УДК 004.93'14

Р. А. Мельник, Р. Б. Тушницький

ДЕКОМПОЗИЦІЯ ПРОСТОРУ ПІД ЧАС КЛАСТЕРИЗАЦІЇ ДАНИХ ВЕЛИКОЇ РОЗМІРНОСТІ

An approach to reduce algorithmic complexity for clustering of large-scale dataset is considered. The main idea is decomposition of item dataset and space by hypercube coordinates. To join clusters from subsets into the result clusters and to minimize the accuracy losses are the main tasks of the algorithm. Some visual patterns with large pixel numbers as test examples were investigated.

Для зменшення часових затрат під час кластеризації даних великих розмірів запропоновано декомпозиційний підхід, що базується на розбитті простору згідно з координатними вісями гіперкубів. Відповідне керування алгоритмом дає змогу об'єднувати кластери – результати з підмножин – у кінцеві при незначних втратах точності. Як приклади практичних даних використані зображення із значними кількостями пікселів.

Методи кластерного аналізу широко використовують для декомпозиції, дослідження, індексування та пошуку, розпізнавання зображень [1–4]. Зокрема, робота [1] містить класифікацію методів кластеризації та спосіб формування контурів виділених кластерів. Роботи [2–3] присвячені кластеризації графових моделей, якими відображають частини зображень. Ієрархічний алгоритм кластеризації в [4] має один етап процедури згортання і пропонує новий критерій об'єднання для зменшення обчислювальних затрат із складності $O(N^3)$ до $O(N^2)$. Для великих обсягів даних з багатьма характеристиками ця складність виявляється також перешкодою для групування. Тому в роботі запропоновано декілька підходів для зменшення складності на основі декомпозиції простору пошуку кластерів. Перші спроби продемонстровано в роботах [5–7].

Декомпозиція даних та каскадне згортання. Практичні класи задач кластеризації даних є часоплинними завдяки об'ємам даних і складності алгоритму. Обчислювальна складність класичного ієрархічного алгоритму сягає значення $O(N^3)$, а певними кроками обмеження перебору можна досягнути величину $O(N^2)$, що підтвердили експерименти роботи [4] з дослідження залежності часу побудови дерева згортання від величини початкової вибірки. В роботі [6] реалізовано наближений підхід кластеризації ключів з розбиттям множини базових ключів. Для подальшого зменшення складності алгоритму ієрархічної кластеризації пропонуємо декомпозиційний підхід, базований на розбитті початкової вибірки даних великої розмірності на ряд підмножин. Розглядаємо дві можливості: розбиття вибірки на частини з виконанням обмежень на кількість значень у підмножині та розбиття простору на частини без обмежень на потужність підмножин значень даних.

За базовий приймаємо класичний ієрархічний алгоритм згортання – побудови дерева кластерів різних рівнів. У цій роботі запропоновано багаторівневу деомпозицію множин ключів і кластерів, яку назвемо *каскадною кластеризацією*.

Розбиваємо вхідну множину ключів $Q(Q_1, Q_2, Q_3, \dots Q_N)$ на p підмножин $O_1(Q_1, Q_2, Q_3, \dots Q_z)$, $O_2(Q_{z+1}, Q_{z+2}, Q_{z+3}, \dots Q_t)$, … , $O_p(Q_{t+1}, Q_{t+2}, Q_{t+3}, \dots Q_N)$. До кожної з підмножин (назвемо їх множинами нульового каскаду, застосуємо алгоритм кластеризації, утворивши множини відповідних кластерів $K_1(k_1, k_2, k_3, \dots)$, $K_2(k_s, k_{s+1}, k_{s+2}, \dots)$, … , $K_p(k_r, k_{r+1}, k_{r+2}, \dots)$, де $k_1, k_2, \dots, k_i, \dots$ – кластери, ключі в яких відносяться до відповідних підмножин $O_1, O_2, \dots, O_p$. Утворимо множину кластерів 1-го каскаду кластеризації K об'єднанням

$$K = K_1(k_1, k_2, k_3,\ldots) \cup K_2(k_s, k_{s+1}, k_{s+2},\ldots) \cup \ldots \cup K_p(k_r, k_{r+1}, k_{r+2},\ldots). \quad (1)$$

Застосуємо до цієї множини алгоритм кластеризації, розглядаючи кожен з кластерів k_1, k_2,…, k_i, ... як базовий, тобто листок дерева згортання. При цьому вони (кластери 1-го каскаду) перемішуються, тобто виступають як незалежні. Утворену множину кластерів 1-го каскаду поділимо на підмножини O_1, O_2, …, O_p, далі здійснюється кластеризація в межах підмножин до отримання множини кластерів 2-го каскаду. Під час отримання кореня дерева згортання на 2-му каскаді матимемо двокаскадну декомпозицію, на третьому каскаді – трикаскадну і т.д. Кількість каскадів визначається кількістю рівнів ієрархічного дерева згортання, на яких здійснюється поділ кластерів на множини (без рівня ключів). Схема поділу та згортання є рекурсивною (рис. 1).

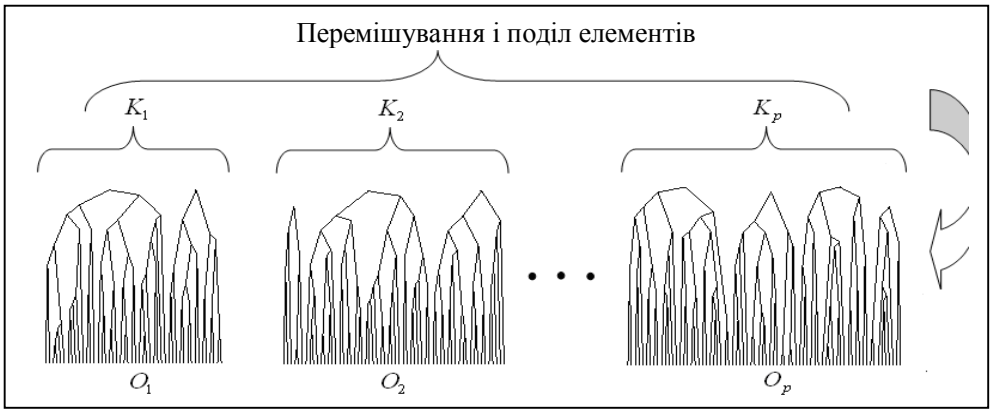


Рис. 1. Каскадне дерево формування кластерів.

На кількість каскадів вказують коефіцієнти поділу: наприклад, 20000 ключів розбивається на 100 груп по 200 ключів з виходом на 2-й каскад по 20 кластерів з групи. Тоді на 2-му каскаді є 100·20=2000 кластерів, які ділимо ще на 10 груп по 200 елементів з виходом на вищий каскад 20 кластерів з групи. Кількість каскадів визначається автоматично залежно від кількості кластерів, до яких застосовується повний перебір. У розробленому пакеті це число вноситься користувачем.

У процесі дослідження алгоритму 10000 ключів генерувались за рівномірним та нормальним законами розподілу з різними центрами координат та однаковими дисперсіями. Тестування проведені для різної кількості множин (відповідно їх потужностей) для 0-го каскаду кластеризації. Як і очікувалось, результати, отримані на кінцевому каскаді, можна охарактеризувати так:

1) якісні та кількісні показники кластерів є тим кращими, чим більша кількість кінцевих кластерів виходить з групи. Менша кількість груп дає кращі результати;

2) часові показники програми зворотні, тобто вони менші для меншої кількості елементів в кожній групі, а кількість груп є максимально можлива.

При каскадній декомпозиції важливим питанням є вибір потужностей множин O_1, O_2, …, O_p в межах каскадів і, відповідно, їх кількості. Вагомим також є попередня обробка даних у межах множин O_1, O_2, …, O_p, що якісно впливає на результат кластеризації.

Можливі два способи розбиття вхідної множини на підмножини O_1, … , O_p:

1) поділ на підмножини без попередньої обробки, наприклад, розбиття випадковим чином. Через перемішування і об’єднання значень, що реально розта-

шовані на далеких відстанях, цей підхід має тільки теоретичне значення. У ньому результати кластеризації порівняно з іншими правилами розбиття є найгіршими. Складність алгоритму в цьому підході визначається кількістю значень n_i у підмножині O_i та кількістю підмножин розбиття, а саме: $O(p \cdot n_i^3)$;

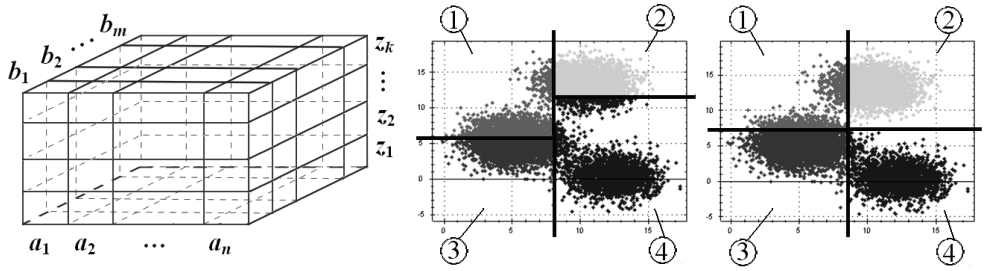
2) поділ на підмножини згідно з кількістю та розташуванням. Розташування контролюється сортуванням вхідної вибірки за однією пріоритетною координатою (ознакою). Далі за цією ж координатою здійснюється поділ на підмножини за значенням кожної з потужностей. До підмножин застосовується алгоритм каскадної кластеризації.

Вибір пріоритетної координати покладається на користувача програми. Складність алгоритму збільшується на одне сортування повної вибірки, тобто дорівнює: $O(p \cdot n_i^3) + O(N^2)$.

Алгоритм розроблено для ілюстрації впливу цілеспрямованого поділу вхідної вибірки даних, яка не може бути кластеризована класичним алгоритмом через великий об'єм даних.

Декомпозиція простору. Підхід з декомпозицією всього простору n-вимірних даних полягає в поділі всіх координат на частини і побудови гіперкубів. Фактично рознесення значень точок n-вимірного простору у відповідний гіперкуб вимагає попереднього опрацювання даних, а саме сортування значень вибірки за всіма координатами.

Нехай простір складається із s n-вимірних даних: $C = C(a, b, \ldots, z)$. Вимірність a поділимо на n частин, b – на m частин, ..., вимірність z – на k частин. Поділ простору за вимірностями на частини за параметрами $n, m, \ldots, k$ позначимо вектором розбиття $l = (n, m, \ldots, k)$. Схематично поділ тривимірного простору зображено на рис. 2*а*.



а *б* *в*

Рис. 2. Схема поділу тривимірного простору на куби (*а*) та поділ двовимірного простору: *б* – гіперкуби із однаковою кількістю точок; *в* – за значеннями по кожній координаті.

Використаємо два способи поділу простору: гіперкуби різних розмірів, але з однаковою кількістю значень даних; другий – на гіперкуби з неконтрольованими кількостями значень даних, але контрольованими розмірами сторін, що задаються користувачем. На рис. 2*б–в* зображено простір що складається із 20000 2-вимірних точок, генерованих за нормальним законом розподілу. На рис. 2*б* вектор розбиття $l = (2, 2)$ дає гіперкуби (прямокутники у 2-вимірному просторі) із однаковою кількістю точок, але різних за розмірами. На рис. 2*в* цей же вектор розбиття та поділ координатних проміжків на рівні частини дає однакові гіперкуби. На рис. 2*б* – кожна підмножина (кожен квадрант) містить по 5000 точок. На рис. 2*в* зображено: в 1-му квадранті 2030 точок, в 2-му – 11350, в 3-му – 4540, в 4-му – 2080 точок.

Складність алгоритму збільшується на сортування вибірки за всіма координатами, тобто дорівнює $O(p \cdot n_i^3) + p \cdot O(N^2)$.

Експериментальні дослідження. Для проведення експериментів використано три множини значень, генерованих за нормальним законом розподілу із математичним очікуванням 0,75 та середнім квадратичним відхиленням 0,95. Повна вибірка – це набір із 10000 5-вимірних значень. Вибірка містить три природних кластери з підмножинами: 5000, 2000, 3000.

Каскадна декомпозиція без попередньої обробки даних (1-й алгоритм) та із сортуванням за пріоритетною координатою (2-й алгоритм). Проведемо три експерименти для поділу на 4 групи по 2500 точок, 8 груп по 1250 точок, та 16 груп по 625 точок. Для обох алгоритмів встановимо кількість результуючих кластерів, які виходять з групи 10% від всієї кількості значень у групі.

Декомпозиція простору із поділом на гіперкуби із однаковою кількістю точок (3-й алгоритм) та за поділом кожної координати (4-й алгоритм). Розбиваємо простір за векторами розбиття $l_1 = (2, 2, 1, 1, 1)$ – 4 групи, $l_2 = (2, 2, 2, 1, 1)$ – 8 груп та $l_3 = (2, 2, 2, 2, 1)$ – 16 груп. Для першого алгоритму гіперкуби матимуть однакову кількість точок – 2500, 1250 та 625 відповідно. Для другого гіперкуби міститимуть різну кількість значень. Для обох алгоритмів встановимо кількість результуючих кластерів з групи 10% від всієї кількості значень у групі.

Для оцінки результатів кластеризації використовуємо функції якості: питомі функції дисперсії, густини, об'єму та відстані (подібності), запропоновані в роботі [5]. На рис. 3 зображено результати роботи чотирьох вище зазначених алгоритмів. На кожному з окремих рисунків три графіка відповідають трьом способам поділу даних чи простору:

– в першому ряду – результати каскадної кластеризації для випадку поділу даних випадковим чином, тобто без попередньої обробки даних;

– в другому ряду – результати для випадку поділу на підмножини із використанням алгоритму сортування *QuickSort* за першою координатою;

– третій ряд ілюструє результати для випадку розбиття простору на гіперкуби із однаковою кількістю точок;

– четвертий ряд – результати для випадку розбиття простору на гіперкуби за їх значеннями по кожній координаті.

Координатами точок графіків є значення функцій якості та номер рівня дерева згортання. На графіках функції сусідства скупчення точок виникають внаслідок стиснення осі абсцис. Масштаби графіків вибрані таким чином, щоб зафіксувати та порівняти хід зміни функції.

Для порівняння алгоритмів у графіках на рис. 4 представлено зведені результати кластеризації для випадку розбиття даних та простору на 8 груп. Всі числові результати експериментів з чотирма алгоритмами зведено у табл. 1. Вони підтверджують ефективність підходу поділу даних та простору на частини.

Проілюструємо результати роботи алгоритмів на зображеннях розмірами 154×103 пікселів (рис. 5*а* – *в*). Вхідна вибірка – це координати *x*, *y* та *color* – середнє арифметичне кольорів пікселів. Всього дані містять 15862 3-вимірних точок. В експерименті з підгрупи (гіперкуба) прийнято вихід кластерів, що дорівнює 10% від їх кількості у цій підгрупі.

На рис. 5 зображено графічні результати кластеризації зображень, а числові характеристики зведено у табл. 2. Представлено кінцеві кластери (позначені одним кольором) на певних рівнях ієрархічного дерева згортання при випадковому перемішуванні значень пікселів та каскадного згортання (*б*), при вказуванні координати *x* як пріоритетної та каскадного згортання (*в*) та відповідного ділення даних та простору (*г*, *д*). З рис. 5 видно, що точність перших двох алгоритмів не висока, а точність третього залежить від вектора розбиття. Зауважимо, що декомпозиційні алгоритми в представленому вигляді не плануються використовувати для

кластеризації зображень, а для опрацювання даних іншої фізичної природи: гени, тексти, хімічні елементи тощо.

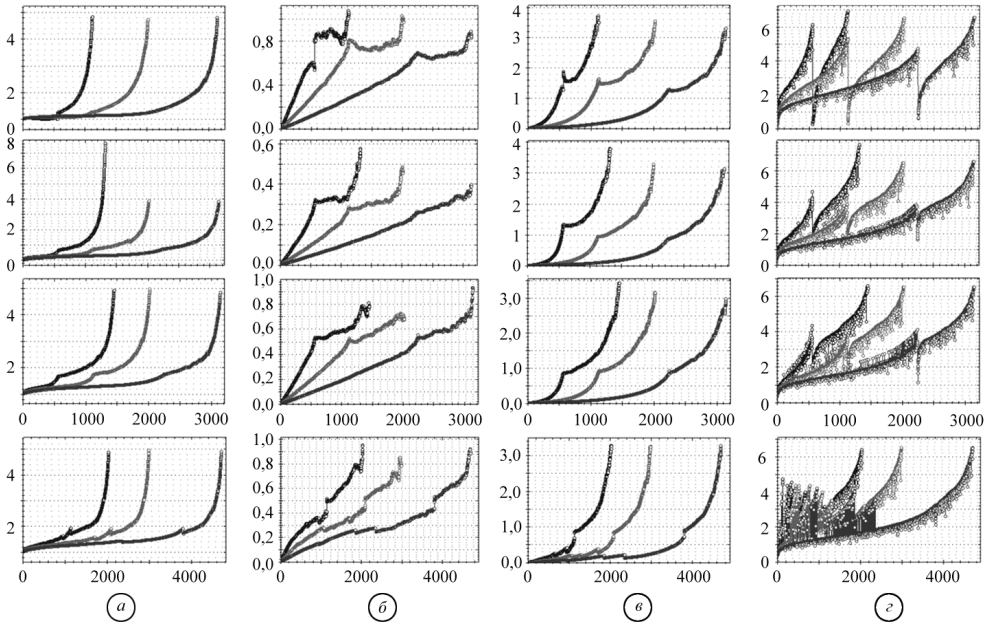


Рис. 3. Залежності функцій якості (вісь ординат) від рівня дерева згортання (вісь абсцис) під час поділу даних та простору на 4, 8 та 16 частин:
а – функція густини; *б* – функція об'єму; *в* – функція дисперсії; *г* – функція сусідства.

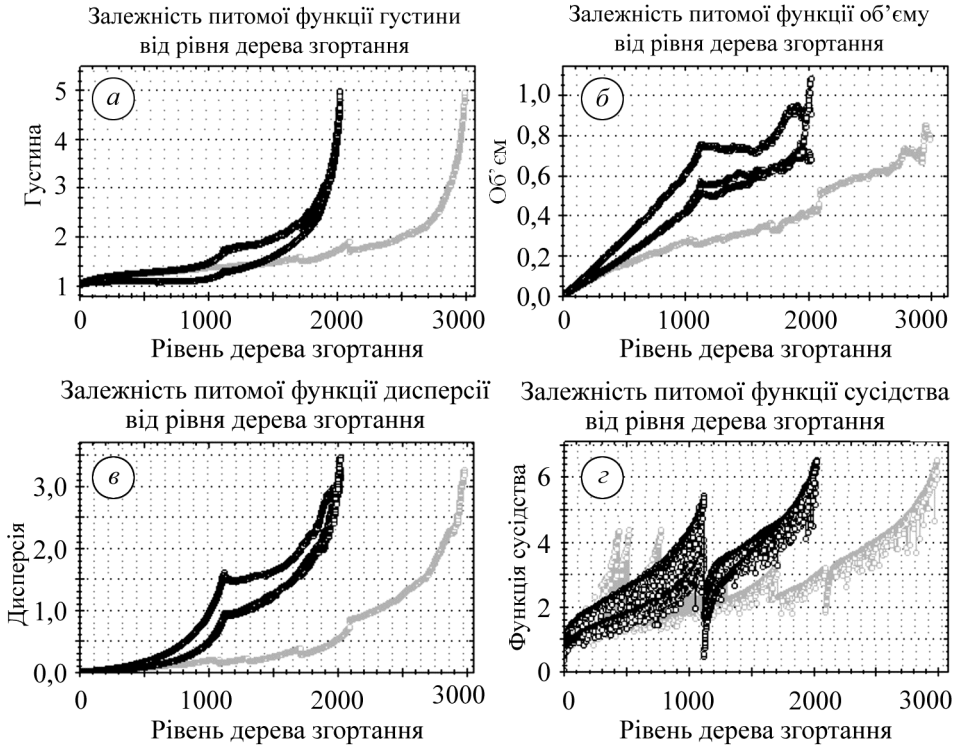


Рис. 4. Залежності функції якості від рівня дерева згортання під час поділу даних та простору на 8 частин:
а – функція густини; *б* – функція об'єму; *в* – функція дисперсії; *г* – функція сусідства.

Таблиця 1. Характеристики кластеризації експериментальних даних

Алгоритм декомпозиції	Попередня обробка	Груп	К-сть в групі	Час, хв	Каскад, разів	Питома дисперсія	Питома густина	Питомий об'єм
Каскадна декомпозиція	Без попередньої обробки	4	2500	23:30	1	1,2576	3,1140	0,7991
		8	1250	04:53	1	1,5791	2,9706	0,9107
		16	625	01:09	2	1,8985	2,7125	1,0281
	Сортування за пріоритетною ознакою	4	2500	17:53	1	0,8571	3,5619	0,6274
		8	1250	04:31	1	0,9039	3,5553	0,6449
		16	625	01:12	2	1,0857	3,6323	0,6817
Декомпозиція гіперкубів	Розбиття на куби із однаковою кількістю точок	4	2500	20:49	1	0,8649	3,6123	0,6279
		8	1250	05:00	1	0,8768	3,5616	0,6352
		16	625	01:30	1	0,8961	3,5023	0,6450
	Розбиття на куби із поділом кожної координати	4	–	34:39	1	0,8529	3,5435	0,6275
		8	–	08:12	1	0,8724	3,5402	0,6349
		16	–	02:21	1	0,8971	3,5534	0,6433

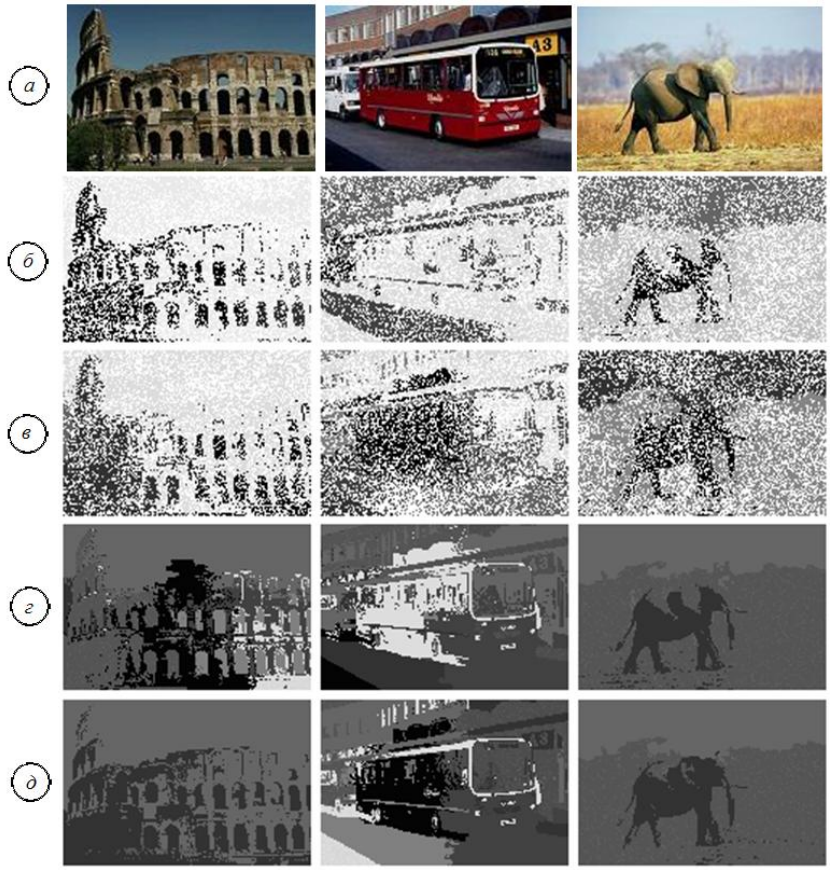


Рис. 5. Кластеризація експериментальних даних отриманих із зображень (*а*): *б* – без попередньої обробки каскадним згортанням; *в* – сортування за пріоритетною ознакою каскадним згортанням; *г* – розбиття на куби із однаковою кількістю точок за вектором розбиття (4, 4, 1); *д* – розбиття на куби з однаковою кількістю точок за вектором розбиття (2, 2, 1).

Таблиця 2. Характеристики кластеризації даних із зображень

м	Попередня обробка	Зображення	Рівень (всього рівнів)	Кластерів на рівні	Каскад, разів	Питома дисперсія	Питома густина	Питомий об'єм	Час, хв	Розбиття
Каскадна декомпозиція	Без попередньої обробки (*г*)	*а*	1527 (1530)	3	2	44,0811	1,5003	3,6189	1,50	16/992
		б	1557 (1566)	9	2	45,9041	1,4808	3,6117	1,58	
		в	1476 (1480)	4	2	34,7678	1,5503	3,1879	1,34	
	Сортування за пріоритетною ознакою (*д*)	*а*	1541 (1548)	7	2	44,2001	1,5076	3,5855	1,51	16/992
		б	1543 (1553)	10	2	46,1983	1,4723	3,6555	1,57	
		в	1498 (1501)	3	2	34,6141	1,5448	3,1900	1,35	
Декомпозиція гіперкубів	Розбиття на куби із однаковою кількістю точок (*е–є*)	*а*	2033 (2043)	10	1	15,1658	2,0907	1,3590	1.51	(4,4,1)
		б	2041 (2051)	10	1	16,0968	2,0411	1,3784	1,52	
		в	2011 (2014)	3	1	11,7393	2,1788	1,2023	1,36	
		а	2968 (2971)	3	1	14,7260	2,0956	1,3417	10,33	(2,2,1)
		б	3005 (3015)	10	1	15,6494	2,0456	1,3617	11,19	
		в	2945 (2948)	3	1	11,5100	2,1889	1,1857	8,24	

З табл. 2 видно, що за питомими об'ємом, густиною та дисперсією найкращі результати отримано застосуванням декомпозиції гіперкубів, що підтверджують графічні результати з рис. 5.

Отже, для кластеризації даних великої розмірності найкращими алгоритмами та підходами є каскадна декомпозиція із попередньою обробкою (2-й алгоритм) та декомпозиція до гіперкубів (4-й алгоритм).

ВИСНОВКИ

Класичний ієрархічний алгоритм не дає змогу опрацьовувати великі об'єми даних через великі часові затрати. Розроблено алгоритми, що базуються на розбитті простору та даних відповідно до нього на частини. Головну увагу приділено керуванню під час продовження згортання кластерів, що виходять з підмножин. Алгоритми дають змогу опрацювати дані, які точним алгоритмом зробити неможливо. Втрати точності відбуваються на межах гіперкубів. Подальші дослідження необхідні для мінімізації зазначених втрат.

Алгоритми рекомендуються для застосування у автоматизованих системах зберігання та пошуку даних різної фізичної природи.

1. *Yip A. M., Ding C., Chan T. F.* Dynamic Cluster Formation Using Level Set Methods // IEEE Trans. on Pattern Analysis and Machine Intelligence. – 2006. – **28**, № 6. – P. 877–889.
2. *Grady L., Schwartz E. L.* Isoperimetric Graph partitioning for Image segmentation // IEEE Trans. on Pattern Analysis and Machine Intelligence. – 2006. – **28**, № 3. – P. 469–475.
3. *Pavan M., Pelillo M.* Dominant sets and Pairwise Clustering // IEEE Trans. on Pattern Analysis and Machine Intelligence. – 2007. – **29**, № 1. – P. 167–172.
4. *Ding C., He X.* Cluster Aggregate Inequality and Multilevel Hierarchical Clustering // Proc. 9th European Conf. Principles of Data Mining and Knowledge Discovery. – 2005. – P. 71–83.
5. *Melnyk R., Tushnytskyy R.* Algorithm Accuracy and Complexity Optimization by Inequality Merging for Data Clustering // Proc. of the Xth Intern. Conf. The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM). – 2009. – P. 453–455.
6. *Мельник Р. А., Алексєєв О. А.* Кластеризація ключів образів на основі декомпозиції їх множини // Відбір і обробка інформації. − 2006. – Вип. 24(100). – С. 110−114.
7. *Мельник Р. А., Тушницький Р. Б.* Каскадна декомпозиція множин великої розмірності при кластеризації ключів образів // Комп'ютерні науки та інформаційні технології. – 2008. – № 604. – С. 249−254.

Національний університет "Львівська політехніка"

Одержано 17.02.2009