УДК 004.62, 004.67 https://doi.org/10.15407/pp2025.02.087

О.В. Царинюк, А.М. Глибовець

СЕГМЕНТАЦІЯ ГЕОПРОСТОРОВИХ РАСТРІВ: АНАЛІЗ ЧАСОВИХ ХАРАКТЕРИСТИК АЛГОРИТМУ AREAONAREAOVERLAYER

У статті досліджується продуктивність алгоритму AreaOnAreaOverlayer, що використовується для сегментації геопросторових растрів за ознаками висоти в середовищі FME. Основну увагу приділено аналізу часових характеристик алгоритму під час обробки великих обсягів даних, зокрема, в задачах класифікації рослинного покриву. Описано експериментальне середовище, типові вхідні дані та вплив геометричних параметрів полігонів на час виконання операції. Отримані результати дають змогу оцінити межі застосування алгоритму та виявити залежності між структурою вхідних даних і обчислювальною складністю.

Ключові слова: просторовий аналіз, сегментація растрів, геоінформаційні системи, часові характеристики, продуктивність алгоритму

O.V. Tsaryniuk, A.M. Hlybovets

SEGMENTATION OF GEOSPATIAL RASTERS: ANALYSIS OF THE TEMPORAL CHARACTERISTICS OF THE AREAONAREAOVERLAYER ALGORITHM

This paper investigates the performance of the AreaOnAreaOverlayer algorithm used for segmenting geospatial rasters based on elevation features within the FME environment. The main focus is on analyzing the algorithm's temporal characteristics when processing large volumes of data, particularly in vegetation cover classification tasks. The study describes the experimental setup, typical input data, and the impact of polygon geometric parameters on execution time. The results provide insight into the algorithm's application limits and reveal dependencies between the structure of input data and computational complexity.

Keywords: spatial analysis, raster segmentation, geographic information systems, temporal characteristics, algorithm performance

.

Вступ

В останні десятиліття геопросторові дані все більше інтегруються у різні сфери людської діяльності. Якщо раніше геоінформаційні системи знаходили застосування лише в окремих галузях, на кшталт логістики чи військової справи, то зараз складно знайти галузь, яка не використовує карти чи геопросторові дані. Перед виробництвом геопросторових даних постають виклики, коли потрібно створювати більш точні геодані, у стисліші терміни та за меншу собівартість.

Основним джерелом геопросторових даних є засоби дистанційного зондування Землі (ДЗЗ). До цих засобів належать супутникові знімки, аерофотозйомка, лідарна зйомка, радіолокаційна зйомка та інші. Кожен із цих засобів має свої обмеження до застосування та специфіку обробки отриманих даних. Наприклад, супутниковий знімок несе в собі спектральні характеристики об'єктів, але на ньому відсутня інформація про висотну складову, а на 3D-моделі радіолокаційної зйомки відсутня інформація про типи об'єктів, розташовані на цій моделі. Лідарна зйомка дозволяє отримати інформацію про земну поверхню у тривимірному просторі, включно із типами об'єктів. Але вона має певні обмеження, зокрема, через необхідність використання літальних апаратів із відносно вузькою смугою охоплення (до 2 км за проліт) та значний обсяг вихідних даних, який коливається в межах 100–10 000 МБ/км².

Огляд літератури

Стрімкий розвиток технологій машинного навчання уможливив вдосконалення існуючих технологій отримання геоданих. У дешифруванні супутникових знімків широко застосовуються згорткові моделі, U-Net [1], ResNet [2], що дозволило значно заощадити ресурси на ручній векторизації супутникових зображень. Математичні моделі, описані у працях Liu, et al [3,4] дозволили створювати висотні моделі місцевості (DHM) з глобальним покриттям завдяки поєднанню відкритих даних лідарної зйомки та супутникових знімків високої роздільної здатності.

Сегментація зображень є однією з найскладніших задач обробки зображень. На сьогодні існує багато підходів і методів сегментації, зокрема, методи сегментації описані в роботах Hofmann & Tiede [5] та Mueller & Corcoran [6]. Більшість досліджень у сфері сегментації рослинності зосереджені на визначенні індивідуальних крон дерев. Цей напрям відіграє ключову роль у детальному вивченні лісових екосистем, що підтверджується роботами Douss, et al [7], Li, et al [8], Lindberg, et al [9], а також Jakubowski, et al [10]. Завдяки цим дослідженням було значно поглиблено наше розуміння характеристик окремих дерев, структури лісів та розподілу біомаси.

На відміну від детального аналізу окремих крон дерев, наше дослідження спрямоване на розробку методів узагальненої сегментації, що дозволяють виділяти великі масиви рослинності з подібними (або майже однаковими) висотними характеристиками. Такі підходи є ефективними для сегментації рослинного покриву на великих територіях, наприклад, на рівні цілих країн. Це особливо важливо для аналізу рослинності, необхідного для регіональних і національних екологічних оцінок, планування землекористування та реалізації масштабних природоохоронних - заходів.

Попередня робота

У попередній частині дослідження Tsaryniuk, et al [11] було розроблено три різні підходи до сегментації геопросторових растрів: методом згорткових фільтрів, методом рандомних точок та методом гексагонів. Кожен із методів використовує різні підходи до сегментації та має свої особливості.

Метод згорткових фільтрів має перевагу у вигляді можливості обробляти великі растри цілісно, без розділення на частини. Проте його використання ускладнюється у випадках, коли потрібно працювати з окремими тайлами: на стиках виникають відмінності в даних, що унеможливлює подальше об’єднання результатів в єдиний набір. Крім того, перетворення растрового зображення у векторну форму є складним під час обробки великих об’ємів даних, а також цей метод схильний до створення полігонів малої площі, які потребують додаткової фільтрації або узагальнення.

Метод рандомних точок дозволяє паралельно обробляти окремі полігони, на яких генеруються точки, що сприяє підвищенню продуктивності. Водночас він вимагає наявності додаткового векторного шару, який підлягає сегментації, тоді як інші методи працюють безпосередньо з растровими даними. У деяких випадках полігони, які використовуються для генерації точок, можуть бути надто великими, що ускладнює та уповільнює процес - обробки.

Метод гексагонів також забезпечує можливість паралельної обробки як на рівні растрових даних, так і на рівні векторної сітки гексагонів. Він не потребує додаткових векторних шарів, на противагу методу рандомних точок. Водночас точність результатів методу гексагонів дещо нижча порівняно з методом згорткових фільтрів.

Оцінка точності цих методів показала, що всі три методи мають достатньо високий показник точності. Для подальшої роботи було обрано метод гексагонів як найперспективніший в плані обробки великих масивів даних, оскільки цей метод має низку переваг у порівнянні з іншими методами: відносно невисока складність обчислень, можливість обробки даних частинами та подальше комбінування частин в єдиний набір даних.

Оскільки нашою основною задачею є побудова алгоритму, що зможе працювати з великими даними, в цій роботі ми зосередилися на визначенні характеристик даних, що впливають на час обробки та ресурси, необхідні для успішного завершення роботи алгоритму.

Опис методу сегментації гексагонами

У нашій роботі ми вирішуємо задачу поєднання геопросторових даних за типами об'єктів з висотною складовою у великих масштабах. Одним із прикладів застосування є сегментація рослинного покриву. Для реалізації запропонованого методу сегментації геопросторових растрів було обрано програмне середовище FME (Feature Manipulation Engine) [12].

FME — це програмна платформа, розроблена компанією Safe Software, призначена для інтеграції, трансформації та конвертації просторових і непросторових даних між різними форматами та структурами. FME підтримує сотні форматів ГІС, CAD, баз даних, растрових зображень, 3D-моделей, а також різноманітні хмарні сервіси. Особливістю FME є можливість створення робочих процесів (workflow) без необхідності програмування, завдяки використанню трансформерів — спеціалізованих інструментів для обробки даних.

Трансформер в FME — це окремий функціональний блок, який виконує певну операцію над даними у процесі трансформації. Наприклад, трансформери можуть змінювати геометрію, фільтрувати об'єкти, змінювати атрибути, об'єднувати дані, аналізувати просторові зв'язки тощо. Кожен трансформер має свою спеціалізацію та набір параметрів для налаштування.

Кастомний трансформер (Custom Transformer) — це користувацький набір з одного або кількох трансформерів, згрупованих в єдиний логічний блок. Він дозволяє створювати повторно використовувані компоненти з власними вхідними та вихідними портами, які можуть бути інтегровані в інші робочі процеси FME. Кастомні трансформери часто використовуються для спрощення складних робочих процесів, підвищення читабельності схем або стандартизації окремих частин трансформації.

Наш метод сегментації гексагонами (Рис. 1) передбачає створення послідовної сітки шестикутників із заданою стороною SIDE_LENGTH, поєднання цієї сітки з матрицею DHM (Digital Height Model) та векторним шаром рослинного покриву. Для генерації гексагонів було використано кастомний трансформер HexagonSampler (Рис. 2), розроблений та опублікований Gauthier [13]. Він створює сітку шестикутників по заданій рамці (у нашому випадку в ролі рамки виступає описуючий прямокутник вхідного набору рослинності). Створення сітки шестикутників складається з наступних етапів:

- ExpressionEvaluator та ExpressionEvaluator_2 обчислюють атрибути _hoffset та _voffset на базі заданого користувачем параметра довжини сторони шестикутника — SIDE_LENGTH:

$$hoffset = SIDE\_LENGTH * 3$$

$$voffset = \cos(30°) SIDE\_LENGTH * 2$$

- 2DgridAccumulator генерує сітку точок з інтервалом _hoffset та _voffset;
- Offsetter, підключений паралельною гілкою, створює копію сітки точок і зміщує її відносно першої сітки на $\frac{\_hoffset}{2}$ та $\frac{\_voffset}{2}$;
- 2DArcReplacer перетворює точки на кола з радіусом SIDE_LENG;

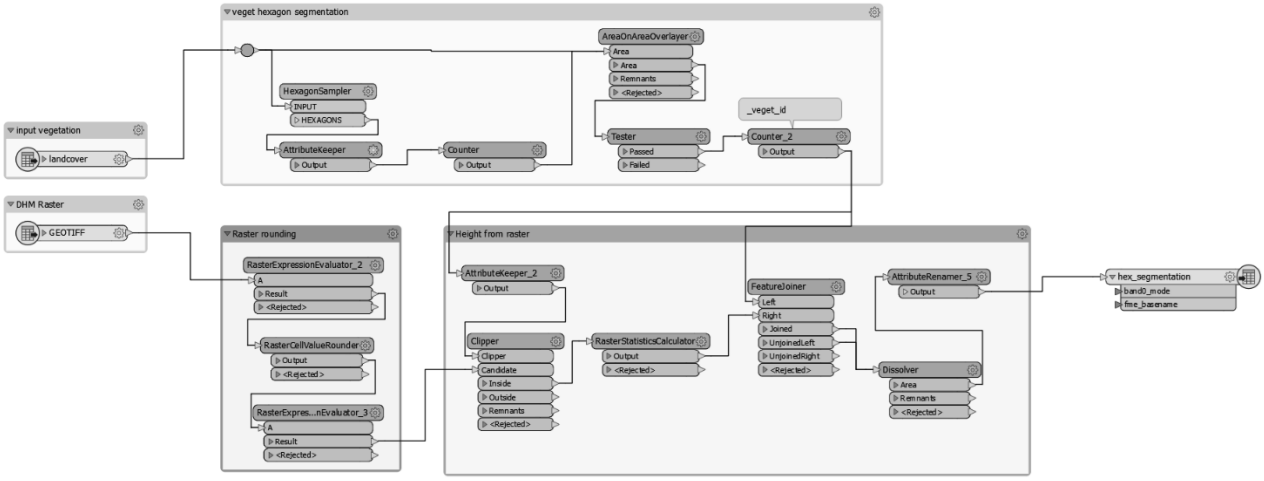

Рис. 1. Реалізація сегментації векторного набору рослинності за висотною характеристикою методом гексагонів у середовищі FME

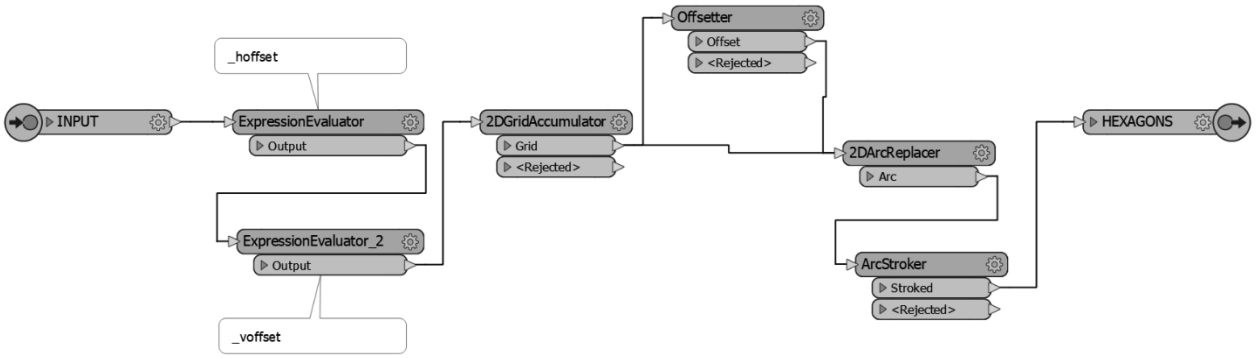

Рис. 2. Загальний вигляд кастомного модуля HexagonSampler

- ArcStroker генералізує коло до шестикутника за заданим числом інтерпольованих граней, яке дорівнює 6.

Результатом роботи HexagonSampler є послідовна нерозривна сітка правильних шестикутників з фіксованою довжиною ребра. AttributeKeeper видаляє непотрібні атрибути, створені HexagonSampler, а Counter створює унікальний id шестикутників, який надалі буде використаний для обробки великих масивів даних окремими частинами. AreaOnAreaOverlayer накладає полігони на полігони, перетинаючи геометрію та спільні атрибути. У нашій моделі він приймає на вхід два набори полігонів: вхідний вектор рослинності та сітку шестикутників. Tester відсіює непотрібні об'єкти, залишаючи лише частини гексагонів, які перетнулися з набором рослинності. Counter_2 створює унікальні id в атрибуті _veget_id, які будуть використані для перенесення значень з растра. Модуль Raster rounding (RasterExpressionEvaluator_2, RasterCellValueRounder, RasterExpressionEvaluator_3) виконує округлення значень пікселів вхідного растра до 3. Оскільки у задачі прописування висот рослинності 3 метри є допустимою похибкою, ця операція допомагає прибрати зайвий «шум» у значеннях. Clipper нарізає вхідний растр по сегментованим полігонам векторної рослинності. На цьому етапі відбувається передача значень _veget_id до растрів. RasterStatisticsCalculator обчислює мінімальне\максимальне\середнє значення пікселів кожного фрагменту растра.

FeatureJoiner відповідає за перенесення мінімального\максимального\середнього значення пікселів із растра до сегментованих полігонів рослинності. Dissolver об'єднує сегменти рослинності з однаковими значеннями висоти, а AttributeRenamer_5 перейменовує атрибути перед записом даних у вихідний файл.

Результатом роботи алгоритму hexagon_segmentation.fmw є векторний шар рослинності, представлений сегментами з різною висотою (Рис. 3).

Рис. 3. Фрагмент векторного шару рослинного покриву. До сегментації – ліворуч, сегментований методом гексагонів з довжиною сторони гексагону 20 метрів та кроком висоти 3 метри – праворуч

Аналіз складності алгоритму AreaonAreaOverlayer

Важливо зазначити, що FME є комерційним програмним забезпеченням із закритим кодом. Внутрішня реалізація алгоритмів, зокрема, методів оптимізації та індексування, залишається недоступною для користувача.

Через це обмеження ми не можемо безпосередньо проаналізувати алгоритмічну складність, переглянувши вихідний код чи документацію, тому було ухвалено рішення дослідити поведінку алгоритму експериментальним шляхом. Зокрема, ми дослідили вплив характеристик вхідних даних (кількість полігонів, їхня форма та кількість вершин) на час виконання алгоритму та використання оперативної пам'яті.

Були створені контрольовані тестові набори даних із чітко визначеними характеристиками — кількістю полігонів, кількістю перетинів та складністю геометрії (кількістю вершин). Це дозволило нам емпірично оцінити залежність часу обробки та витрат ресурсів від параметрів вхідних даних.

У дослідженні ми також детально проаналізували роботу трансформера AreaOnAreaOverlayer, який відіграє ключову роль у процесі сегментації. AreaOnAreaOverlayer виконує просторовий аналіз та створює нові об'єкти на перетинах вхідних полігонів. Немає інформації, які саме методи оптимізації використовуються у FME, але припускаємо, що там може бути застосоване просторове індексування, наприклад, R-дерева. Аналіз часу виконання алгоритму AreaOnAreaOverlayer дозволяє наблизитись до розуміння його ефективності та окреслити межі продуктивності методу гексагонів для сегментації великих масивів геопросторових даних.

Особлива увага приділялася аналізу часу виконання, враховуючи потенційне просторове індексування та геометричні операції. Для цього було створено тестові набори полігональних даних з різними

характеристиками та зроблено вимірювання продуктивності алгоритму на наборах даних розміром від 20 000 до 10 000 000 полігонів. Тестові набори даних генерувались із відомими кількостями полігонів, перетинами полігонів, вершинами полігонів. Було експериментально виміряно час роботи алгоритму AreaOnAreaOverlayer та кількість ресурсів (оперативної пам'яті), необхідних для обробки конкретного датасету.

Для перевірки гіпотези, що загальна кількість вершин полігонів має вплив на час роботи алгоритму, ми створили 2 типи наборів тестових даних: 'round' і 'squere'. Перший тип (round) – це сферичні полігони, які утворені з кола з кутом інтерполяції вершин 22,5гр. та зміщенням по осях X та Y (Рис. 4а). Другий тип (squere) – це полігони у формі квадратів, які зміщені один від одного на 1\4 довжини сторони квадрата по осях X та Y (Рис. 4б).

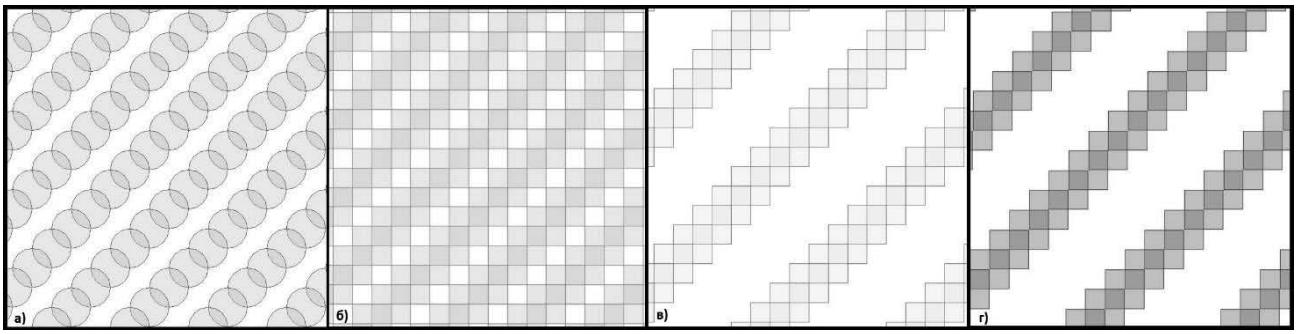

Рис. 4. Тестові набори для оцінки складності алгоритму AreaOnAreaOverlayer:
а) round_data, б) squere_data, в) v2_squere_data, г) v3_squere_data

Заміри часу та ресурсів проведені на 12 наборах кожного типу даних з загальною кількістю полігонів від 20 000 до 10 000 000. Результати замірів наведені у Таблицях 1, 2, 3, 4:

Таблиця 1

Результати замірів часу роботи та обсягу оперативної пам'яті на тестових даних (round_data)

Features_Count	Memory_KB	time_fin
20 000	252 872	2.8
100 000	911 008	13.1
1 000 000	8 323 876	151.6
2 000 000	17 502 468	316.2
3 000 000	26 613 348	541.2
4 000 000	30 941 884	804.8
5 000 000	41 899 008	1 131.4
6 000 000	46 550 676	1 547.4
7 000 000	61 031 896	2 090.4
8 000 000	65 557 424	2 691.9
9 000 000	70 041 100	3 256.1
10 000 000	74 203 936	4 191.9

Таблиця 2

Результати замірів часу роботи та обсягу оперативної пам’яті на тестових даних (squere_data)

Features_Count	Memory_KB	time_fin
20 000	334 408	3.6
100 000	1 277 492	16.5
1 000 000	12 935 884	205.8
2 000 000	22 070 072	500.4
3 000 000	32 607 576	861.8
4 000 000	47 216 188	1 373.4
5 000 000	51 416 280	2 011.2
6 000 000	71 398 116	2 861.8
7 000 000	74 950 820	3 747.1
8 000 000	104 255 576	6 227.3
9 000 000	104 426 492	9 888.2
10 000 000	104 196 532	10 370.2

Таблиця 3

Результати замірів часу роботи та обсягу оперативної пам’яті на тестових даних (v2_squere_data)

Features_Count	Memory_KB	time_fin
20 000	272 340	3.1
100 000	967 692	15.6
1 000 000	9 790 160	190.6
2 000 000	21 019 932	398.3
3 000 000	31 670 228	677.2
4 000 000	35 156 372	1 008.9
5 000 000	48 899 756	1 523.6
6 000 000	53 473 136	2 065
7 000 000	72 671 100	2 782.3
8 000 000	76 267 516	3 762.8
9 000 000	79 681 916	5 350.1
10 000 000	82 591 852	6 639.2

Таблиця 4

Результати замірів часу роботи та обсягу оперативної пам’яті на тестових даних (v3_squere_data)

Features_Count	Memory_KB	time_fin
20 000	269 380	2.2
100 000	964 524	13.1
1 000 000	7 535 360	166.5
2 000 000	16 001 076	376.4

3 000 000	23 667 180	644.8
4 000 000	34 240 536	838
5 000 000	37 579 208	1 057.7
6 000 000	51 428 532	1 565.8
7 000 000	54 452 632	2 199.7
8 000 000	58 342 284	2 874.3
9 000 000	77 111 808	3 633.3
10 000 000	80 141 216	3 421.1

Після отримання результатів було виявлено, що перший тип даних (квадрати) обробляється суттєво довше, та потребує більше оперативної пам'яті ніж другий тип, хоча має меншу загальну кількість вершин. Ми висунули припущення, що причиною такої роботи алгоритму була кількість взаємних перетинів полігонів.

Якщо вважати a,b,c,d…рядами полігонів у тестовому наборі, а 1,2,3,4,5… їхнім порядковим номером у напрямку зміщення, то, наприклад, у першому наборі полігонів round_data (Рис. 4а), об'єкт b5 має перетин лише з двома сусідніми об'єктами свого ряду b4 та b6. А полігон b5 з набору squere_data (Рис. 4б), має шість перетинів: b3,b4,b6,b7,a5,c5.

Для підтвердження гіпотези, що кількість перетинів суттєво впливає на час роботи та необхідну кількість ресурсів, ми ухвалили рішення створити ще 2 типи наборів: squere_v2 та squere_v3. У squere_v2 кількість взаємних перетинів була зменшена з 6 до 4, а у squere_v3 до двох.

У ході експерименту було виявлено, що кількість пар об'єктів, які перетинаються, мають значно більший вплив на час роботи, ніж кількість вершин у цих об'єктах. На графіках (Рис. 5) зображено відношення швидкості обробки даних до кількості об'єктів та кількості вершин. З цих графіків можна зробити висновок, що швидкість роботи алгоритму AreaOnAreaOverlayer більше залежить від кількості взаємних перетинів полігонів, ніж від кількості вершин у цих полігонах. Також швидкість обробки даних за одиницю часу суттєво зменшується зі збільшенням обсягу даних. Тому для подальшої адаптації архітектури методу сегментації для обробки великих масивів даних потрібно враховувати ці фактори.

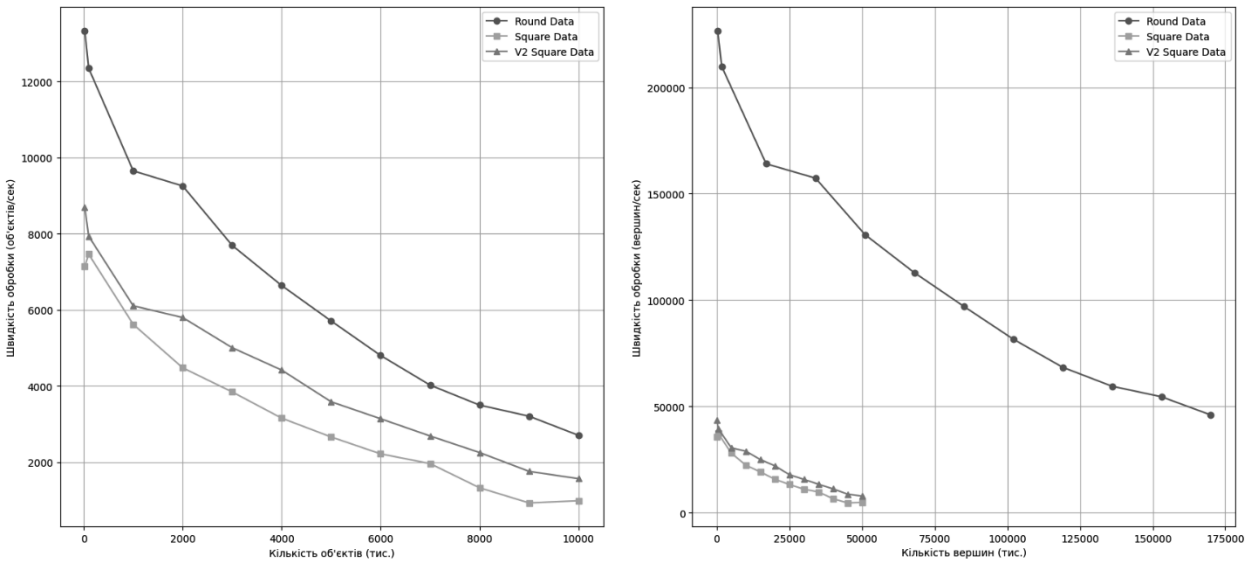

Рис. 5. Графіки відношення швидкості обробки даних до кількості об'єктів та кількості вершин

Під час обробки даних FME читає і оперує даними в оперативній пам'яті (ОП). Це дозволяє ефективно обробляти дані, але водночас створює потребу у достатньому обсязі ОП для обробки наявного масиву даних. Наскільки впливає обсяг доступної ОП на час обробки видно на прикладі тестових наборів прямокутників «squere» (Рис. 6). У разі встановлених 128Гб оперативної пам'яті на ПК, FME процес стає помітно повільнішим досягнувши 80% об'єму наявної ОП. Досягнувши ліміту по ОП, FME починає використовувати дисковий кеш (якщо він доступний в операційній системі). Додаткові процеси оптимізації використаної пам'яті, зайвий час запису\читання даних із диска пояснюють суттєве збільшення часу роботи алгоритму.

Для оцінки часової складності роботи алгоритму AreaOnAreaOverlayer ми розрахували значення кутового коефіцієнта k за отриманими графіками (Рис. 6, 7, 8).

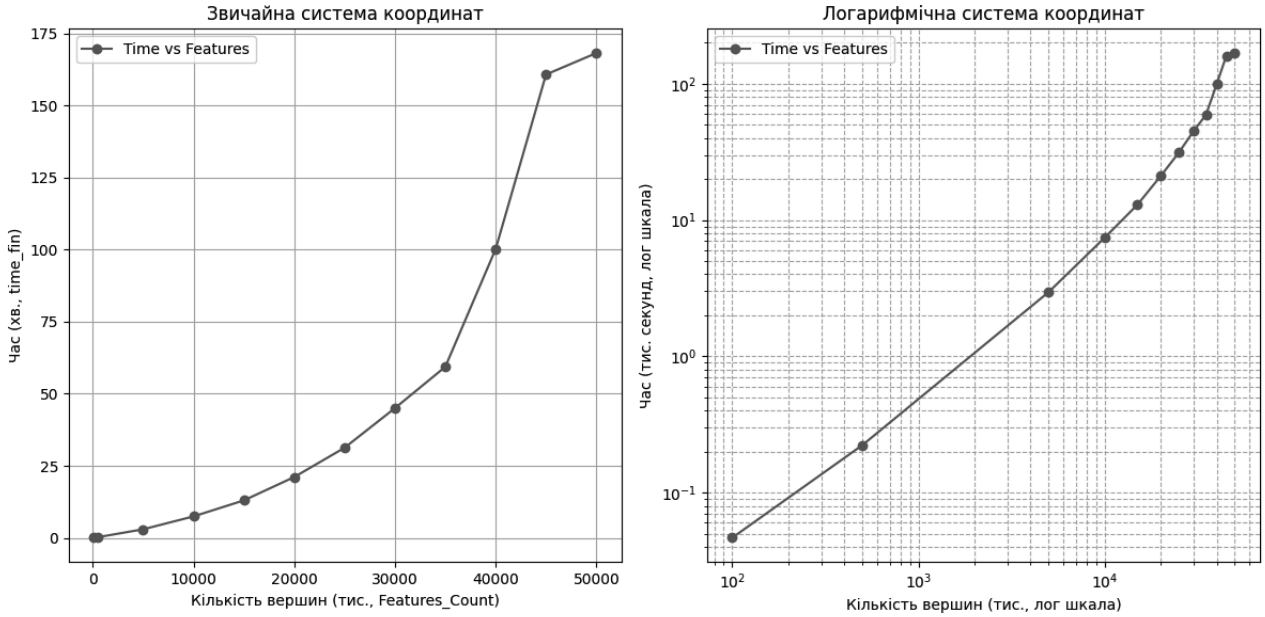

Рис. 6. Графіки часу виконання алгоритму AreaOnAreaOverlayer на наборі squere_data

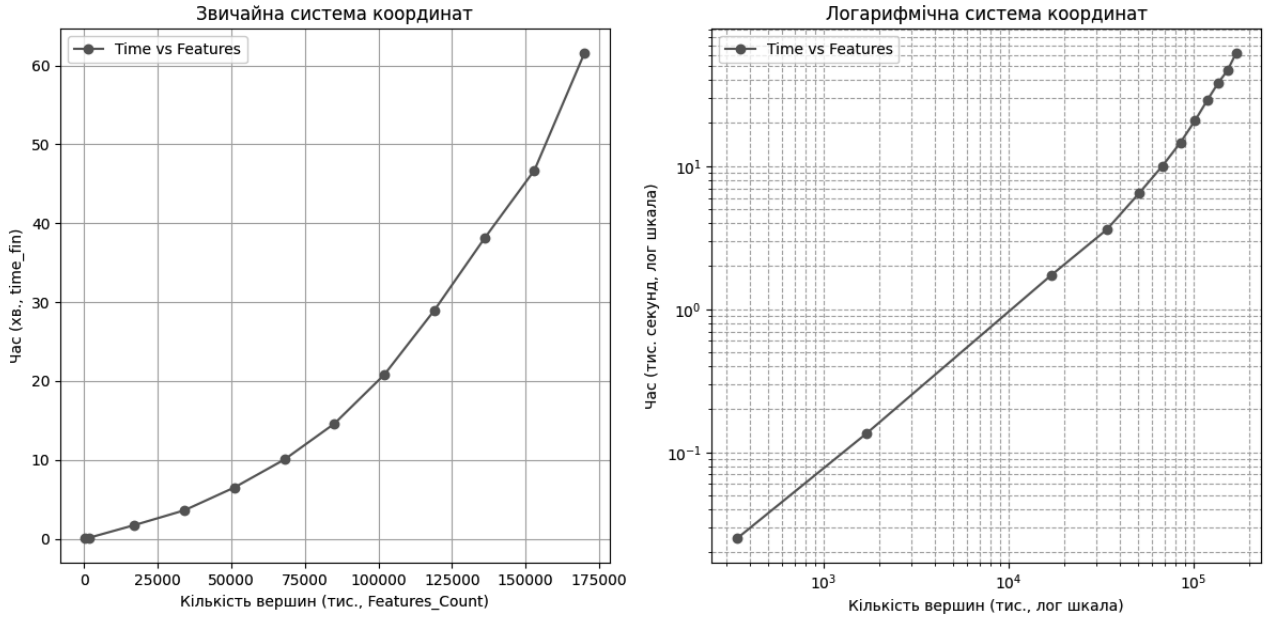

Рис. 7. Графіки часу виконання алгоритму AreaOnAreaOverlayer на наборі round_data

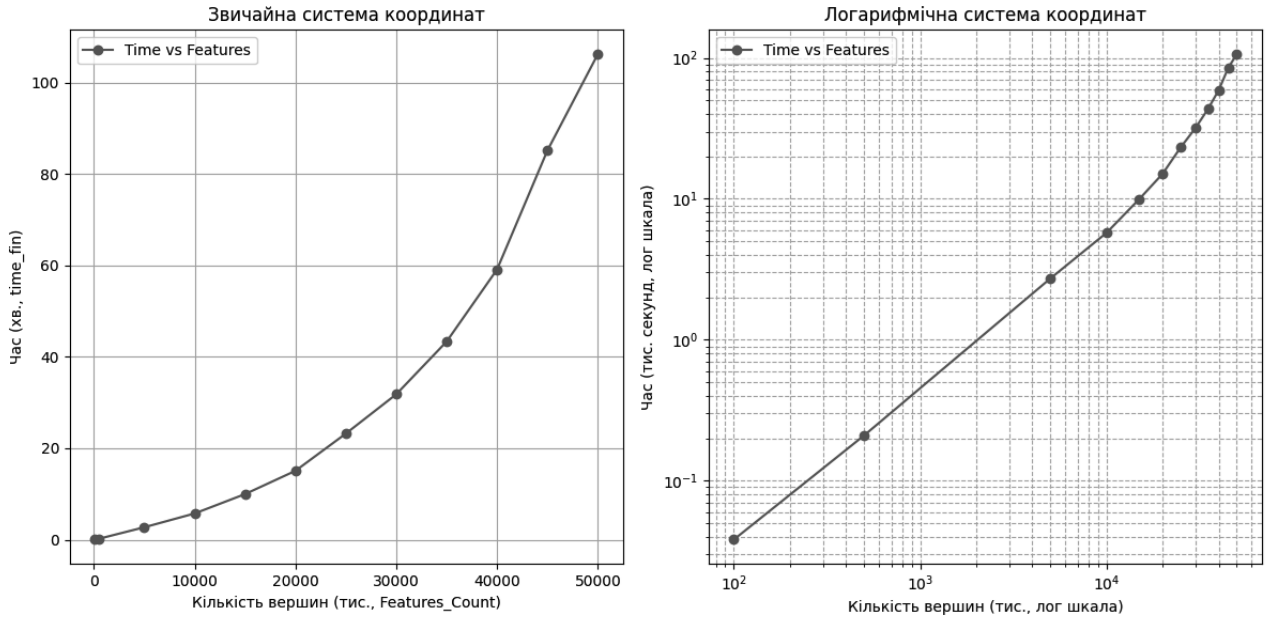

Рис. 8. Графіки часу виконання алгоритму AreaOnAreaOverlayer на наборі squere_data_v2

Розрахуємо коефіцієнт нахилу K з рівняння прямої (1) у логарифмічній системі координат:

$$Y' = KX' + B \quad (1),$$
$$X' = \log_{10}(X),$$
$$Y' = \log_{10}(Y).$$

Візьмемо з графіків дві координати з 1-го та 4-го набору: X_1=20 000, X_2=2 000 000, а значення Y_1 та Y_2 визначимо відповідно до часу роботи кожного типу наборів даних.

Формула для обчислення коефіцієнта нахилу:

$$K = \frac{X_2 - X_1}{Y_2 - Y_1}$$

Розраховані значення для різних наборів даних: round_data: K=1.0264, square_data: K=1.0715, square_data_v2: K=1.0544, square_data_v3: K=1.1166.

Середній кутовий коефіцієнт у логарифмічній шкалі за всіма наборами даних становить 1.06. Це означає, що алгоритм має майже лінійну складність.

Висновки

У цьому дослідженні було проведено аналіз алгоритму AreaOnAreaOverlayer у середовищі FME з метою оцінки часу виконання та впливу різних характеристик вхідних векторних полігонів на час обробки та використання ресурсів. Отримані результати вказують на те, що найбільший вплив на швидкість виконання алгоритму має кількість взаємних перетинів полігонів, а не їхня загальна складність (кількість вершин). Це випливає з порівняння отриманих даних за наборами round_data та square_data: набір round_data обробляється помітно швидше та потребує суттєво менше оперативної пам'яті внаслідок меншої кількості взаємних перетинів полігонів у наборі.

Іншим важливим фактором є об'єм доступної оперативної пам'яті, оскільки FME обробляє дані в оперативній пам'яті, що забезпечує швидке виконання алгоритму. Але досягнувши 80% заповнення доступної оперативної пам'яті, швидкість обробки суттєво знижується через використання дискового кешування.

Отримані результати є ключовими для подальшої розробки архітектури автоматизованого пайп-лайну по сегментації геопросторових растрів. Попри майже лінійну складність алгоритму, немає потреби у розділенні вхідного набору на частини. Однак варто враховувати об'єм доступної оперативної пам'яті, оскільки під час обробки великих масивів даних, ресурсів може не вистачити для успішного результату. У подальшій роботі ми плануємо дослідити залежність розміру вхідних даних від обсягу оперативної пам'яті, необхідної для роботи алгоритму, та розробити мо-

дуль, який буде відповідати за розподілення обчислень.

References

1. Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation*. http://arxiv.org/abs/1505.04597.
2. He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition*. http://arxiv.org/abs/1512.03385
3. S. Liu, et al, The overlooked contribution of trees outside forests to tree cover and woody biomass across Europe. Science Advances 9(37), 2023, doi: 10.1126/sciadv.adh4097.
4. Liu S, Brandt M, Nord-Larsen T, Chave J, Reiner F, Lang N, Tong X, Ciais P, Igel C, Pascual A, Guerra-Hernandez J, Li S, Mugabowindekwe M, Saatchi S, Yue Y, Chen Z, Fensholt R. The overlooked contribution of trees outside forests to tree cover and woody biomass across Europe. Sci Adv. 2023 Sep 15;9(37):eadh4097. doi: 10.1126/sciadv.adh4097. Epub 2023 Sep 15. PMID: 37713489; PMCID: PMC10881069.
5. P. Hofmann, D. Tiede,Image segmentation based on hexagonal sampling grids, South-Eastern European Journal of Earth Observation and Geomatics 3, 2014 pp. 173-177.
6. J.N. Mueller, J.N. Corcoran, A Random Point Initialization Approach to Image Segmentation with Variational Level-sets. 2021, http://arxiv.org/abs/2112.12355.
7. R. Douss, I.R Farah, Extraction of individual trees based on Canopy Height Model to monitor the state of the forest. Trees, Forests and People 8, 2022, doi: 10.1016/j.tfp.2022.100257
8. W. Li, Z. Niu, S. Gao, N. Huang, H. Chen, Correlating the horizontal and vertical distribution of LiDAR point clouds with components of biomass in a Picea crassifolia forest. Forests 5(8), 2014, pp. 1910–1930. doi: 10.3390/f5081910.
9. E. Lindberg, J. Holmgren, H. Olsson, Classification of tree species classes in a hemi-boreal forest from multispectral airborne laser scanning data using a mini raster cell method. International Journal of Applied Earth Observation and Geoinformation 100, 2021, doi: 10.1016/j.jag.2021.102334.
10. M.K. Jakubowski, W. Li, Q. Guo, M. Kelly, Delineating individual trees from lidar data: A comparison of vector- and raster-based segmentation approaches. Remote Sensing 5(9), 2013, pp. 4163–4186. doi: 10.3390/rs5094163.
11. Tsaryniuk, O., Hlybovets, A., & Oletsky, O. (2023). *Automated Pipelines for Large-Scale Height-Based Vegetation Segmentation*.
12. Safe Software, FME: Feature Manipulation Engine[cited 04.05.2025]: https://www.safe.com/fme.
13. FME Hub, HexagonSampler, [cited 04.05.2025]: https://hub.safe.com/publishers/larry/transformers/hexagonsampler.

Одержано: 06.05.2025
Внутрішня рецензія отримана: 14.05.2025
Зовнішня рецензія отримана: 15.05.2025

Про авторів:
Царинюк Олександр Васильович,
PhD Комп'ютерні науки, аспірант.
https://orcid.org/0000-0003-1394-2040

Глибовець Андрій Миколайович,
доктор технічних наук, професор,
декан.
https://orcid.org/0000-0003-4282-481X

Місце роботи авторів:
Національний університет
«Києво-Могилянська академія»
https://www.ukma.edu.ua/