Інтеграція PySpark Streaming з AWS EMR та Step Functions для обробки великих даних в реальному часі

https://doi.org/10.31713/MCIT.2025.039

Богдан Красько
Національний університет водного господарства та природокористування
м. Рівне, Україна
b.v.krasko@nuwm.edu.ua

Петро Грицюк
Національний університет водного господарства та природокористування
м. Рівне, Україна
p.m.hrytsiuk@nuwm.edu.ua

Анотація—**У статті розглядається інтеграція PySpark Streaming з AWS EMR та Step Functions для обробки великих даних в реальному часі. Описано рішення двох ключових проблем, що виникають при запуску та оновленні потокових обробок даних: забезпечення безперервної роботи стрім-джоб без downtime і автоматичне оновлення версій PySpark Streaming jobs. Для вирішення цих завдань пропонується використання AWS Step Functions для оркестрації запуску і зупинки джоб, а також S3 Bucket для визначення лідера, що гарантує запуск тільки однієї активної джоби. Окрім того, у статті розглянуто механізм автоматичного перезапуску джоб у разі помилок та створення подій для оновлення версій завдань. Такий підхід дозволяє забезпечити безперервну і масштабовану обробку потокових даних, мінімізуючи ризики downtime та забезпечуючи автоматизацію процесів на всіх етапах.**

Ключові слова; AWS, PySpark, BigData

I. ВСТУП

Зростання обсягів даних та швидкості їх генерації в сучасних інформаційних системах зумовлює потребу у переході від пакетної до потокової обробки. Традиційні підходи, що передбачають накопичення даних з подальшою періодичною обробкою, стають недостатніми в умовах, коли критично важливо приймати рішення на основі актуальної інформації. Приклади таких сценаріїв включають фінансові транзакції, моніторинг кібербезпеки, IoT-пристрої, аналіз поведінки користувачів у режимі реального часу та системи рекомендацій.

Apache Spark, а саме його модуль Structured Streaming, є однією з найпопулярніших платформ для обробки потокових даних завдяки високій масштабованості, здатності інтегруватися з різними джерелами даних та підтримці розподіленої обробки у кластерах. Використання Spark у хмарних середовищах, таких як Amazon Web Services (AWS), дозволяє досягати гнучкості в управлінні обчислювальними ресурсами, спрощує розгортання інфраструктури та знижує витрати на підтримку.

Зокрема, Amazon EMR on EKS надає можливість запускати Spark-джоби у середовищі Kubernetes, що забезпечує еластичне масштабування та уніфіковане управління контейнерними робочими навантаженнями. Проте постає низка викликів, пов'язаних з безперервною роботою PySpark Streaming job: обмеження тривалості виконання завдань у кластері, потреба в автоматичному перезапуску після збоїв, а також необхідність керованого оновлення версій джоб без зупинки потоків даних.

Для вирішення цих проблем у даній роботі запропоновано інтеграцію PySpark Streaming з сервісами AWS Step Functions та Amazon EventBridge, які виконують роль оркестраторів і механізмів реагування на події, а також використання Amazon S3 як сховища метаданих для синхронізації виконання. Такий підхід забезпечує:

- постійну доступність стрімінгових джоб;
- автоматичне самовідновлення у разі виникнення помилок;
- безперервність бізнес-процесів навіть під час оновлення версій Spark-джоб.

Таким чином, інтеграція PySpark Streaming з AWS-сервісами створює потужну інфраструктуру для обробки великих даних у реальному часі, яка поєднує у собі надійність, масштабованість та керованість.

II. ВИКЛАДЕННЯ ОСНОВНОГО МАТЕРІАЛУ

Організація потокової обробки великих даних у сучасних інформаційних системах пов'язана з низкою суттєвих викликів. З одного боку, бізнес прагне до зменшення затримки між появою події та її аналізом, щоб реагувати на зміни середовища у реальному часі. З іншого боку, інфраструктура, у якій розгортаються системи потокової обробки, накладає певні обмеження на тривалість виконання завдань, доступність ресурсів та узгодженість результатів. Особливу увагу слід звернути на архітектури, що базуються на Apache Spark, адже цей інструмент завдяки модулю Structured Streaming

є одним із провідних рішень для аналітики потокових даних [1].

Разом із тим, запуск Spark-джоб у середовищі Amazon EMR on EKS створює певні складності. Оскільки кожна джоба виконується у контейнеризованому середовищі Kubernetes, вона підпорядковується політикам управління життєвим циклом контейнерів і не може працювати безкінечно довго. Це суперечить самій природі потокових обчислень, які розраховані на безперервне виконання. У випадку, коли джоба завершується через перевищення часу життя, існує ризик втрати даних або виникнення дублювання під час спроби її перезапустити. Вирішенням цієї проблеми стає інтеграція декількох сервісів AWS, які в комплексі формують механізм безперервної роботи PySpark Streaming у режимі «always-on».

Ключову роль у цій архітектурі відіграє сервіс AWS Step Functions, який виконує функцію оркестратора. Завдяки йому кожна нова Spark-джоба може запускатися з наперед визначеною періодичністю, наприклад кожні тридцять хвилин. Таким чином обмеження на тривалість виконання окремої джоби нівелюється регулярними перезапусками. Проте цей підхід породжує іншу проблему: щоразу запускається новий процес, і якщо не реалізувати механізм контролю, у системі може одночасно працювати кілька стрімінгових джоб. Це спричинить дублювання обробки подій та помилки у бізнес-логіці.

Щоб уникнути цього, застосовується механізм координації на базі сховища Amazon S3. У спеціально відведеному бакеті створюється lock-файл, який виконує роль «маяка» для визначення активного процесу. Коли нова джоба стартує, вона намагається оновити lock-файл. Якщо це їй вдається, вона отримує статус лідера та продовжує виконання. Якщо ж lock-файл уже зайнятий, процес завершується, не завдаючи шкоди системі. Попередні джоби, які втрачають право лідера, перевіряють lock-файл і зупиняють свою роботу. У результаті, незалежно від кількості запущених процесів, обробку подій завжди здійснює лише один активний стрімінговий процес.

Надійність системи підвищується завдяки інтеграції з сервісом Amazon EventBridge. Потокові системи завжди залишаються вразливими до збоїв – це можуть бути помилки у коді, вичерпання ресурсів кластера або зовнішні чинники. У традиційних архітектурах такі збої призводили б до зупинки системи на невизначений час. Натомість EventBridge відстежує події завершення джоб і у разі виявлення помилки автоматично ініціює новий запуск Step Functions. Таким чином формується механізм самовідновлення, який дозволяє системі продовжувати роботу навіть у випадку критичних збоїв. Це особливо актуально для доменів, де будь-яка затримка в обробці подій може мати суттєві негативні наслідки, наприклад, у фінансових транзакціях чи системах виявлення кіберзагроз [2].

Окремим важливим аспектом є оновлення версій стрімінгових джоб. У потокових системах практично неможливо допустити «простоїв» під час переходу на нову версію, оскільки це призведе до втрати даних або затримки у реагуванні. У запропонованій архітектурі ця задача вирішується за допомогою подієво-орієнтованого підходу. При публікації нової версії джоби, наприклад, у вигляді JAR-файлу в Amazon S3 чи нового контейнера в Amazon ECR, генерується подія, яку обробляє EventBridge. Далі EventBridge ініціює запуск Step Functions, які створюють новий процес із оновленим кодом. Як тільки нова джоба успішно стартує, вона оновлює lock-файл у S3 та отримує право лідера. Стара версія, втративши це право, завершується. Завдяки цьому процес оновлення відбувається безшовно: користувачі не помічають жодних переривань, а система залишається узгодженою.

Таким чином, архітектура, що поєднує PySpark Streaming з Amazon EMR on EKS, Step Functions, EventBridge та S3, забезпечує високу стійкість до збоїв, безперервність виконання та керованість. Вона дозволяє ефективно працювати з великими обсягами даних у режимі реального часу, гарантуючи як точність результатів, так і оперативність реагування. Подібні архітектурні рішення уже знаходять широке застосування у промисловості, фінансах та сфері цифрових сервісів, а їх подальший розвиток відкриває перспективи для створення ще більш масштабованих та інтелектуальних систем аналітики потокових даних [3].

III. ВИСНОВКИ

Інтеграція PySpark Streaming з Amazon EMR on EKS, Step Functions, EventBridge та S3 забезпечує надійну та масштабовану архітектуру для обробки даних у реальному часі. Такий підхід гарантує безперервність роботи, автоматичне відновлення після збоїв і стабільність результатів, що робить його ефективним рішенням для роботи з великими потоками даних.

СПИСОК ЛІТЕРАТУРИ

[1] Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., & Stoica, I. (2016). Discretized streams: Fault-tolerant streaming computation at scale. Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '13), 423–438. https://doi.org/10.1145/2517349.252273

[2] Amazon Web Services. (2022). Build Modern Data Streaming Architectures on AWS. AWS Whitepaper. https://docs.aws.amazon.com/whitepapers/latest/build-modern-data-streaming-analytics-architectures/build-modern-data-streaming-analytics-architectures.html

[3] Karau, H., Konwinski, A., Wendell, P., & Zaharia, M. (2015). Learning Spark: Lightning-fast big data analysis. O'Reilly Media.