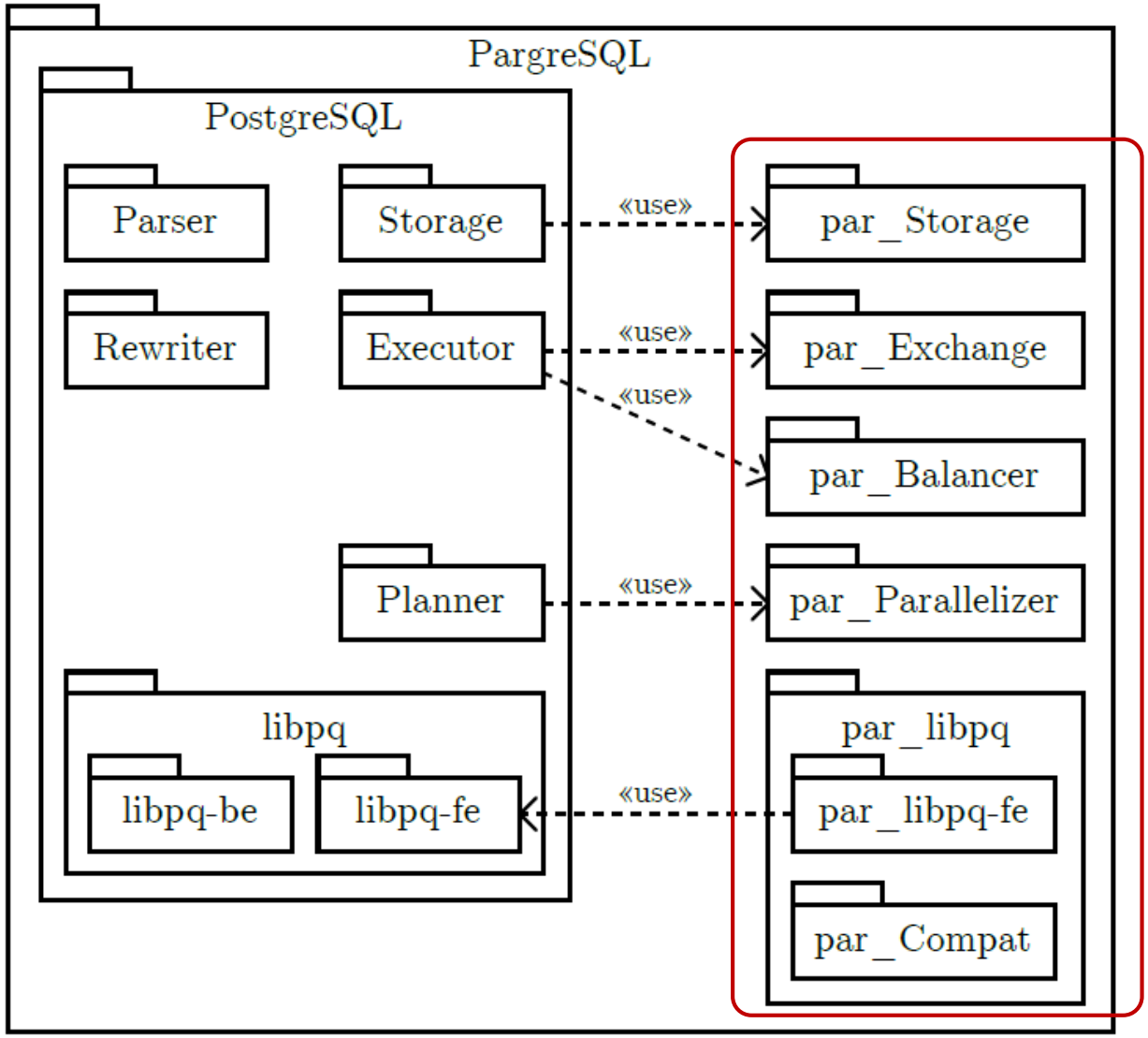


Сравнение с аналогами [1)]

Алгоритм	Время (сек.)	
	Набор RANDOM [2)] $\lvert\mathcal{D}\rvert = 2 \cdot 10^7$	Набор TORNADO [3)] $\lvert\mathcal{D}\rvert = 2 \cdot 10^7$
Apriori	166.6	7.6
FP-Growth	66.7	28.6
Eclat	73.7	31.6
PDIC	**44.1**	**4.0**

[1)] Borgelt C. Frequent Item Set Mining. WIREs: Data Mining and Knowledge Discovery. 2012. vol. 2, no. 6, pp. 437–456.
[2)] IBM Quest Synthetic Data Generator. URL: https://ibmquestdatagen.sourceforge.io/.
[3)] Zymbler M. Parallel algorithm for frequent itemset mining on Intel manycore systems. CIT. 2019. vol. 26, no. 4. pp. 209–221.

Параллельная СУБД PargreSQL



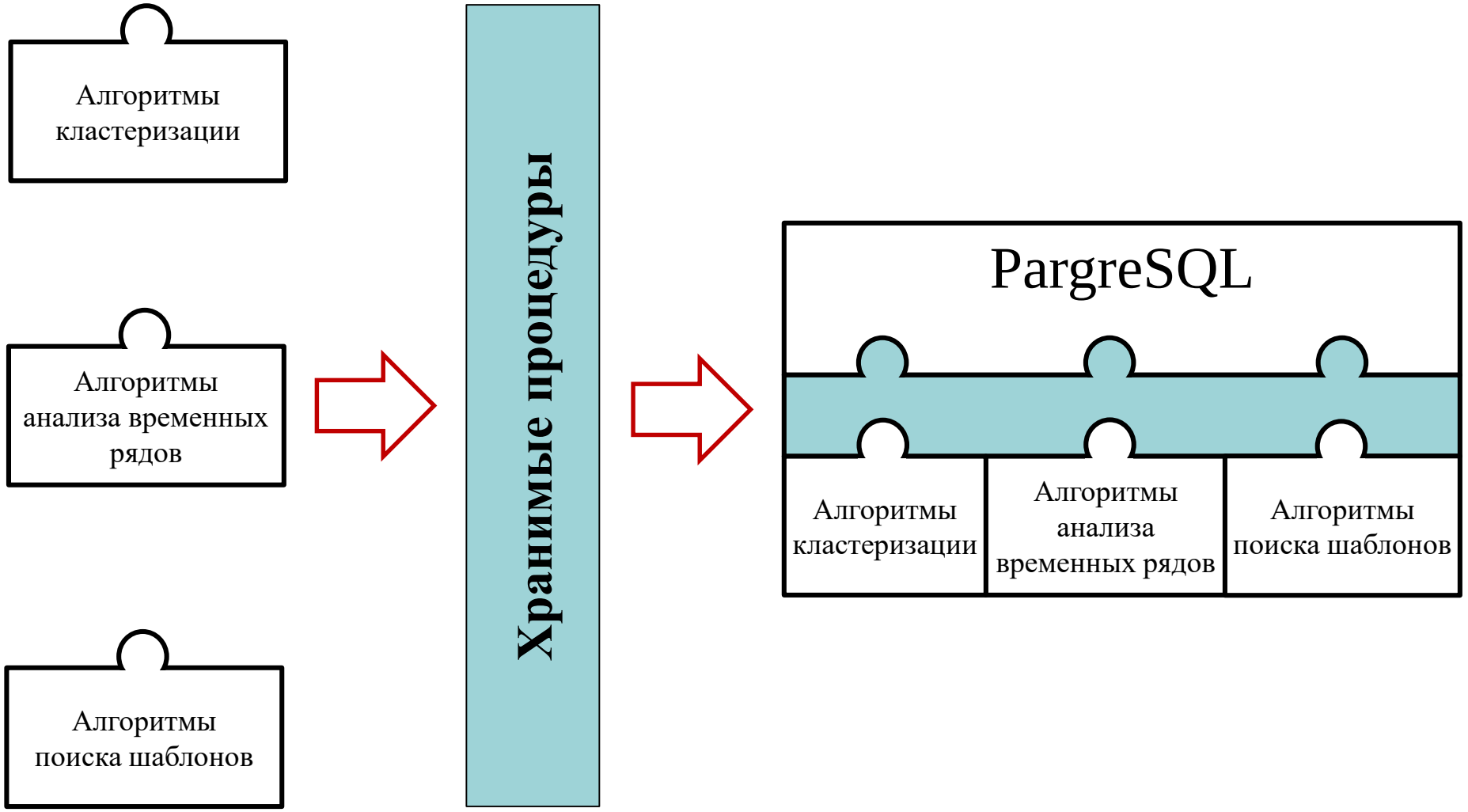
0.5%
исходного кода PostgreSQL

Сравнение с аналогами [1)]

Место	СУБД (платформа)	Транзакций в мин.
1	Oracle Database 11g R2 ER (SPARC SuperCluster with T3-4 Servers)	30 249 688
2	IBM DB2 9.7 (IBM Power 780 Server Model 9179-MHB)	10 366 254
3	Oracle Database 11g ER (SPARC Enterprise T5440 Server Cluster)	7 646 486
4	**PargreSQL (Торнадо ЮУрГУ)**	**2 202 531**
5	Oracle Database 10g ER (HP Integrity rx5670 Cluster)	1 184 893

[1)] Консорциум TPC. Стандартный тест TPC-C. http://www.tpc.org/tpcc/

Внедрение алгоритмов анализа данных в PargreSQL



Основные результаты, выносимые на защиту

1. Разработан и исследован комплекс параллельных алгоритмов интеллектуального анализа данных средствами СУБД на кластерных вычислительных системах с многоядерными ускорителями:
   1) параллельный алгоритм поиска похожих подпоследовательностей временного ряда для кластеров с многоядерными ускорителями;
   2) параллельный алгоритм поиска аномальных подпоследовательностей временного ряда для многоядерных ускорителей;
   3) алгоритм кластеризации графа для параллельной СУБД на основе фрагментного параллелизма;
   4) алгоритм нечеткой кластеризации данных для параллельной СУБД на основе фрагментного параллелизма;
   5) параллельный алгоритм кластеризации данных для многоядерных ускорителей;
   6) параллельный алгоритм поиска частых наборов для многоядерных ускорителей
2. Предложен метод внедрения параллелизма в последовательные СУБД с открытым исходным кодом
3. Предложен подход к внедрению параллельных алгоритмов интеллектуального анализа данных в реляционные СУБД