**Tarea 1**

**Comparativa de herramientas Big Data**

[**Servicios Big Data**](https://aula.ucam.edu/courses/305)

Germán Olate Quezada

***JOSÉ HUERTO MOREDA***

*MBA Data Analytics*

Índice de contenidos

[*1. Comparativa de herramientas 1*](#_Toc74047801)

[2. *Comparativa de herramientas del ecosistema Hadoop con una misma finalidad*  *2*](#_Toc74047802)

[*3. Puntos en comun y diferencias 3*](#_Toc74047803)

[*5. Comparativa de Herramientas adicional*  3](#_Toc74047805)

[*6. Referencias bibliográficas 5*](#_Toc74047812)

# Comparativa de herramientas

La comparativa que realizaré será la de las herramientas de **procesamiento de datos en tiempo real**

Se construyeron **Storm, Spark y Flink** para el procesamiento en tiempo real y en memoria (in-memory) de Big Data, construidos sobre el planificador de recursos de YARN y HDFS.

El procesamiento in-memory es un medio potente de ejecutar aplicaciones de Big Data incluso a más velocidad, logrando un rendimiento óptimo de 100x en algunas tareas.

**Storm:**Storm es el hadoop del mundo de Streaming. Es el marco de transmisión de código abierto más antiguo y uno de los más maduros y confiables. Es una transmisión real y es buena para casos de uso simples basados ​​en eventos

**Spark** :

Se ha convertido en el verdadero sucesor de hadoop en el procesamiento por lotes y el primer marco de trabajo que admite completamente la arquitectura Lambda (donde se implementan tanto Batch como Streaming; Batch para la corrección, Streaming para la velocidad).

**Flink** :

Flink también tiene antecedentes académicos similares a los de Spark. Mientras que Spark vino de UC Berkley, Flink vino de la Universidad TU de Berlín. Al igual que Spark, también es compatible con la arquitectura Lambda. Pero la implementación es bastante opuesta a la de Spark. Si bien Spark es esencialmente un lote con Spark streaming como micro-lote y un caso especial de Spark Batch, Flink es esencialmente un verdadero motor de transmisión que trata el lote como un caso especial de transmisión con datos limitados

# Comparativa de herramientas del ecosistema Hadoop con una misma finalidad.

**Spark:**

|  |  |
| --- | --- |
| *Ventajas* : | *Desventajas* |
|  |  |
| Compatible con la arquitectura Lambda, viene gratis con Spark | No es una transmisión real, no es adecuada para requisitos de baja latencia |
|  |  |
| Alto rendimiento, bueno para muchos casos de uso donde no se requiere sublatencia | [Demasiados parámetros para ajustar. Difícil hacerlo bien. Escribí una publicación sobre mi experiencia personal mientras sintonizaba Spark Streaming](http://why-not-learn-something.blogspot.in/2016/06/spark-streaming-performance-tuning-on.html) |
|  |  |
| Tolerancia a fallas por defecto debido a la naturaleza de micro lotes |  |
|  |  |
| API de nivel superior fáciles de usar | Va a la zaga de Flink en muchas funciones avanzadas |
|  |  |
| Gran comunidad y mejoras agresivas |  |
|  |  |
| Exactamente una vez |  |
|  |  |

**Flink** :

|  |  |
| --- | --- |
| *ventajas* : | *Desventajas* |
|  |  |
|  |  |
| Líder de innovación en el panorama de Streaming de código abierto | Un poco tarde en el juego, hubo una falta de adopción inicialmente |
|  |  |
| Primer marco de transmisión real con todas las funciones avanzadas como procesamiento de tiempo de eventos, marcas de agua, etc. | La comunidad no es tan grande como Spark, pero ahora crece a un ritmo acelerado |
|  |  |
| Baja latencia con alto rendimiento, configurable según los requisitos | No se conoce la adopción de Flink Batch a partir de ahora, solo es popular para la transmisión. |
|  |  |
| Ajuste automático, no demasiados parámetros para ajustar |  |
|  |  |
|  |  |
|  |  |
| Ser ampliamente aceptado por grandes empresas a gran escala como Uber, Alibaba. |  |
|  |  |

Storm:

|  |  |
| --- | --- |
| *ventajas*: | *Desventajas* |
|  |  |
|  |  |
| Latencia muy baja, transmisión real, madurez y alto rendimiento | Sin gestión estatal |
|  |  |
| Excelente para casos de uso de transmisión no complicados | Sin funciones avanzadas como procesamiento de tiempo de eventos, agregación, ventanas, sesiones, marcas de agua, etc. |
|  |  |
|  | Garantía de al menos una vez |

# Puntos en común y diferencias

Común

Principalmente su baja latencia

Diferencias

Principalmente en la era que se crean y en la evolución de cada una de ellas según las nuevas necesidades de procesamiento y velocidad

Sutilezas en el uso de memoria, capacidad de escalar, tolerancia a fallas

### Comparativa herramientas adicional

**Samza:**

Cubrirá Samza en breve. Samza se parece a Kafka Streams en el enfoque. Hay muchas similitudes. Ambos marcos han sido desarrollados por los mismos desarrolladores que implementaron Samza en LinkedIn y luego fundaron Confluent donde escribieron Kafka Streams. Ambas tecnologías están estrechamente relacionadas con Kafka, toman datos sin procesar de Kafka y luego devuelven los datos procesados ​. Usa la misma filosofía de Kafka Log. Samza es una especie de versión escalada de Kafka Streams. Si bien Kafka Streams es una biblioteca destinada a microservicios, Samza es un procesamiento de clúster completo que se ejecuta en Yarn

|  |  |
| --- | --- |
| Ventajas: | Desventajas: |
| Muy bueno para mantener grandes estados de información (bueno para el caso de uso de unir flujos) usando rocksDb y kafka log. | Estrechamente acoplado con Kafka y Yarn. No es fácil de usar si alguno de estos no está en su canal de procesamiento. |
|  |  |
| Tolerante a fallas y de alto rendimiento con las propiedades de Kafka | Garantía de procesamiento al menos una vez. No estoy seguro de si ahora es compatible exactamente una vez como Kafka Streams después de Kafka 0.11 |
|  |  |
| Una de las opciones a considerar si ya usa Yarn y Kafka en la canalización de procesamiento. | Falta de funciones avanzadas de transmisión como marcas de agua, sesiones, disparadores, etc. |
|  |  |
| Buen ciudadano de hilo |  |
|  |  |
| Baja latencia, alto rendimiento, maduro y probado a escala |  |

# Referencias bibliográficas

<https://medium.com/@chandanbaranwal/spark-streaming-vs-flink-vs-storm-vs-kafka-streams-vs-samza-choose-your-stream-processing-91ea3f04675b>

<http://spaceanalytics.blogspot.com/2016/08/introduccion-ecosistema-hadoop.html>