по результатам вычисления статических характеристик шероховатости поверхностей при измерениях ординат профиля шероховатости индукционными контактными датчиками с расширенным динамическим диапазоном, перемещающихся по измерительной линейной базе. Обоснована возможность двумерного анализа шероховатости для аппроксимация тренда волнистости по результатам вычисления параметров шероховатости простых поверхностей протяженных изделий в авиастроении.

Литература

1. Мирошниченко, И. В. Формирование математической модели волнистости по результатам вычисления шероховатости протяженных изделий [Текст] / И. В. Мирошниченко // Технологический аудит и резервы производства. — 2014. — № 2/1(16). — С. 11–15. doi:10.15587/2312-8372.2014.23433
2. Детлинг, В. С. Інформаційно-вимірювальна система забезпечення якості шорсткості поверхні [Текст] / В. С. Детлинг, В. П. Зинченко, И. В. Мирошниченко // Вісник Черкаського Державного технологічного університету. — 2006. — Спецвипуск. — С. 135–137.
3. Гагарин, А. А. Шероховатость как геометрическая характеристика поверхностей изделий авиационной техники [Текст]: материалы конференции / А. А. Гагарин, И. В. Мирошниченко // XI Міжнародна науково-технічна конференція АВІА-2013, 21–23 травня 2013 р., Київ. — Том 1. — С. 1.3–1.16.
4. Марчук, М. О. Проблематика розробки інформаційних технологій контролю якості шорсткості поверхні [Текст] / М. О. Марчук, І. В. Мірошниченко // Технологічні комплекси. — 2012. — № 1, 2(5, 6). — С. 57–61.
5. Мирошниченко, И. В. Вероятностные характеристики статической математическо модели шероховатости [Текст] / И. В. Мирошниченко // Вісник Східноукраїнського національного університету імені Володимира Даля. — 2013. — № 4(193), Ч. 2. — С. 109–113.
6. Мирошниченко, И. В. Математические модели геометрических характеристик поверхности протяженных объектов [Текст] / И. В. Мирошниченко // Адаптивні системи автоматичного управління. — 2013. — Вип. 1(22). — С. 45–55.
7. Detling, V. S. Information-logical model error of random statistical characteristics measurements [Text] / V. S. Detling, C. Kartunov, I. V. Miroshnichenko // International scientific conference, Gabrovo, 23–24 Nov. 2007. — P. 322–327.
8. Скоростная оптическая система измерения шероховатости SORM 3plus [Электронный ресурс]. — Режим доступа: \www/ URL: http://www.emg-automation.com/nc/cn/automation/qs-systems/online-roughness-measurement-sorm-3plus/action/open-download/download/brochure-sorm-rus/
9. Зинченко, В. П. Алгоритм оптимального проектирования самолетов [Текст] / В. П. Зинченко, В. В. Борисов, И. В. Мирошниченко // Вісник Східноукраїнського національного університету імені Володимира Даля. — 2011. — № 13(167). — С. 70–74.
10. Мирошниченко, И. В. Об одном способе классификации статистических измерительных задач [Текст] / И. В. Мирошниченко // Математичне та комп'ютерне моделювання. — 2012. — Вип. 7. — С. 132–139.
11. Дороговець, М. Основи метрології та вимірювальної техніки [Текст]. Том 1. Основи метрології / М. Дороговець, В. Мотало, Б. Стадник. — Львів: Львівська політехніка, 2005. — 650 с.

СТАТИЧНІ І ДИНАМІЧНІ МАТЕМАТИЧНІ МОДЕЛІ ШОРСТКОСТІ ПРОТЯЖНИХ ПОВЕРХОНЬ

Показана можливість формування динамічної математичної моделі шорсткості — хвилястості для простих поверхонь за результатами оцінок статичних математичних моделей шорсткості поверхонь, отриманих за результатами вимірювань ординат профілю шорсткості індукційними контактними датчиками з розширеним динамічним діапазоном, що переміщаються по незалежній лінійній базі.

Ключові слова: статистична математична модель, динамічна математична модель, шорсткість, хвилястість.

Мирошниченко Иван Владимирович, *старший преподаватель, кафедра автоматизации проектирования энергетических процессов и систем, Национальный технический университет Украины «Киевский политехнический институт», Украина,* ***e-mail: goodgod@ukr.net****.*

Мірошниченко Іван Володимирович, *старший викладач, кафедра автоматизації проектування енергетичних процесів та систем, Національний технічний університет України «Київський політехнічний інститут», Україна.*

Miroshnichenko Ivan, *National Technical University of Ukraine «Kyiv Polytechnic Institute», Ukraine,* ***e-mail: goodgod@ukr.net***

УДК 004.825
DOI: 10.15587/2312-8372.2015.37422

Волосюк Ю. В.

МЕТОДИ ПАРАЛЕЛЬНОЇ РЕАЛІЗАЦІЇ АЛГОРИТМІВ КЛАСТЕРИЗАЦІЇ ТЕКСТОВИХ ДАНИХ

Розглянуто загальний алгоритм організації паралельних обчислень. Наведено особливості організації процесу паралельних обчислень, визначено критерії, що вказують на здатність алгоритму до представлення в паралельному вигляді. Розглянуто програмні засоби для розпаралелювання алгоритмів та розроблена версія алгоритму Maximin, побудована на основі паралельних обчислень. Отримані в роботі результати підтвердили доцільність використання паралельної реалізації зазначеного алгоритму.

Ключові слова: *паралельні обчислення, кластеризація, Maximin, алгоритмізація, продуктивність.*

1. Вступ

Стрімке зростання обсягу інформації, представленому в електронному вигляді в сучасному інформаційному просторі, спонукає до більш поглибленого вирішення задачі кластеризації документів. Завданням кластеризації є розподіл множини документів на заздалегідь невідому кількість підмножин.

При залученні в процес нового документа необхідно виконати його віднесення до однієї з існуючих груп або створити нову, в якій документ, що додається, буде центроїдом. Процес додавання нового документа має невелику обчислювальну складність при відомих метриках, на основі яких виконується розрахунок міри подібності документа з документами з груп відомих тематик. Зазначений спосіб віднесення документів має свої недоліки: неможливо нескінченно довго додавати нові елементи без урахування параметрів всіх документів, наявних у вибірці, що поділяється. З іншого боку, періодичне виконання процесу розбиття на кластери є завданням, складність якого зростає зі зростанням кількості документів, що оброблюються.

На даний час в процесі вирішення задачі кластеризації великий інтерес представляє використання апаратних конфігурацій, побудованих на багатоядерних центральних процесорах та кластерах обчислювальних вузлів. Для таких обчислювальних систем актуальною є проблема оптимізації обчислень через те, що більшість програм виконується синхронно в межах одного процесу, що не дає можливості максимально використовувати ресурси комп'ютера та мінімізувати час простою усіх ядер процесору. Тому постає актуальне завдання щодо вдосконалення алгоритмів кластеризації, які б ефективно виконувались на системах з паралельною архітектурою [1].

2. Аналіз літературних даних та постановка проблеми

Загальні принципи паралельної реалізації алгоритмів інтелектуального аналізу даних були описані в роботах І. І. Холода та З. О. Каршиєва [2], варіанти розпаралелювання алгоритмів кластеризації розглянуті у роботах О. О. Островського [3], Є. В. Котельникова [4], В. Б. Барахніна [5] та інших. Також була успішно застосована апаратно-програмна архітектура CUDA для створення паралельних версій алгоритмів кластеризації [6]. Відповідно, проблема впровадження паралельної реалізації алгоритмів кластеризації висувається в ряд актуальних.

3. Об'єкт, мета і завдання дослідження

Об'єкт дослідження — паралельна реалізація процесу кластеризації текстових даних.

Метою дослідження є оцінка ефективності паралельної реалізації алгоритму кластеризації даних.

Завданнями, вирішення яких може вважатися необхідним для досягнення поставленої мети, були обрані:

1) дослідження принципів організації паралельних обчислень та визначення критеріїв, що вказують на здатність алгоритмів до представлення в паралельному вигляді;

2) вирішення завдання кластеризації за допомогою паралельних обчислень з використанням одного з алгоритмів;

3) проведення експерименту та порівняльний аналіз ефективності використання паралельних обчислень в реалізації процесу кластеризації.

4. Організація паралельних обчислень

Загальний принцип багатопроцесорної обробки як спосіб підвищення ефективності процесу кластеризації можна описати наступним чином: розподілити обчислювальний процес на N визначених вузлів, які здійснюють окремі фрагменти обчислювального алгоритму. Виконати завдання реалізації паралельних обчислень можливо при наявності завдань з багатократним повторюванням обчислень при варіаціях деяких початкових умов для кожного циклу обчислень. Зазначимо, що при цьому бажано, щоб в таких завданнях параметри наступних циклів обчислень мали мінімально виражену залежність від результатів попередніх циклів, а також необхідно виключи конфлікти в процесі читання/запису даних.

Переважна більшість алгоритмів, які використовуються на даний час для задач кластеризації і категоризації документів, є послідовними і використовують наявні обчислювальні ресурси не на повну потужність. Вони мають обмеження, які не дозволяють виконувати процес обробки даних в декілька обчислювальних потоків. По-перше, це чітка послідовність виконуваних операцій і обов'язкове завершення попереднього логічного етапу обробки до початку наступного. По-друге, всі обчислення виконуються на основі даних, отриманих при аналізі всього масиву оброблюваних документів, що, в свою чергу, досить сильно ускладнює реалізацію, а також підвищує накладні витрати, пов'язані з передачею інформації між вузлами системи [7].

При великій кількості оброблюваних документів дуже сильно зростає час виконання процесу кластеризації, і цілком виправданою метою є прискорення обробки. При виконанні кластеризації документів доцільно виділити наступні етапи роботи [3]:

1. Збір і завантаження вихідних даних. Цей процес залежить від джерела даних. Це можуть бути дані в текстових форматах, які добре піддаються обробці і завантаженню, імпорт із зовнішніх ресурсів, який залежить від продуктивності зовнішнього сервера, а також від пропускної здатності мережі, або імпорт з форматів, обробка яких має деяку специфіку. Оцінювати трудомісткість даної процедури і можливість впровадження паралельної реалізації на цьому етапі потрібно по кожному конкретному випадку.

2. Етап підготовки. Включає первинну обробку документів. У випадку обробки текстових даних до цього етапу входять процеси виділення ключових термів та словосполучень з тексту документів, а також обчислення міри подібності між документами. На даному етапі впровадження паралельної обробки може призвести до істотних переваг в продуктивності. Зазначимо, що дії з аналізу змісту є незалежними один від одного і не вимагають наявності повної інформації на всіх обчислювальних вузлах.

3. Процес кластеризації. Виходячи з методики роботи алгоритмів кластеризації, які задовольняють умови їх паралельної реалізації, слід зазначити, що найбільш складним обчислювальним процесом є обхід всіх об'єктів вибірки і перевірка кожного на роль стовпа. Зрозуміло, що даний процес можливо вдало реалізувати паралельно, хоча він і вимагає наявності інформації про відстані між об'єктами на всіх обчислювальних вузлах.

4. Візуалізація результатів. Це допоміжний етап, яких полегшує роботу з вихідними даними і отриманими

кластерами. Для реалізації цього етапу немає потреби в великих обчислювальних потужностях для роботи, незалежно від кількості документів у вибірці. Оптимізація роботи на даному етапі досягається оптимізацією на рівні зберігання даних, тобто на рівні сервера бази даних.

5. Адаптація алгоритму до паралельної реалізації обчислень

Розглянемо загальний алгоритм організації паралельних обчислень [8]. Нехай існує колекція даних a розмірності m. Необхідно для кожного елемента a_i розрахувати новий елемент, використовуючи функцію $f(a_i)$, і сформувати послідовність b розмірністю m.

1. Основний потік програми отримує на вході масив a і функцію f.
2. Виконання основним потоком програми запиту до операційної системи про інформацію щодо ресурсів: кількості встановлених ядер/процесорів.
3. На основі кожного ядра/процесора створюється потік.
4. Основний потік призупиняє свою роботу і передає керування пулу потоків.
5. Пул потоків вибирає по черзі всі потоки і видає кожному на обробку черговий елемент масиву.
6. Пул чекає доти який-небудь з потоків не звільниться, щоб завантажити його даними для обробки. Цей процес триває до моменту закінчення елементів масиву.
7. Основний потік програми об'єднує результати і готовий виконувати наступні операції.

Для реалізації паралельних обчислень в роботі було обрано алгоритм Maximin. В зазначеному алгоритмі автоматичний пристрій самостійно встановлює кластери, на які поділяється вихідна множина об'єктів. Для розподілу даних необхідно визначити критерії при умовах, коли невідомі ані класи, ані їх ознаки та кількість. Тому процес організовується таким чином, щоб серед усіх можливих варіантів групувань знайти такий, коли групи мають найбільшу компактність і при цьому максимально відмінні одна від одної. Необхідним мінімумом інформації для реалізації таких алгоритмів є дані для призначення словника ознак об'єктів [9].

В алгоритмі Maximin вхідні дані задані n векторами, які потребують розподілу на кластери, виконуючи пошук представників елементів кожного кластеру, кількість яких заздалегідь невідома. На першому кроці випадково визначається центр першого кластера, потім максимально віддалений від нього об'єкт стає центром другого кластера. Об'єкти поділяються на два кластери за критерієм мінімальної відстані до центру кластера. На наступному кроці в кожному кластері визначається максимально віддалений від центру об'єкт та із знайдених об'єктів обирається максимум серед максимумів. Якщо ця відстань виявилась більшою, ніж половина середньої відстані між центрами всіх побудованих кластерів, то відповідний йому об'єкт стає центром наступного кластера.

Для впровадження паралельного виконання зазначеного алгоритму було використано процесор з багатоядерною архітектурою і програмна реалізація алгоритму на мові C# з використанням бібліотеки TPL, яка динамічно масштабує ступінь паралелізму для найбільш ефективного використання всіх доступних процесорів системи.

Для того, щоб розпаралелити зазначений алгоритм, необхідно, щоб в кожний момент часу могли одразу виконуватися n операцій, де n — кількість ядер/процесорів, тобто необхідно створити n-розмірну множину асинхронних операцій. При цьому істотний ефект буде отримано лише у тому випадку, якщо це буде не набір одиничних операцій для кожного пристрою, а набір послідовностей. Це обумовлено тим, що при паралельній обробці даних над однією колекцією все рівно зберігається ризик простою обчислювачів, відповідно, у разі відсутності даних для обробки виникає ситуація простою у зв'язку з неможливістю переходу до наступної логічної операції через те, що виникає потреба в часі на синхронізацію результатів.

Очевидно, що найбільш ефективним способом паралельної реалізації зазначеного алгоритму є запуск пулу асинхронних потоків-обробників для окремої колекції. При цьому на підсумкову продуктивність буде в значній мірі впливати кількість синхронних операцій.

Принципи розпаралелювання алгоритму Maximin:

1. Всі об'єкти кластеризуються: планувальник, працюючи з об'єктами і потоками, по черзі видає кожному вільному потоку новий об'єкт, за рахунок чого вдається збільшити продуктивність.
2. Коли визначені всі класи з наборами своїх об'єктів, планувальник може оброблювати об'єкти потоками. Це дозволяє виграти у продуктивності, тому що асинхронність операцій проявляється на рівні окремого об'єкта [10].

Паралельна реалізація алгоритму Maximin:

1. Випадково генерується перший центр класу.
2. Створюється другий центр класу, максимально віддалений від першого.
3. Для кожного об'єкта обирається відстань до найближчого центру. Для кожного об'єкта виділяється окремий потік, завдяки чому отримуємо перевагу в продуктивності на даному кроці.
4. З мінімальних відстаней обирається максимальне значення.
5. Виконується перевірка, чи є більшою максимальна відстань, ніж половина середньої відстані між існуючими центрами. І якщо результат позитивний, то новим центром стає об'єкт, для якого було обрано максимальну відстань, надалі відбувається перехід до кроку 3. В іншому випадку кластеризація завершується. На даному кроці паралельну реалізацію можливо реалізувати тільки для операцій пошуку і обчислення середньої відстані, що дає невеликий приріст продуктивності.

6. Аналіз результатів проведених експериментів

В якості об'єктів для подальшого розбиття на множини були обрані текстові дані, тому що це один з найбільш очевидних прикладів використання алгоритму Maximin. Для того, щоб абстрагуватися від конкретної предметної області, в текстах були виділена множина ознак, а кожен текст був представлений у вигляді вектора. Далі, текст будемо називати об'єктом, тому що з погляду розв'язуваної задачі на рівні алгоритмів робота ведеться виключно з об'єктами без прив'язки до будь-якої предметної області.

Оцінка продуктивності проводилася для обраного алгоритму в однопотоковому і багатопотоковому ре-

жимах з різними наборами вихідних даних. Кількість об'єктів змінювалась від 50000 до 1600000, кількість класів — від 2 до 16.

Дослідження проводилися на чотирьохядерному процесорі Core i5 3570K для одного, двох і чотирьох потоків в пулі.

На рис. 1 відображені результати паралельної реалізації алгоритму Maximin. На осі абсцис наведені кількість документів, а на осі ординат викладена кількість секунд, витрачених на виконання процесу кластеризації. Перший старт модуля виконувався для 50000 об'єктів, другий — для 100000, третій — 200000, четвертий — для 400000, п'ятий — для 800000 і шостий — для 1600000 об'єктів.

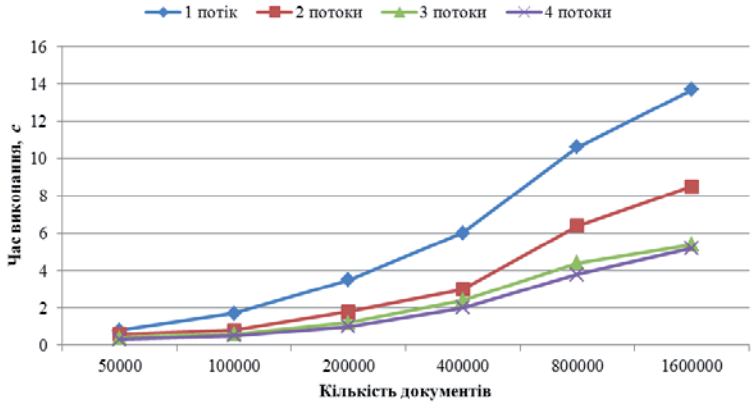

Рис. 1. Час кластеризації даних за допомогою алгоритму Maximin при різних наборах вхідних даних

В середньому час кластеризації даних за допомогою алгоритму Maximin щодо одного потоку зменшився на 58 % для двох потоків, на 115 % — для трьох і на 147 % — для чотирьох потоків. Доведено, що із збільшенням кількості об'єктів кластеризації збільшується ефект від паралельної реалізації обчислень.

7. Висновки

Дослідження одного з алгоритмів кластеризації на предмет можливості його паралельної реалізації дозволили сформулювати наступні висновки:

— для обраного алгоритму можливо реалізувати паралельні обчислення, оскільки в ньому присутні мінімум дві операції з некорелюючими результатами;

— паралельна реалізація обчислень демонструє зменшення часу виконання алгоритму вже при двох процесорах;

— збільшення продуктивності алгоритму лінійно залежить від збільшення числа обчислювачів;

— із збільшенням кількості об'єктів кластеризації збільшується продуктивність паралельних обчислень.

Таким чином, отримані результати підтвердили доцільність впровадження паралельної реалізації обчислень в алгоритмах кластеризації текстових даних, що збільшує ефективність процесу кластеризації.

Література

1. Шпаковский, Г. И. Реализация параллельных вычислений: кластеры, многоядерные процессы, грид, квантовые компьютеры [Текст] / Г. И. Шпаковский. — Минск: БГУ, 2010. — 155 с.
2. Холод, И. И. Методика распараллеливания алгоритмов интеллектуального анализа данных [Текст] / И. И. Холод, З. А. Каршиев // Известия СПбГЭТУ «ЛЭТИ». — 2013. — № 3. — С. 38–45.
3. Островский, А. А. Реализация параллельного выполнения алгоритма FCM-кластеризации [Текст] / А. А. Островский // Прикладная информатика. — 2009. — № 2. — С. 101–106.
4. Пескишева, Т. А. Параллельная реализация алгоритма обучения системы текстовой классификации [Текст] / Т. А. Пескишева, Е. В. Котельников // Вестник УГАТУ. — 2011. — Т. 15, № 5(45). — С. 130–136.
5. Барахнин, В. Б. Оценка эффективности метода параллельной реализации процесса кластеризации текстовых документов на основе алгоритма Fris-Cluster [Текст] / В. Б. Барахнин // Вестник НГУ. — 2012. — № 10. — С. 417–422.
6. Chang, D. Hierarchical clustering with CUDA/GPU [Text] / D. Chang, M. Kantardzic, M. Ouyang // Proceedings of ISCA PDCCS. — 2009. — P. 130–135.
7. Wang, H. Equivalence Class Based Parallel Algorithm for Mining MFI [Text] / H. Wang // Applied Mechanics and Materials. — 2015. — Vol. 713–715. — P. 1712–1715. doi:10.4028/www.scientific.net/amm.713-715.1712
8. Борисова, И. А. Использование FRiS-функций для решения задачи SDX [Текст] / И. А. Борисова, Н. Г. Загоруйко // International Conference «Classification, Forecasting, Data Mining» CFDM. — Varna, 2009. — P. 110–116.
9. Барсегян, А. А. Анализ данных и процессов [Текст]: учеб. пособие / А. А. Барсегян, М. С. Куприянов, И. И. Холод, М. Д. Тесс, С. И. Елизаров. — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2009. — 512 с.
10. Троелсен, Э. Язык программирования C# 2010 и платформа NET 4 [Текст]: пер. с англ. / Э. Троелсен. — 5-е изд. — М.: Вильямс, 2011. — 1392 с.

МЕТОДЫ ПАРАЛЛЕЛЬНОЙ РЕАЛИЗАЦИИ АЛГОРИТМОВ КЛАСТЕРИЗАЦИИ ТЕКСТОВЫХ ДАННЫХ

Рассмотрен общий алгоритм организации параллельных вычислений. Приведены особенности организации процесса параллельных вычислений, определены критерии, указывающие на способность алгоритма к представлению в параллельном виде. Рассмотрены программные средства для распараллеливания алгоритмов и разработана версия алгоритма Maximin, построенная на основе параллельных вычислений. Полученные в работе результаты подтвердили целесообразность использования параллельной реализации указанного алгоритма.

Ключевые слова: параллельные вычисления, кластеризация, Maximin, алгоритмизация, производительность.

Волосюк Юрій Вікторович, *кандидат технічних наук, доцент, кафедра інформатики та соціально-гуманітарних дисциплін, Миколаївська філія ПВНЗ «Європейський університет», Україна,* ***e-mail: relax_eu@mail.ru****.*

Волосюк Юрий Викторович, *кандидат технических наук, доцент, кафедра информатики и социально-гуманитарных дисциплин, Николаевский филиал ЧВУЗ «Европейский университет», Украина.*

Volosyuk Yuri, *Mykolaiv branch of European University, Ukraine,* ***e-mail: relax_eu@mail.ru***