Open in app

Jhansi Karee

Following 209 Followers About

¿Cómo pueden los consumidores de Kafka paralelizar más allá del número de particiones?

Jhansi Karee 27 de julio de 2019 · 3 min de lectura

Como mencioné en mi artículo anterior , la forma de Kafka de lograr el paralelismo es tener múltiples consumidores dentro de un grupo. Esto escalaría a los consumidores, pero esta escala no puede ir más allá del número de particiones, ya que una partición puede asignarse como máximo a un consumidor en un grupo. Una forma fácil de resolver este problema es tener particiones altas para un tema. Esto podría tener sus propios impactos .

Enfrentamos un desafío similar en el que el paralelismo logrado mediante la ejecución de varios consumidores en un grupo no era suficiente. Buscábamos escalar más allá de esto.

¿Problema?

En O ur consumidor, estábamos recibiendo los registros en una corriente y hacer una llamada a la API. En su respuesta, reconocíamos el registro y pasamos al siguiente.

Although the end point we were hitting in our consumer can process very high number of parallel requests, we were making only the parallel requests as many as our consumers/partitions. We were limited to making only the number of parallel requests

as many as our consumers/partitions, because at any point of time we make only one request from each of our consumer.

We were not leveraging our end point which can process high number of parallel requests which is much beyond the number of parallel requests we were making.

We already had a decent number of partitions, we didn't want to increase beyond this. When we researched, if there is any way of increasing the parallelism without increasing the number of partitions. We came across some good suggestions which actually worked for us.

Solution:

Kafka consumers parallelising beyond the number of partitions, is this even possible? Yes, we may not be able to run more number of consumers beyond the number of partitions. However, parallelism could also be achieved by parallel processing multiple records within a consumer. Apart from increasing the consumers to parallel process, we also parallel processed the records within each consumer.

> There could be only one consumer thread in a consumer. However, we could spawn multiple application threads for processing those received records.

How did we implement this?

- If we have to parallel process multiple records, this means we need to **receive the records in batch instead of a single record**. Hence, we received the records in batch.
- Process the consumer record, in our case made the API call using application thread. We had **multiple application threads to parallel process the received records**.
- Make the consumer thread wait until all the API calls of the received batch are done.

- Once the entire batch is processed. Acknowledge the batch, release the consumer thread and get the next batch and process it in a similar way

One might take this approach not just to increase the parallelism beyond the number of partitions, but also to effectively utilise the consumer resources. By parallel processing within a consumer, we could achieve the same level of parallelism with lesser number of consumers.

Can we parallelise beyond the number of partitions. Not always, let's see when:

- Antes de procesar el primer mensaje si no podemos procesar el segundo mensaje. Con el paralelismo, no podemos garantizar el orden del procesamiento de los mensajes.
- Otro sistema del que depende el consumidor no podría manejar cargas elevadas.

Conclusión:

Podemos paralelizar a los consumidores más allá del número de particiones a través del procesamiento paralelo de registros dentro de un consumidor. Sin embargo, solo cuando el otro sistema también puede aceptar una carga tan alta y cuando no hay dependencia del orden de procesamiento.

Kafka Grupo de consumidores Kafka Consumidor de Kafka Procesamiento en paralelo

Acerca AyudaLegal
de

Get the Medium app

Download on the App Store GET IT ON Google Play