**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA HỆ THỐNG THÔNG TIN**

**A blue logo with a black background

Description automatically generated**

**BÁO CÁO SEMINAR CHƯƠNG 5**

**DỮ LIỆU LỚN – IS405.P11.HTCL**

**APACHE BEAM**

**GVHD: ThS. Nguyễn Hồ Duy Tri**

Trương Vĩnh Thuận - 21522653

Hoàng Quốc Việt - 21522790

Phùng Thiên Phúc - 21521297

**HO CHI MINH CITY, 2024**

**Mục lục**

[1. Thông tin chung 4](#_Toc184461101)

[1.1 Apache Beam là gì? [1] 4](#_Toc184461102)

[1.2 Sự ra đời của Apache Beam 5](#_Toc184461103)

[1.3 Tại sao nên sử dụng Apache Beam 6](#_Toc184461104)

[1.4 Apache Beam Hoạt Động Như Thế Nào? 7](#_Toc184461105)

[2 Đặc trưng, ưu/nhược điểm 8](#_Toc184461106)

[2.1 Đặc trưng của Apache Beam 8](#_Toc184461107)

[2.1.1 Windowing [3] 8](#_Toc184461108)

[2.1.2 Các đặc trưng khác 10](#_Toc184461109)

[2.2 Ưu điểm, nhược điểm của Apache Beam 11](#_Toc184461110)

[2.2.1 Ưu điểm 11](#_Toc184461111)

[2.2.2 Nhược điểm 11](#_Toc184461112)

[3. Một vài trường hợp cụ thể đã áp dụng sản phẩm 12](#_Toc184461113)

[4. So sánh với Spark 14](#_Toc184461114)

[5. Cách cài đặt, kết nối với Spark [4] 15](#_Toc184461115)

[5.1 Ví dụ WordCount đơn giản với Apache Beam 15](#_Toc184461116)

[5.2 WordCount với dữ liệu streaming 19](#_Toc184461117)

[6. Tài liệu tham khảo 21](#_Toc184461118)

**Danh mục hình ảnh**

[Hình 1 Ngôn ngữ lập trình và Runner trong Apache Beam 4](#_Toc183463217)

[Hình 2 Sự ra đời của Apache Beam 5](#_Toc183463218)

[Hình 3 Tại sao nên sử dụng Apache Beam 6](#_Toc183463219)

[Hình 4 Các thành phần trong Apache Beam 7](#_Toc183463220)

[Hình 5 Cách hoạt động của Apache Beam 8](#_Toc183463221)

[Hình 6 Windowing trong Apache Beam 9](#_Toc183463222)

[Hình 7 Các đặc trưng khác của Apache Beam 10](#_Toc183463223)

[Hình 8 Ứng dụng của Apache Beam 12](#_Toc183463224)

[Hình 9 Logo Spotify 12](#_Toc183463225)

[Hình 10 Logo Airbus 13](#_Toc183463226)

# Thông tin chung

## Apache Beam là gì? [1]

Apache Beam (Batch + strEAM) là một framework mã nguồn mở được phát triển bởi Google, dùng để xây dựng các luồng dữ liệu (data pipelines) có thể chạy trên nhiều hệ thống xử lý dữ liệu phân tán khác nhau, như Apache Spark, Apache Flink, Google Cloud Dataflow, và nhiều hệ thống khác. Beam cung cấp một API thống nhất để lập trình các pipeline xử lý dữ liệu theo cả hai chế độ: batch processing (xử lý hàng loạt) và stream processing (xử lý luồng dữ liệu thời gian thực).

*\*Luồng dữ liệu (Data Pipeline) là một chuỗi các bước xử lý được thiết kế để tự động hóa quá trình thu thập, chuyển đổi, và phân phối dữ liệu từ nguồn đến đích.*

[2]A diagram of a diagram

Description automatically generated

Hình Ngôn ngữ lập trình và Runner trong Apache Beam

Beam được thiết kế để mang lại sự tự do cho người dùng trong việc xây dựng các pipeline xử lý dữ liệu batch hoặc streaming, sử dụng ngôn ngữ lập trình mà họ lựa chọn.

Các ngôn ngữ được hỗ trợ bao gồm Java, Python, Go, và SQL.

Ngoài ra, người dùng cũng có thể chọn runner (Cloud Dataflow, Apache Flink, Apache Spark,…), là nơi pipeline sẽ thực sự được thực thi.

## 1.2 Sự ra đời của Apache Beam

A diagram of a flowchart

Description automatically generated

Hình Sự ra đời của Apache Beam

Bài báo gốc về MapReduce được xuất bản vào năm 2004. Theo thời gian, MapReduce ngày càng được sử dụng rộng rãi trong hệ thống của Google. Tuy nhiên, khi xử lý các pipeline phức tạp với nhiều giai đoạn, người dùng bắt đầu đối mặt với một số hạn chế. FlumeJava, được công bố qua một bài báo vào năm 2011, đã ra đời như một giải pháp cho vấn đề này. Song song đó, nhu cầu về các pipeline có khả năng mở rộng, chính xác và độ trễ thấp đã dẫn đến sự phát triển của MillWheel. Sau đó, hai công nghệ này được kết hợp lại, tạo nên nền móng cho Apache Beam.

## Tại sao nên sử dụng Apache Beam

A diagram of a cloud computing system

Description automatically generated with medium confidence

Hình Tại sao nên sử dụng Apache Beam

* *Platform Independence*: Beam cho phép lập trình viên viết một pipeline một lần và nó có thể chạy trên nhiều hệ thống khác nhau (gọi là runners), như Spark, Flink, Google Cloud Dataflow,.. giúp giảm bớt phụ thuộc vào một nền tảng cụ thể.
* *Unified Model*: Beam cung cấp một mô hình thống nhất cho cả xử lý hàng loạt và xử lý luồng giúp ta không cần phải chọn công cụ khác nhau cho hai loại xử lý này.
* *Scalability*: Beam có thể xử lý dữ liệu ở quy mô lớn, từ GB đến PB, và cả trong thời gian thực.
* *Flexibility*: Beam cung cấp nhiều windowing và triggering để xử lý dữ liệu luồng và batch một cách linh hoạt.

## Apache Beam Hoạt Động Như Thế Nào?

A diagram of a computer component

Description automatically generated

Hình Các thành phần trong Apache Beam

Các pipeline trong Apache Beam được chia thành các thành phần chính:

* *PCollection*: Đại diện cho một tập hợp dữ liệu có thể là dữ liệu batch hoặc stream. Trong Beam, các đối tượng PCollection được sử dụng làm đầu vào và đầu ra, và các phép biến đổi (transforms) được áp dụng để xử lý chúng.
* *PTransform*: Áp dụng một phép biến đổi (transformation) lên một hoặc nhiều PCollection. Có các loại phép biến đổi phổ biến như ParDo, GroupByKey, Combine, và Flatten.
* *Runner*: Là các nền tảng xử lý dữ liệu mà Beam có thể chạy trên, như Google Cloud Dataflow, Apache Spark, Apache Flink, hoặc trực tiếp trên local.
* **Cách hoạt động của một pipeline Apache Beam cơ bản:**

A diagram of a process

Description automatically generated

Hình Cách hoạt động của Apache Beam

* *Input/Source*: Lấy dữ liệu từ nguồn (có thể từ Google Cloud Storage, Kafka, hoặc nguồn khác).
* *Transformations*: Áp dụng các bước xử lý dữ liệu như lọc, nhóm, tổng hợp, hoặc biến đổi.
* *Output/Sink*: Lưu dữ liệu đã xử lý vào đích (như BigQuery, ElasticSearch, HDFS, ...).

# Đặc trưng, ưu/nhược điểm

## 2.1 Đặc trưng của Apache Beam

### 2.1.1 Windowing [3]

Với tập dữ liệu vô hạn, việc thu thập tất cả các phần tử là không thể vì các phần tử mới liên tục được thêm vào và có thể vô cùng nhiều (như dữ liệu streaming). Khi làm việc với PCollections vô hạn, windowing đặc biệt hữu ích.

Windowing chia nhỏ một PCollection dựa trên dấu thời gian của các phần tử trong nó. Các phép biến đổi (transforms) như GroupByKey và Combine, vốn nhóm nhiều phần tử theo một khóa chung, thực hiện việc nhóm theo từng window — nghĩa là chúng xử lý mỗi PCollection như một chuỗi các window hữu hạn, mặc dù toàn bộ tập hợp có thể có kích thước vô hạn.

A white square with a black and white text

Description automatically generated

Hình Windowing trong Apache Beam

* **Các loại Windowing:**

1. *Fixed-Time Windows (Cửa sổ thời gian cố định):*

Đặc điểm: Chia dữ liệu thành các cửa sổ có độ dài cố định, không trùng lặp (ví dụ: mỗi giờ, mỗi ngày).

Ứng dụng: Dùng để thực hiện các phép toán tổng hợp theo thời gian, như đếm số lượng phần tử trong mỗi giờ.

1. *Sliding Windows (Cửa sổ trượt):*

Đặc điểm: Các cửa sổ có độ dài cố định, nhưng có thể "trượt" để bao gồm một phần dữ liệu mới tại mỗi lần cập nhật.

Ứng dụng: Dùng khi cần tính toán liên tục trên một cửa sổ di động, như tính toán trung bình của các giá trị trong khoảng thời gian liên tục.

1. *Session Windows (Cửa sổ phiên làm việc):*

Đặc điểm: Các cửa sổ được tạo ra dựa trên sự gián đoạn giữa các sự kiện, với khoảng thời gian không cố định.

Ứng dụng: Dùng cho các tập dữ liệu có sự kiện không đồng đều, chẳng hạn như theo dõi hành vi của người dùng trong một phiên làm việc trên website.

1. *Global Windows (Cửa sổ toàn cục):*

Đặc điểm: Tất cả dữ liệu được nhóm vào một cửa sổ duy nhất mà không chia nhỏ theo thời gian.

Ứng dụng: Dùng khi muốn xử lý toàn bộ dữ liệu một lần mà không phân chia theo thời gian cụ thể.

### 2.1.2 Các đặc trưng khác

A diagram of a computer program

Description automatically generated with medium confidence

Hình Các đặc trưng khác của Apache Beam

* *Portable Framework*: Apache Beam cho phép bạn viết một lần và chạy trên nhiều runner khác nhau mà không cần thay đổi mã nguồn.
* *Unified Batching and Streaming*: Beam hỗ trợ cả hai kiểu xử lý batch và stream với cùng một API.
* *Triggering*: Beam cung cấp các phương pháp triggering để kiểm soát khi nào đầu ra của dữ liệu luồng được xuất ra.
* *Stateful Processing*: Beam hỗ trợ xử lý có trạng thái (stateful processing), cho phép lưu trữ trạng thái trong suốt quá trình xử lý luồng.
* *Event-time Processing*: Beam hỗ trợ xử lý dựa trên event-time thay vì processing-time, giúp xử lý chính xác dữ liệu với độ trễ.

*\*Event-time đảm bảo tính chính xác cao, xử lý dữ liệu dựa trên thời điểm thực tế khi sự kiện xảy ra, ngay cả khi có độ trễ.*

*\*Processing-time đơn giản hơn và nhanh hơn nhưng không chính xác khi dữ liệu đến muộn hoặc bị chậm (thời gian dữ liệu được hệ thống xử lý).*

## 2.2 Ưu điểm, nhược điểm của Apache Beam

### 2.2.1 Ưu điểm

* *Hỗ trợ đa dạng platform*: Beam hỗ trợ nhiều platform khác nhau như Spark, Flink, Google Cloud Dataflow, giúp dễ dàng chuyển đổi giữa các nền tảng mà không cần thay đổi code.
* *Tính thống nhất*: Một API duy nhất cho cả batch và stream giúp đơn giản hóa việc lập trình khi bạn cần xử lý cả hai loại dữ liệu.
* *Element-wise Transformation*: Beam xử lý dữ liệu trên từng phần tử (element-wise), giúp dễ dàng thực hiện các phép biến đổi tùy chỉnh.
* *Tính mở rộng tốt*: Beam có thể xử lý dữ liệu ở quy mô lớn và phù hợp với những ứng dụng yêu cầu khả năng mở rộng cao.

### 2.2.2 Nhược điểm

* *Độ phức tạp khi sử dụng*: Do tính linh hoạt cao, Apache Beam có thể khá phức tạp cho những người mới làm quen.
* *Hiệu suất phụ thuộc vào runner*: Hiệu suất của pipeline phụ thuộc vào nền tảng (runner) mà bạn chọn. Các runner khác nhau sẽ có hiệu năng khác nhau.
* *Không mạnh mẽ như Spark hoặc Flink*: Mặc dù Beam hỗ trợ nhiều nền tảng, nhưng trong một số trường hợp nhất định, hiệu suất của nó có thể không tốt bằng khi bạn sử dụng trực tiếp Spark hoặc Flink.

# 3. Một vài trường hợp cụ thể đã áp dụng sản phẩm

A screenshot of a cell phone

Description automatically generated

Hình Ứng dụng của Apache Beam

Tính linh hoạt của Beam không chỉ dừng lại ở việc hỗ trợ các ngôn ngữ lập trình mà còn mở rộng đến nhiều trường hợp sử dụng khác nhau.

Với Apache Beam, ta có thể thiết lập các pipeline để xử lý mọi thứ, từ các tác vụ đơn giản như thu thập dữ liệu và thực hiện các phép biến đổi cơ bản, cho đến việc xây dựng các giải pháp phân tích thông minh liên tục.

* **Case Study 1: Google Cloud Dataflow**

A green text on a black background

Description automatically generated

Hình Logo Spotify

Google Cloud Dataflow là dịch vụ commercial dựa trên Apache Beam và được sử dụng rộng rãi trong các ứng dụng xử lý dữ liệu thời gian thực. Một ví dụ là Spotify, họ đã sử dụng Dataflow (với Beam) để xử lý dữ liệu phát nhạc của người dùng theo thời gian thực, cung cấp các thống kê cho người dùng ngay lập tức.

* **Case Study 2: Airbus**



Hình Logo Airbus

Airbus đã sử dụng Beam để xử lý và phân tích một lượng lớn dữ liệu cảm biến từ các hệ thống máy bay. Beam cho phép họ xử lý cả dữ liệu lịch sử (batch) và dữ liệu trực tiếp từ các cảm biến (stream) một cách thống nhất.

# 4. So sánh với Spark

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **Apache Beam** | **Apache Spark** |
| API thống nhất | Thống nhất cho cả batch và streaming | Batch và streaming có API riêng biệt |
| Platform | Hỗ trợ nhiều runner (Spark, Flink, Dataflow, etc.) | Chạy chủ yếu trên Spark |
| Cộng đồng và phát triển | Mới hơn, nhưng phát triển nhanh | Đã phát triển lâu hơn và có cộng đồng lớn |
| Windowing & Triggering | Hỗ trợ linh hoạt hơn cho xử lý stream | Spark Streaming 2.x hạn chế hơn Beam |
| Khả năng mở rộng | Phụ thuộc vào runner | Tối ưu hóa tốt trên Spark |
| Trạng thái (Stateful) | Hỗ trợ tốt stateful processing | Hỗ trợ stateful nhưng ít linh hoạt hơn Beam |
| Tính di động | Di động, chạy được trên nhiều nền tảng | Tập trung vào Spark, không di động như Beam |

# 5. Cách cài đặt, kết nối với Spark [4]

* **Cài đặt môi trường**

A white background with black text

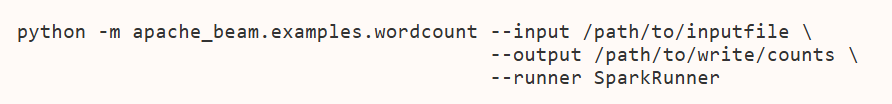
Description automatically generated

* **Cài đặt Apache Beam**

A black text on a white background

Description automatically generated

* **Cú pháp thực thi 1 pipeline**



## 5.1 Ví dụ WordCount đơn giản với Apache Beam

* **Mở Terminal trỏ đến thư mục chứa file pipeline**

A screenshot of a computer

Description automatically generated

* **File input.txt trong thư mục input để thực hiện đếm từ:**

A screenshot of a computer

Description automatically generated

* **Mở Spark để triển khai trên Spark Runner**

A screenshot of a computer screen

Description automatically generated

* **Pipeline**

A computer screen shot of a program code

Description automatically generated

* **Thực hiện pipeline trên Spark Runner**

A screen shot of a computer code

Description automatically generated

* **Kết quả: tạo ra 3 file output**

A screenshot of a computer

Description automatically generated

* **File output 1:**

A screenshot of a phone number

Description automatically generated

* **File output 2:**

A screenshot of a computer

Description automatically generated

* **File output 3:**

A screenshot of a phone

Description automatically generated

* **Thực hiện trên Direct Runner**



* **Kết quả: Output chỉ có 1 file duy nhất**

A screenshot of a computer

Description automatically generated

* **Kết luận:**
* Khi thực hiện trên SparkRunner, Spark phân chia công việc giữa các worker nodes để xử lý song song. Mỗi partition của dữ liệu được xử lý độc lập, và kết quả của từng partition được ghi ra một file riêng.
* Việc này giúp tránh việc tổng hợp kết quả (shuffling), vốn có thể gây tốn tài nguyên nếu dữ liệu lớn.

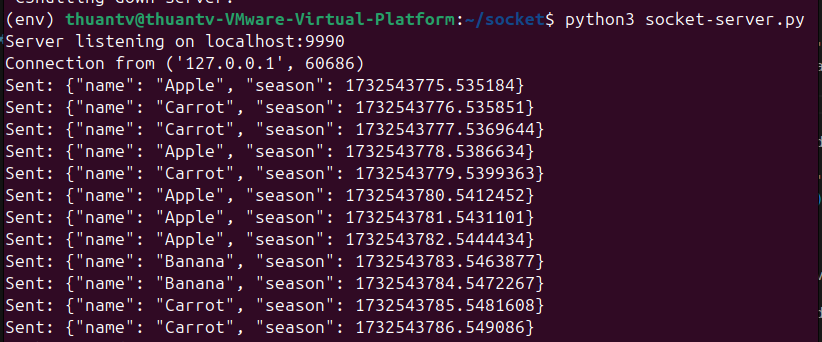
## 5.2 WordCount với dữ liệu streaming

* **Pipeline**

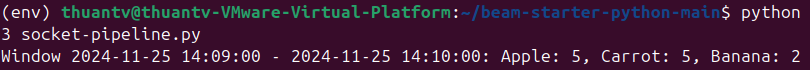
A screen shot of a computer code

Description automatically generated

* **Socket gửi dữ liệu streaming mỗi 1s gồm tên và timestamp của event**



* **Với Fixed Window 60s, kết quả:**



* **Với Fixed Window 5s, kết quả:**

A screenshot of a computer screen

Description automatically generated

# 6. Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | baeldung, "Baeldung," [Online]. Available: https://www.baeldung.com/apache-beam. [Accessed 17 11 2024]. |
| [2] | G. C. Tech. [Online]. Available: https://www.youtube.com/@googlecloudtech/featured. [Accessed 17 11 2024]. |
| [3] | "Tour of Beam," [Online]. Available: https://tour.beam.apache.org/tour/python/windowing/windowing-concept. [Accessed 17 11 2024]. |
| [4] | [Online]. Available: https://beam.apache.org/get-started/wordcount-example/#wordcount-example. [Accessed 25 11 2024]. |