

Vegetation change detection based on time series analysis by Apache Spark and RasterFrame



Dung Mai Thi Nguyen*, Thu Hoai Thi Vu

Hanoi University of Mining and Geology, Hanoi, Viet Nam

ARTICLE INFO

Article history:
Received 18th Sept. 2020
Revised 09th Jan. 2021
Accepted 02nd Feb. 2021

Keywords:

Apache Spark,
MODIS,
NDVI,
RasterFrames,
Spatial bigdata,
Time series analysis.

ABSTRACT

Spatial big data has a large scale and complex, therefore, it cannot be collected, managed, and analyzed by traditional data analytic software shortly. These platforms in many situations are restricted to vectors data. However, the raster data generated by the sensors on the enormous number of satellites now needs to be processed in parallel on the cluster environment. The article introduces the satellite image data analyzing method using the RasterFrames library on the Apache Spark platform. The RasterFrames library examines raster data for Python, Scala, and SQL, bringing the power of Spark DataFrames to access to Earth Observation, cloud computing, and data science. In the experimental part, the NDVI and the change in the average value of NDVI in the time series are calculated to demonstrate the vegetation mantle changes in Phu Tho province. These results are the reference data source in the assessment of weather, climate, and environmental changes in the study area during that time.

Copyright © 2021 Hanoi University of Mining and Geology. All rights reserved.

*Corresponding author

E - mail: nguyenthimaidung@humg.edu.vn

DOI: 10.46326/JMES.2021.62(1).06



Tạp chí Khoa học Kỹ thuật Mỏ - Địa chất

Trang điện tử: <http://tapchi.humg.edu.vn>



Đánh giá biến động lớp phủ thực vật dựa trên phân tích chuỗi thời gian với Apache Spark và RasterFrames

Nguyễn Thị Mai Dung*, Vũ Thị Hoai Thu

Trường Đại học Mỏ - Địa chất, Hà Nội, Việt Nam

THÔNG TIN BÀI BÁO

Quá trình:

Nhận bài 18/9/2020

Sửa xong 09/01/2021

Chấp nhận đăng 02/02/2021

Từ khóa:

Apache Spark,

Dữ liệu không gian lớn,

MODIS,

NDVI,

Phân tích chuỗi thời gian,

RasterFrames.

TÓM TẮT

Dữ liệu không gian lớn có khối lượng lớn và phức tạp, không thể được thu thập, quản lý và xử lý bằng các phần mềm xử lý dữ liệu truyền thống trong thời gian ngắn. Các nền tảng xử lý dữ liệu này trong nhiều trường hợp chỉ giới hạn ở dữ liệu vector. Tuy nhiên, dữ liệu raster được tạo ra bởi các cảm biến trên số lượng lớn vệ tinh hiện nay cần được xử lý song song trên môi trường cụm. Bài báo giới thiệu phương pháp xử lý dữ liệu ảnh vệ tinh sử dụng thư viện RasterFrames trên nền tảng Apache Spark. Thư viện RasterFrames xử lý dữ liệu raster cho Python, Scala và SQL, mang sức mạnh của Spark DataFrames vào việc truy cập dữ liệu quan sát Trái đất (Earth Observation), điện toán đám mây và khoa học dữ liệu. Trong phần thực nghiệm, chỉ số thực vật NDVI và sự thay đổi giá trị trung bình của NDVI theo chuỗi thời gian đã được tính toán để chỉ ra sự biến đổi lớp phủ thực vật tại khu vực tỉnh Phú Thọ từ năm 2013-2015. Các kết quả này sẽ là nguồn dữ liệu tham khảo trong đánh giá sự biến đổi về thời tiết, khí hậu, môi trường của khu vực nghiên cứu trong khoảng thời gian đó.

© 2021 Trường Đại học Mỏ - Địa chất. Tất cả các quyền được bảo đảm.

1. Mở đầu

Tập dữ liệu hình ảnh vệ tinh rất lớn và phức tạp đến nỗi khó xử lý chúng bằng các công cụ quản lý cơ sở dữ liệu có sẵn hoặc các ứng dụng xử lý dữ liệu truyền thống. Việc phân tích dữ liệu lớn đòi hỏi các thuật toán phức tạp dựa trên kỹ thuật học máy và học sâu để xử lý dữ liệu theo thời gian thực với độ chính xác và hiệu quả cao. Truy vấn dữ liệu từ các hệ thống vệ tinh quan sát Trái đất thường gặp phải vấn đề khó khăn như: các đặc tính đa

nguồn, đa tỷ lệ, tỷ lệ lớn động và phi tuyến tính. Vấn đề nằm ở việc truy cập dữ liệu do kích thước khổng lồ của hình ảnh vệ tinh và thực tế là việc phân tích chúng đang gặp những vấn đề cần phải tháo gỡ. Đã có rất nhiều nghiên cứu về việc truy cập và phân tích dữ liệu vệ tinh, đặc biệt là trong nghiên cứu mức độ ô nhiễm ở một quốc gia, tình hình lũ lụt hoặc cháy rừng. Trong bài báo này, nhóm nghiên cứu giới thiệu một nền tảng phân tích và xử lý dữ liệu ảnh vệ tinh dựa trên cụm Apache Spark kết hợp với RasterFrames là một thư viện xử lý dữ liệu không gian địa lý cho Python và SQL. RasterFrames cung cấp chế độ hiển thị dữ liệu vào DataFrame đối với bất kỳ kiểu dữ liệu ảnh vệ tinh quan sát Trái đất, cho phép truy vấn không

*Tác giả liên hệ

E - mail: nguyenthimaidung@humg.edu.vn

DOI: 10.46326/JMES.2021.62(1).06

gian, thời gian, thực hiện các phép toán số học và tương thích với hệ sinh thái của các thuật toán Spark ML. Cụ thể, bài báo sử dụng RasterFrames và Apache Spark để tính toán chỉ số NDVI, sau đó trích xuất sự thay đổi của NDVI trong một khoảng thời gian nhất định để theo dõi sự biến đổi của lớp phủ thực vật tại khu vực nghiên cứu.

2. Cơ sở lý thuyết và vùng thực nghiệm

2.1. Nền tảng Apache Spark

Apache Spark là một nền tảng tính toán phân cụm mã nguồn mở được phát triển sơ khởi vào năm 2009 bởi Matei Zaharia tại Đại học California, Berkeley RAD Lab. Spark ban đầu được mở nguồn theo chương trình BSD (Berkeley Software Distribution) và Spark đã được trao cho Apache Software Foundation vào năm 2013, trở thành dự án cao cấp nhất của ASF vào năm 2014 và được phát triển cho đến nay. Spark có hơn 400 cộng tác viên và nhà quản lý riêng biệt từ các công ty như Facebook, Yahoo, Intel, Netflix, Databricks,... cho phép xây dựng các mô hình dự đoán nhanh chóng với việc tính toán được thực hiện trên một nhóm các máy tính, có thể tính toán cùng lúc trên toàn bộ tập dữ liệu mà không cần phải trích xuất mẫu tính toán thử nghiệm. Tốc độ xử lý của Spark có được do việc tính toán được thực hiện cùng lúc trên nhiều máy khác nhau. Đồng thời việc tính toán được thực hiện ở bộ nhớ trong (in-memories) hay thực hiện hoàn toàn trên RAM.

Apache Spark được biết đến là giải pháp hiệu quả nhất cho xử lý dữ liệu lớn và được hầu hết các ngành công nghiệp và cộng đồng chấp nhận (Databricks). Apache Spark cung cấp mô hình lập trình hỗ trợ nhiều loại ứng dụng, bao gồm ETL, học máy, xử lý luồng dữ liệu và tính toán đồ thị. Spark bổ sung hai tính năng mới cho MapReduce như lặp lại, tương tác và các ứng dụng trực tuyến. Nhờ xử lý in-memory nên Spark cung cấp các phân tích dữ liệu thời gian thực cho các chiến dịch quảng cáo, máy học (machine learning), hay các website mạng xã hội. Một trong những ưu điểm lớn nhất của Spark là tính dễ sử dụng. Spark có giao diện người dùng thân thiện. Spark cung cấp các API thân thiện cho Scala Java, Python và Spark SQL (hay còn gọi là Shark). Việc Spark được xây dựng từ các khối đơn giản sẽ giúp tạo các hàm do người dùng xác định một cách dễ dàng. Nền tảng Apache Spark là mã nguồn mở, sử dụng các server

chung, chạy trên đám mây (cloud). Spark cần một lượng lớn RAM vì nó xử lý mọi thứ ở bộ nhớ. Việc thiết lập các Spark Cluster khá tốn kém nhưng khi yêu cầu xử lý dữ liệu thời gian thực thì Spark là lựa chọn tối ưu vì chỉ cần ít hệ thống cho xử lý một lượng lớn dữ liệu với thời gian ngắn. Một lựa chọn khác để giảm chi phí là sử dụng một nhà cung cấp cho Spark như DataBricks, EarthAI hoặc chạy các quy trình EMR/Mapreduce trên đám mây với AWS.

Trong những năm gần đây, một số giải pháp xử lý dữ liệu không gian trên nền tảng dữ liệu lớn đã được công bố như MD-HBase (Nishimura và nnk., 2011), Parallel-Secondo (Lu và Guting, 2012), Hadoop-GIS (Ablimit và nnk., 2013), GeoTrellis (Kini và Emanuele, 2014), GeoMesa (Hughes và nnk., 2015), SpatialHadoop (Eldawy và Mokbel, 2015), GeoSpark (Yu và nnk., 2015) và SpatialSpark (You và nnk., 2015). Các nghiên cứu này đều tập trung xử lý các vấn đề về dữ liệu không gian địa lý lớn, tuy nhiên vẫn tồn tại sự khác biệt giữa những thách thức đặt ra và các yêu cầu kỹ thuật cụ thể.

Magellan (Ram Sriharsha) là giải pháp thực thi phân tán trong phân tích dữ liệu không gian địa lý lớn. Công cụ này được triển khai trên Apache Spark và khai thác triệt để các kỹ thuật cơ sở dữ liệu hiện đại như sắp xếp các lớp dữ liệu hiệu quả, tổng quát hóa, tìm kiếm tối ưu. Nó hỗ trợ đầy đủ các tính năng cơ bản của OpenGIS như các hàm dự báo không gian SQL, các thuật toán không gian topology. Một bộ phần mềm xử lý dữ liệu không gian lớn khác được phát triển trên nền tảng Apache Spark là SparkSpatialSDK (Shangguan và nnk., 2017), đã xem xét những đặc trưng của dữ liệu không gian, bổ sung cấu trúc dữ liệu không gian và API cho phép người dùng dễ dàng thực hiện các phép phân tích không gian với dữ liệu không gian địa lý lớn. Một số nghiên cứu đã tiến hành so sánh việc triển khai một số phép truy vấn trên cơ sở dữ liệu không gian truyền thống PostGIS/PostgreSQL và GeoSpark SQL. Kết quả chỉ ra rằng PostGIS/PostgreSQL hoạt động tốt hơn so với GeoSpark SQL trong truy vấn không gian có tính chọn lọc cao như truy vấn dạng điểm hoặc truy vấn theo cửa sổ. Nhìn chung, GeoSpark SQL hoạt động tốt hơn khi thực hiện các phép truy vấn không gian như kNN và kết nối không gian (Huang và nnk., 2017). STARK (Hagedorn và nnk., 2017), thực hiện phân tích dữ liệu không gian-thời gian

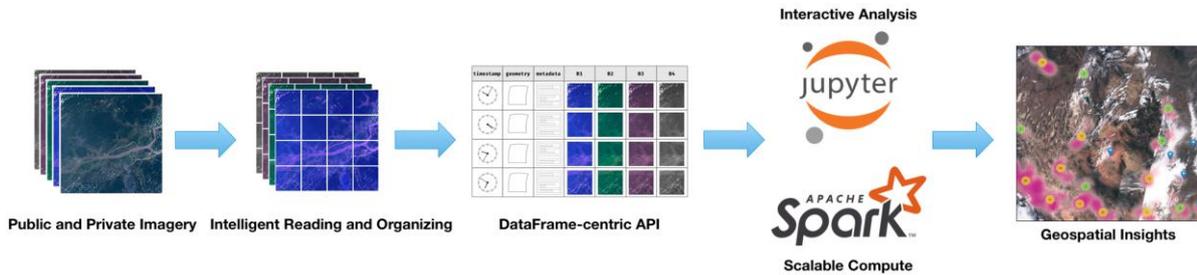
trên Spark, là một nền tảng thích hợp chặt chẽ với Apache Spark, hỗ trợ các kiểu dữ liệu không gian và thời gian cũng như các thuật toán xử lý Fei Xiao (2018) đề xuất hệ thống xử lý dữ liệu không gian lớn áp dụng cho giám sát điều kiện địa lý của Trung Quốc, gồm bốn lớp là lưu trữ dữ liệu không gian, RDDs không gian, các phép toán xử lý dữ liệu không gian và ngôn ngữ truy vấn không gian.

Mục tiêu của bài báo là nghiên cứu giải pháp xử lý ảnh vệ tinh trên nền tảng dữ liệu lớn Apache Spark. Dữ liệu ảnh vệ tinh được tải về và được xử lý bằng các công cụ Apache Spark. Sự phát triển của DataFrame đã được tiếp tục với Spark SQL, đưa DataFrames vào tính toán phân tán dữ liệu không gian lớn. Thông qua một số cải tiến mới, Spark SQL cho phép các nhà khoa học dữ liệu làm việc với DataFrames quá lớn so với bộ nhớ của một máy tính. Các DataFrames này có thể thao tác qua SQL tiêu chuẩn, cũng như các ngôn ngữ lập trình Python, R, Java, Scala (Hình 1).

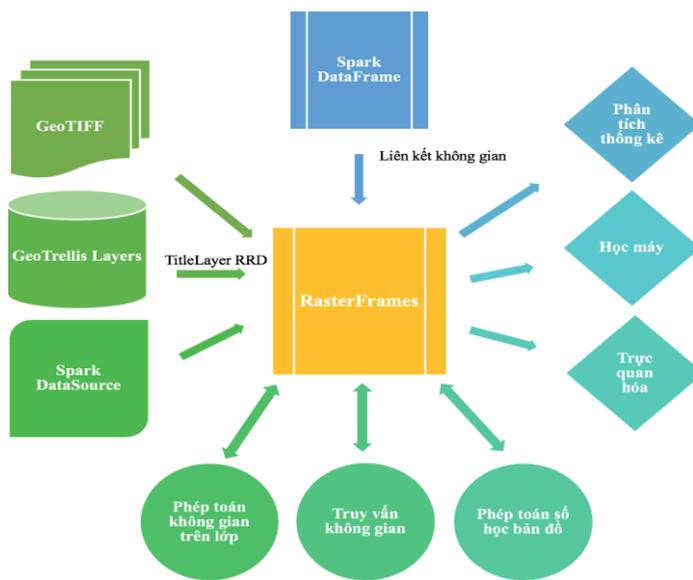
2.2. RasterFrames

RasterFrames là một dự án của Eclipse Foundation LocationTech, kết hợp phân tích dữ liệu quan sát Trái đất, điện toán đám mây và khoa học dữ liệu dựa trên DataFrame.

RasterFrames kết hợp truy cập dữ liệu quan sát Trái đất, điện toán đám mây và khoa học dữ liệu dựa trên nền tảng DataFrames. Cung cấp khả năng truy cập và hiển thị dữ liệu raster qua DataFrames, thực hiện các truy vấn không gian thời gian, các phép toán đại số trên dữ liệu raster và khả năng tương thích với các thuật toán Spark ML. Bằng cách sử dụng DataFrames như một mô hình tính toán thống nhất, RasterFrames cho phép các nhà phân tích, các nhà khoa học dữ liệu, các chuyên gia về khoa học không gian dễ dàng làm việc với dữ liệu quan sát Trái đất trong cấu trúc dữ liệu DataFrames quen thuộc (Hình 2). Ngoài ra do RasterFrames được xây dựng trên nền tảng Apache Spark, các giải pháp được thử nghiệm trên máy tính từ khai phá dữ liệu đến xử lý các tập dữ liệu lớn có thể dễ dàng được giới hạn tỷ lệ để chạy trên tài nguyên tính toán phân cụm và đám mây.



Hình 1. Xử lý dữ liệu không gian trên nền tảng Apache Spark.



Hình 2. Các thành phần cấu thành nên RasterFrames.

Thông qua Spark DataSource, RasterFrames có thể đọc các định dạng raster khác nhau - bao gồm GeoTIFF, JP2000, MRF và HDF - và từ một loạt các dịch vụ, như HTTP, FTP, HDFS, S3 và WASB. Nó cũng hỗ trợ đọc các định dạng vector GeoJSON và WKT / WKB. RasterFrame có thể thực hiện các phép lọc, chuyển đổi, tổng quát hóa, tái chia mẫu và phân loại thông qua hơn 200 hàm raster và vector.

Là một phần của dự án LocationTech, RasterFrames được xây dựng dựa trên nền tảng vững chắc được cung cấp bởi GeoMesa (phép toán không gian), GeoTrellis (phép toán raster), JTS (mô hình hình học) và SFCurve (lập chỉ mục không gian thời gian), tích hợp các khía cạnh khác nhau của các dự án này thành một thể thống nhất, phân tích dữ liệu dựa trên DataFrame. Hình 2 mô tả các thành phần cấu thành nên RasterFrames.

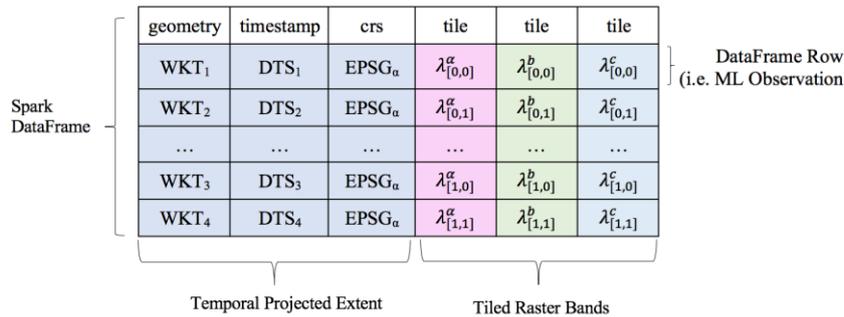
RasterFrames giới thiệu một kiểu dữ liệu riêng mới được gọi là tile cho Spark SQL. Mỗi ô tile chứa 2 ma trận 2 chiều gồm các giá trị "cell" (pixel) với thông tin giải thích số lượng các ô đó. Một "RasterFrames" là một Spark DataFrame với một hay nhiều cột kiểu tile (Hình 3). Một cột tile thường biểu thị một kênh ảnh độc lập của dữ liệu

ảnh viễn thám tương ứng với một dải sóng trong dải quang phổ, được phân tách thành từng mảng có kích thước nhất định. RasterFrames cũng hỗ trợ để làm việc với dữ liệu vector với định dạng GeoJSON. Ngoài các cột tile, còn quản lý thêm các cột geometry (giới hạn hoặc phạm vi/đường bao) xác định vị trí của dữ liệu, thông tin về hệ thống tọa độ (crs) và cột timestamp biểu thị thời gian thu nhận dữ liệu. Các cột này được sử dụng trong câu lệnh WHERE khi thực hiện truy vấn trên ảnh.

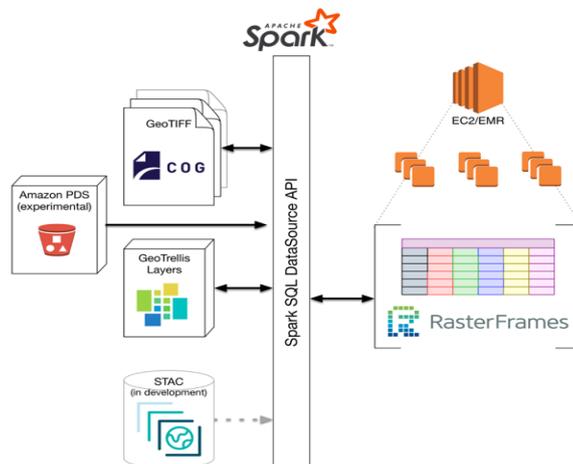
Dữ liệu raster có thể được đọc từ một số nguồn. Thông qua API Spark SQL DataSource, RasterFrames có thể được xây dựng từ các tập hợp GeoTIFFs, GeoTrellis Layers và danh mục các dữ liệu thực nghiệm từ bộ dữ liệu Landsat 8 và MODIS trên Amazon Web Services (AWS) Public Data Set (PDS) (Hình 4).

2.3. Dữ liệu và vùng thực nghiệm

Dữ liệu đầu vào sử dụng trong nghiên cứu là ảnh vệ tinh MODIS Nadir BRDF-Adjusted Surface Reflectance Data Product 500m với định dạng dữ liệu GeoTIFFs được lấy từ nguồn Amazon Web Services PDS.



Hình 3. Kiểu dữ liệu Tile sử dụng trong RasterFrame.



Hình 4. Các nguồn dữ liệu raster sử dụng trong RasterFrames.

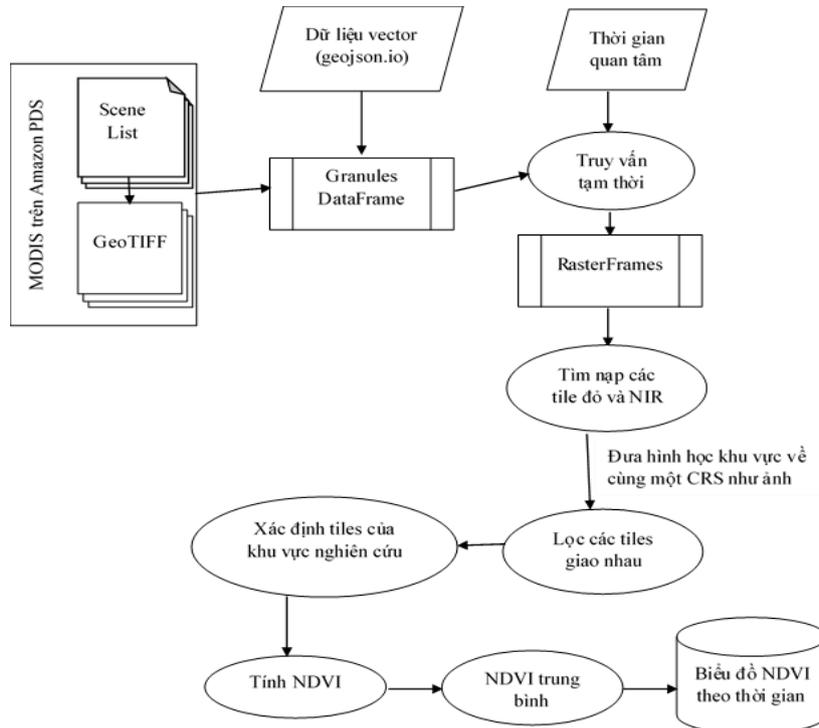
Dữ liệu ảnh MODIS được thu nhận trong khoảng thời từ tháng 01/1/2013 đến tháng 31/12/2015 cho khu vực tỉnh Phú Thọ (Hình 5).

Số lượng ảnh cung cấp bởi Amazone Web Services PDS là 2 cảnh ảnh trong một ngày và thuộc tính của ảnh sẽ được thể hiện dưới dạng lược đồ:

col_name	data_type	acquisition_date	timestamp
product_id	string	gid	string
		b01	string
		b01qa	string
		b02	string
	
		b07qa	String



Hình 5. Khu vực nghiên cứu.



Hình 6. Lưu đồ giải thuật chi tiết phân tích chuỗi thời gian của NDVI.

3. Phương pháp phân tích chuỗi thời gian

3.1. Phân tích chuỗi thời gian giá trị NDVI

Hình 6 mô tả lưu đồ giải thuật xử lý và phân tích chuỗi thời gian giá trị NDVI trên tập dữ liệu đầu vào. Phương pháp này có thể tóm tắt ở các bước chính như sau:

1. Sử dụng RasterFrames kết nối đến nguồn dữ liệu từ dịch vụ đám mây Amazon.
2. Xác định vùng dữ liệu vùng quan tâm và thời gian thu thập của dữ liệu ảnh.
3. Thực hiện các tính toán giá trị trên các DataFrames.
4. Thực hiện các phân tích theo từng chuỗi thời gian.
5. Biểu diễn các kết quả phân tích trên các biểu đồ.

3.2. Cài đặt chương trình

Ở phần này của bài báo trình bày cài đặt giải thuật phân tích chuỗi thời gian với ngôn ngữ Python, các bước chính của phương pháp được cài đặt như sau:

- Kết nối đến nguồn dữ liệu Amazon

```
path='RG_PhuTho_offset1km_1.geojson'
PT_vector=spark.read.geojson(path)
cat=spark.read.format('aws-pds-modis-catalog').load().repartition(50)
PT_cat = cat\
.filter(
(cat.granule_id == 'h27v06') &
(cat.acquisition_date >= lit('2013-01-01')) &
(cat.acquisition_date < lit('2013-12-01'))
)\
.crossJoin(PT_vector)
```

- Xác định vùng dữ liệu và thời gian thu nhận ảnh

```
raster_cols = ['B01', 'B02'] # red and near-
infrared để tính toán NDVI
PT_rf = spark.read.raster(
PT_cat.select(['acquisition_date', 'granule_id']
+ raster_cols + PT_vector.columns),
catalog_col_names=['B01', 'B02']) \
.withColumn('PT_native',
st_reproject('geo_simp', lit('EPSG:4326'),
rf_crs('B01')) \
.filter(st_intersects('PT_native',
rf_geometry('B01'))))\
```

- Tính toán NDVI

```
rf_PT_tile = PT_rf\
.withColumn('dims', rf_dimensions('B01')) \
.withColumn('PT_tile',
rf_rasterize('PT_native',
rf_geometry('B01'),'OBJECTID', 'dims.cols',
'dims.rows')) \
.persist()
rf_ndvi = rf_PT_tile \
.withColumn('ndvi',
rf_normalized_difference('B02', 'B01')) \
.withColumn('ndvi_masked', rf_mask('ndvi',
'PT_tile'))
```

- Phân tích chuỗi thời gian

```
time_series = rf_ndvi\
.groupby(
year('acquisition_date').alias('year'),
weekofyear('acquisition_date').alias('week'))\
.agg(rf_agg_mean('ndvi_masked').alias('ndvi'))
```

- Biểu diễn kết quả phân tích

```
ts_pd = time_series.toPandas()
ts_pd.sort_values(['year', 'week'],
inplace=True)
plt.figure(figsize=(20,8))
plt.plot(ts_pd['year_week'], ts_pd['ndvi'],'go-
')
```

4. Kết quả thực nghiệm

Chỉ số thực vật NDVI và sự thay đổi giá trị trung bình của NDVI trong một khoảng thời gian nhất định được tính toán để chỉ ra sự biến đổi lớp phủ thực vật tại khu vực tỉnh Phú Thọ.

NDVI thường sử dụng để theo dõi hạn hán, dự đoán sản xuất nông nghiệp, hỗ trợ dự đoán các khu vực cháy rừng và lập bản đồ sự xâm lấn sa mạc. NDVI để theo dõi thảm thực vật toàn cầu vì nó giúp làm cân bằng cho việc thay đổi điều kiện chiếu sáng, độ dốc bề mặt, hướng và các yếu tố ngoại lai khác (Lillesand 2004). Chỉ số NDVI được tính toán theo tỷ số giữa kênh sóng đỏ (Red) và kênh cận hồng ngoại (NIR).

$$NDVI = \frac{Band(NIR) - Band(Red)}{Band(NIR) + Band(Red)}$$

Quá trình xử lý dữ liệu thay vì được thực hiện trên từng dữ liệu raster độc lập, RasterFrames cung cấp khả năng xử lý trên một tập hợp các dữ

liệu raster. Tập hợp này là một danh sách các URL tham chiếu đến các tệp raster cũng có thể là Spark DataFrame, Pandas DataFrame, tệp CSV hoặc chuỗi CSV.

Thực nghiệm kết hợp dữ liệu ranh giới khu vực nghiên cứu với danh mục các tệp raster và chỉ

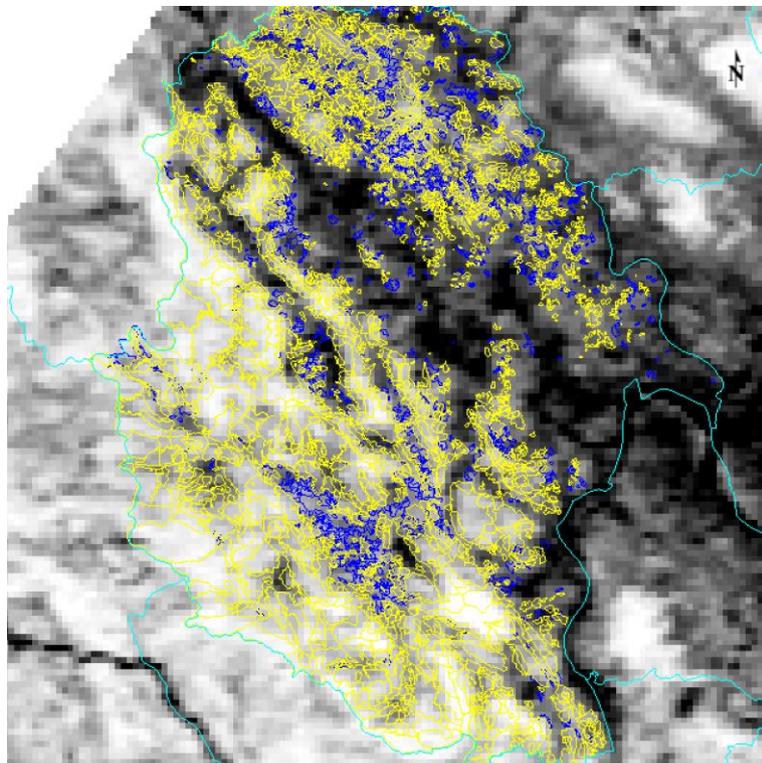
xử lý trên các kênh ảnh dùng để tính toán chỉ số NDVI (Hình 7).

Chỉ số thực vật NDVI trong khoảng thời gian từ 1/1/2013 đến 30/12/2015 được tính toán cho toàn bộ dữ liệu. Hình 8 thể hiện giá trị của tháng 12/2013 được tính toán, kết xuất từ chương trình

Showing only top 5 rows

B01	B02	PT_native	extent	crs
		MULTIPOLYGON (((10882123.751413476 23827...	[1.0837811065018026E7, 2268379.0601196797, 1.0956419120449172E7, 2386987.1155508263]	[+proj=sinu +lon_0=0.0 +x_0=0.0 +y_0=0.0 +a=6371007.181 +b=6371007.181 +units=m]
		MULTIPOLYGON (((10882123.751413476 23827...	[1.0837811065018026E7, 2268379.0601196797, 1.0956419120449172E7, 2386987.1155508263]	[+proj=sinu +lon_0=0.0 +x_0=0.0 +y_0=0.0 +a=6371007.181 +b=6371007.181 +units=m]
		MULTIPOLYGON (((10882123.751413476 23827...	[1.0837811065018026E7, 2268379.0601196797, 1.0956419120449172E7, 2386987.1155508263]	[+proj=sinu +lon_0=0.0 +x_0=0.0 +y_0=0.0 +a=6371007.181 +b=6371007.181 +units=m]
		MULTIPOLYGON (((10882123.751413476 23827...	[1.0837811065018026E7, 2268379.0601196797, 1.0956419120449172E7, 2386987.1155508263]	[+proj=sinu +lon_0=0.0 +x_0=0.0 +y_0=0.0 +a=6371007.181 +b=6371007.181 +units=m]
		MULTIPOLYGON (((10882123.751413476 23827...	[1.0837811065018026E7, 2268379.0601196797, 1.0956419120449172E7, 2386987.1155508263]	[+proj=sinu +lon_0=0.0 +x_0=0.0 +y_0=0.0 +a=6371007.181 +b=6371007.181 +units=m]

Hình 7. Tập hợp các dữ liệu raster sử dụng trong tính toán chỉ số NDVI (Kênh sóng Đỏ và cận hồng ngoại) của ảnh MODIS.



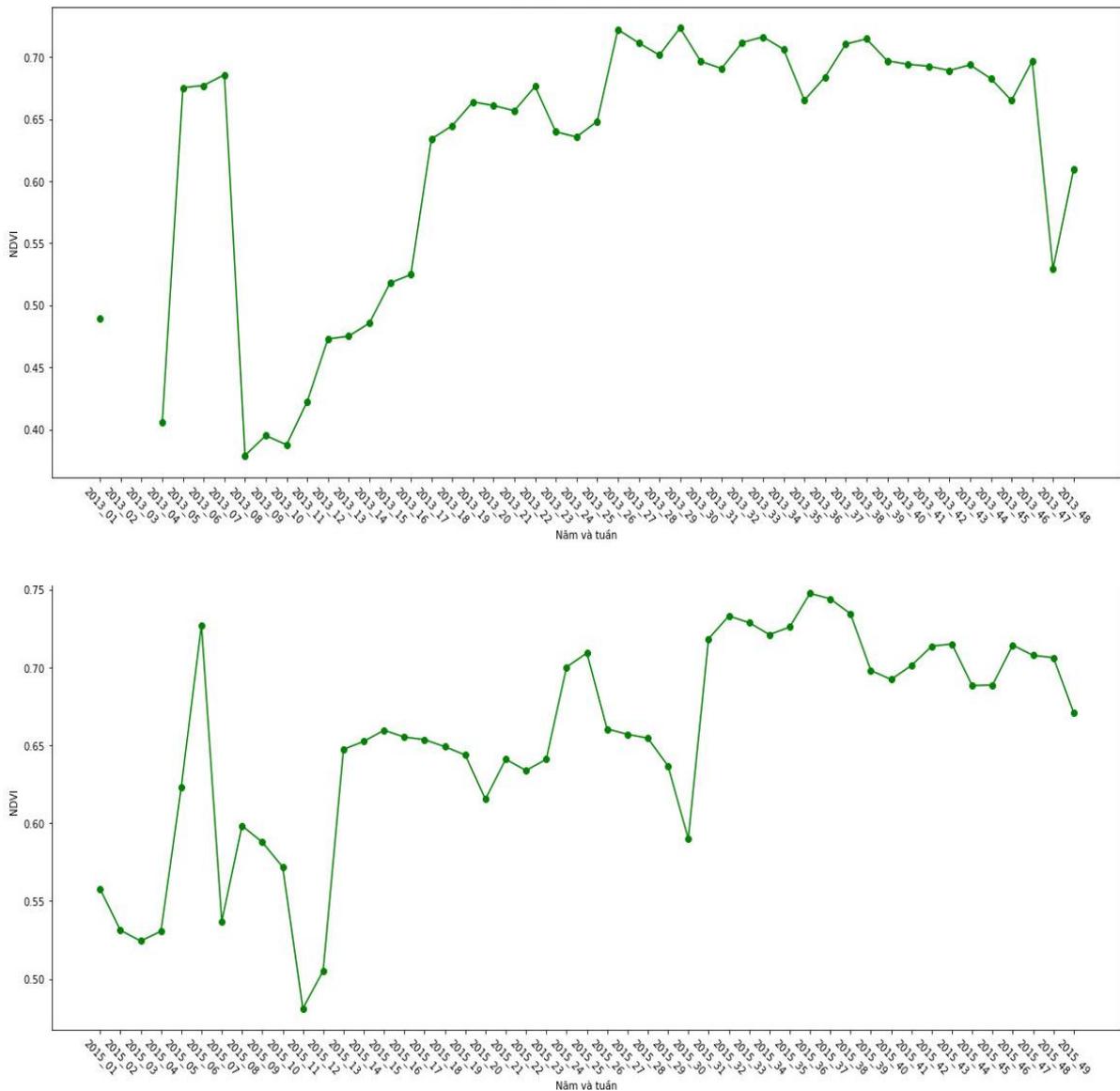
Hình 8. NIDV tại 12/2013 và bản đồ hiện trạng.

và bản đồ hiện trạng lớp phủ thực vật năm 2013 của tỉnh Phú Thọ. Kết quả cho thấy độ chính xác của phương pháp tính toán đề xuất. Khu vực trên ảnh có giá trị NDVI cao (>0.6) tương ứng với vùng lớp phủ rừng (màu vàng) trên bản đồ lớp phủ. Tiếp theo, phân tích theo chuỗi thời gian được thiết lập bằng cách sử dụng giá trị NDVI trung bình theo tuần trong khoảng thời gian này tại khu vực tỉnh Phú Thọ bằng việc sử dụng các hàm tính toán dựa trên nhóm và thời gian được tích hợp sẵn trên Pyspark và hàm tập hợp của RasterFrames để thực hiện việc tính toán. Hình 9 biểu diễn giá trị biến động NDVI theo tuần trong các năm 2013 và 2015. Kết quả cho thấy độ biến động ở 2 năm là

khá đồng đều. Giá trị NDVI trung bình có xu hướng giảm mạnh từ khoảng tháng 11÷3 năm sau chứng tỏ thực vật đang trút lá có thể do thời điểm này là mùa khô. Từ tháng 4÷10 giá trị NDVI trung bình tăng chứng tỏ thực vật đang dần phát triển ổn định do thời điểm này là mùa mưa không bị hạn hán, cháy rừng, nhiều ánh nắng mặt trời giúp thực vật quang hợp tốt.

5. Kết luận

Phương pháp xử lý và phân tích dữ liệu ảnh vệ tinh dựa trên nền tảng Apache Spark kết hợp với thư viện xử lý dữ liệu không gian RasterFrames. Dữ liệu đầu vào là tập các ảnh



Hình 9. Đồ thị biểu diễn xu hướng trong chuỗi thời gian NDVI (1/2013-12/2013 và 1/2015- 12/2015).

vệ tinh MODIS được thu nhận trong khoảng thời gian từ tháng 1/2013 đến tháng 12/2015 khu vực tỉnh Phú Thọ. Chỉ số NDVI theo chuỗi thời gian được tính toán và phân tích để đánh giá sự biến đổi của lớp phủ thực vật tại khu vực nghiên cứu.

Phương pháp đề xuất cho thấy khả năng mở rộng để có thể xử lý lượng dữ liệu lớn và kết nối từ nguồn dữ liệu phong phú. Các kết quả thực nghiệm được thực hiện trên một máy tính, tuy nhiên có thể mở rộng thành 1 cụm tính toán dựa trên nền tảng Apache Spark. Kết quả bài báo đã góp phần khẳng định vai trò của các nền tảng công nghệ dữ liệu lớn, điện toán đám mây, khoa học dữ liệu trong các bài toán liên quan đến quản lý, giám sát tài nguyên thiên nhiên và môi trường. Thực hiện các phân tích đa thời gian phức tạp hơn để có những đánh giá chi tiết và mở rộng nguồn dữ liệu ảnh đầu vào là một trong những hướng nghiên cứu tiếp theo.

Lời cảm ơn

Kết quả nghiên cứu này là một trong những nội nghiên cứu của đề tài “Xử lý ảnh vệ tinh trên nền tảng dữ liệu lớn”, Mã số T19-29.

Nhóm nghiên cứu xin chân thành cảm ơn Đề tài đã hỗ trợ cho nghiên cứu này.

Những đóng góp của tác giả

Nội dung khoa học của bài báo có sự đóng góp của tất cả các tác giả, cụ thể:

Nguyễn Thị Mai Dung: đề xuất phương pháp, chỉnh sửa bản thảo bài báo; Vũ Thị Hoài Thu: cài đặt chương trình, viết bản thảo bài báo.

Tài liệu tham khảo

Aji, A., Sun, X., Vo, H., Liu, Q., Lee, R., Zhang, X., Saltz, J. and Wang, F. (2013). Demonstration of Hadoop-GIS: a spatial data warehousing system over MapReduce. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 528-531. ACM.

Boyi Shangguan, Peng Yue, Zhaoyan Wu, and Liangcun Jiang. (2017). Big spatial data processing with Apache Spark. In *Agro-Geoinformatics*, 2017. IEEE.

Eldawy, A. and Mokbel, M. F., (2015). SpatialHadoop: A MapReduce framework for

spatial data. In *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*, 1352-1363. IEEE.

Databricks. Apache Spark – What is Spark. <http://databricks.com/spark>.

Fei Xiao. (2017). A Big Spatial Data Processing Framework Applying to National Geographic Conditions Monitoring. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Volume XLII-3, 2018 ISPRS TC III Mid-term Symposium “Developments, Technologies and Applications in Remote Sensing”, 7-10 May, Beijing, China.

Huang, Z., Chen, Y., Wan, L., and Peng, X. (2017). GeoSpark SQL: An Effective Framework Enabling Spatial Queries on Spark. In *ISPRS International Journal of Geo- Information*, 6(9), 285.

Hughes, J. N., Annex, A., Eichelberger, C. N., Fox, A., Hulbert, A. and Ronquest, M. (2015). Geomesa: a distributed architecture for spatio-temporal fusion. In *SPIE Defense+ Security, International Society for Optics and Photonics*, 94730F-94730F.

Kini, A. R. (2014). Emanuele. Geotrellis: Adding geospatial capabilities to spark. In *Spark Summit*.

Lu, J., and Guting, R. H. (2012). Parallel secondo: boosting database engines with hadoop. In *Parallel and Distributed Systems (ICPADS), (2012) IEEE 18th International Conference on*, 738-743. IEEE.

MODIS on AWS <https://docs.opendata.aws/modis-pds/readme.html>.

Nishimura, S., Das, S., Agrawal, D. and El Abbadi, A. (2011), June. Md-hbase: A scalable multi-dimensional data infrastructure for location aware services. In *Mobile Data Management (MDM), 2011 12th IEEE International Conference on*, 1, 7-16). IEEE.

Ram Sriharsha, <https://github.com/harsha2010/magellan>.

RasterFrames. <http://rasterframes.io/>.

Stefan Hagedorn, Philipp Gotze, Kai-Uwe Sattler.

- (2017). Big Spatial Data Processing Frameworks: Feature and Performance Evaluation. *In 20th International Conference on Extending Database Technology (EDBT)*.
- Thomas L., and, Ralph W. (2004). *Kiefer, Jonathan Chipman*. Remote sensing and image interpretation. Wiley.
- You, S., Zhang, J. and Gruenwald, L. (2015). Large-scale spatial join query processing in cloud. In *Data Engineering Workshops (ICDEW), 2015 31st IEEE International Conference on* (pp. 34-41). IEEE.
- Yu, J., Wu, J., and Sarwat, M. (2015). Geospark: A cluster computing framework for processing large-scale spatial data. *In Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems* (p.70). ACM.