Ingest Component Analysis – Producer Unit Blocking Analysis

Maruzsi Csenge Virág

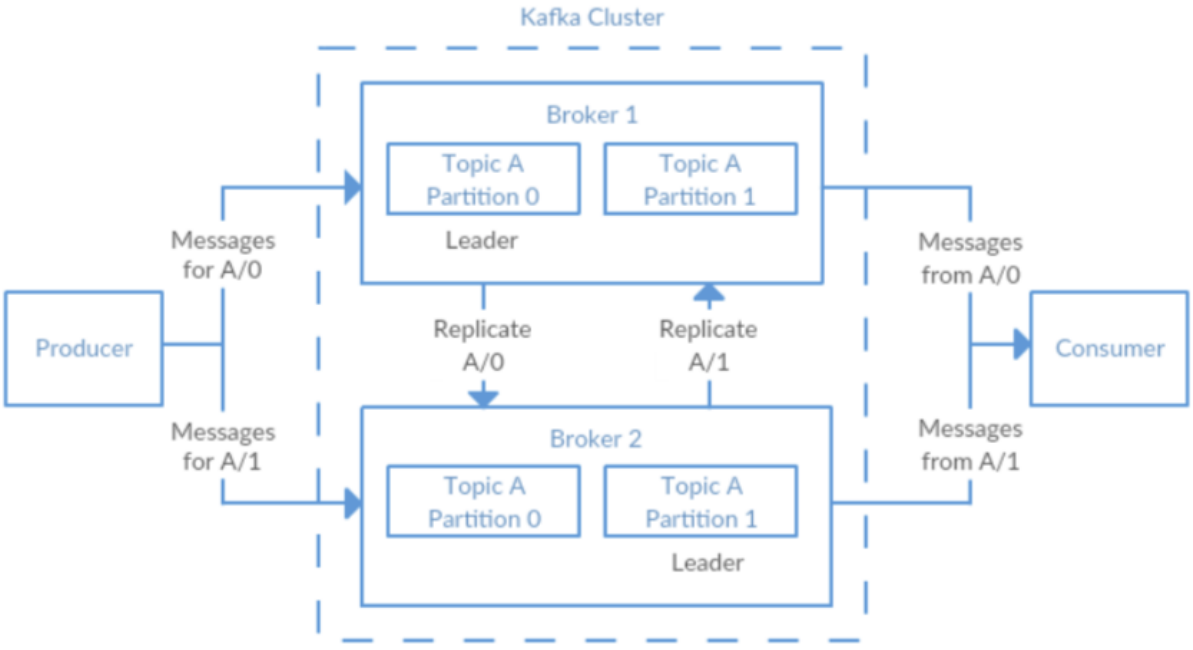
Beszámoló

Bevezetés

A világban egyre több és több adat keletkezik. Információt gyűjtünk, elemezzük, manipulálunk és hozunk létre még többet kimenetként. Minden alkalmazás adat létrehozásával jár, legyenek azok naplók, mérőszámok, felhasználói aktivitás, kimenő üzenetek vagy bármi más. Ennek a nagy mennyiségű adatnak a hatékony feldolgozására szolgálnak az úgynevezett *Big Data* rendszerek. Az adatokat el kell juttatnunk a létrehozási helyétől a – tipikusan felhőben lévő – feldolgozási helyéig. Minél gyorsabban csináljuk ezt, annál inkább agilisebb és reszponzívabb lehet a rendszer. Minél kevesebb erőfeszítést teszünk azért, hogy az adatok mozogjanak, annál többet tudunk koncentrálni a fontosabb munkákra. Ezért is egy kritikus összetevő a csővezeték egy adatközpontú vállalkozásban. Az Apache Kafka egy publikáló-feliratkozó rendszer. Gyakran hívják "megosztott változási napló"-nak vagy még gyakrabban "megosztott közvetítő felület"-nek. A fájlrendszer vagy adatbázis változási napló arra lett tervezve, hogy tartós bejegyzéseket szolgáltasson minden tranzakciónak, így elérhetőek, hogy konzisztensen építsék a rendszer állapotát. Hasonlóan, a Kafka-beli adatok tartósan vannak tárolva, rendszerezve, és determinisztikusan visszaolvashatóak. Továbbá az adatok meg lehetnek osztva a rendszeren belül, hogy az esetleges hibák ellen védelmet biztosítsanak, valamint lehetőséget nyújt a skálázhatóságra.

A Kafka felépítése

A Kafka kiszolgálók úgy lettek tervezve, hogy egy BigData adatfeldolgozó rendszer tagjaként működjenek. Egy számítógép egységnyi kiszolgálón belül az egyik vezérlőként fog funkcionálni. A vezérlő felelős az adminisztratív folyamatokért, beleértve a partíciók kiosztását a kiszolgálóknak és felügyelni a kiszolgáló hibákat. Ha egyetlen kiszolgáló a számítógép egységben birtokol egy partíciót, akkor ez a kiszolgáló a partíció vezetője. Egy partíció lehet több kiszolgálóhoz kiosztva, ami azt eredményezi, hogy a partíció többszörözve van, ahogyan az alábbi ábrán is látszik:



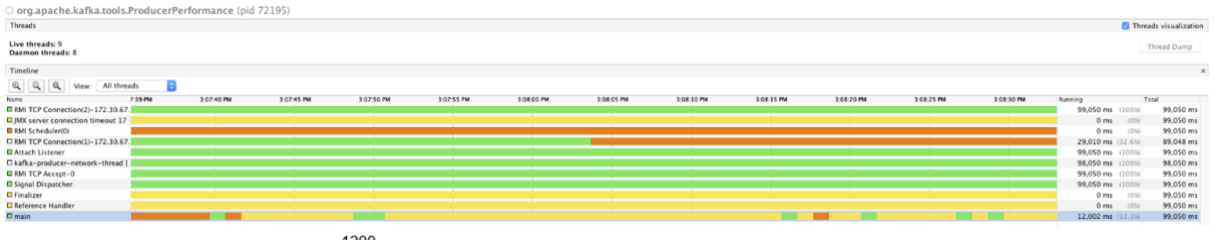
Ez biztosítja az üzenetek redundanciáját a partíciókon belül, ha esetleg egy másik kiszolgáló venné át a vezetést leállás esetén. Ugyanakkor mind a partíciókon tevékenykedő fogyasztóknak és termelőknek is csatlakozniuk kell egy vezetőhöz.
A fő jellemzője az Apache Kafka-nak a visszatartás vagy az üzenetek tartós tárolása egy bizonyos ideig. A Kafka kiszolgálók egy alap téma-visszatartási beállítással vannak konfigurálva: vagy visszatartja az üzeneteket egy bizonyos ideig (pl. 7 napig), vagy addig, amíg a téma elér egy bizonyos méretet byte-ban (pl. 1GB). Amint ezeket a határokat eléri, az üzenetek lejárnak és törlődnek, aminek a következtében mindig rendelkezésre áll egy minimális mennyiségű adat.

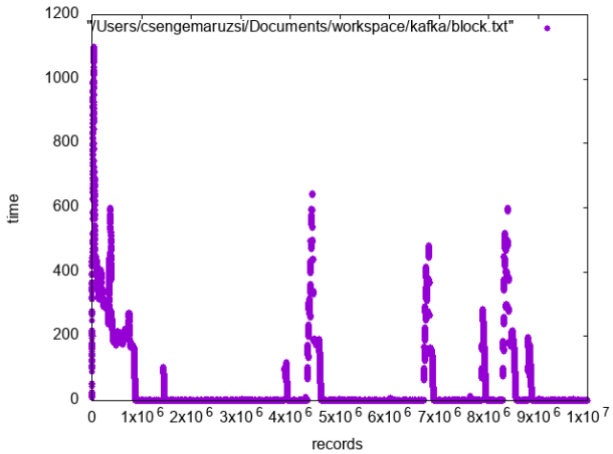
A késleltetési idő és a Producer-szál blokkolásának kapcsolata

A forgalom generálására a Kafka disztribúció részét képező kafka-producer-perf-test.sh programot használtuk. Ez a termelő teljesítményét vizsgálja úgy, hogy véletlen tartalmú üzenetekkel bombázza a rendszert, és vizsgálja, hogy milyen sebességgel, mennyi üzenetet tud feldolgozni. A mérési eredményeket 5 másodperces ablakonként mutatja. Több paraméter is megadható, ebből hármat állítottunk be: a rekordok számát (num-records), egy darab rekord méretét (record-size) és a másodpercenkénti elküldött rekordok számát (throughput). Ezek közül az első kettőt végig változatlanul hagytuk (10 000 000 rekordot küldtünk, egy mérete pedig 1024 byte volt), és a harmadik paraméter függvényében vizsgáltuk a rekordok késleltetését.
Azért a késleltetésre esett a választás, hogy ilyen módon vizsgálhassuk, hogy egy-egy üzenetet mikor tud feldolgozni a rendszer, mennyi üzenet az optimális másodpercenként, hogy megtudhassuk, kb milyen beállítások mellett lesz megfelelő a feldolgozási idő.

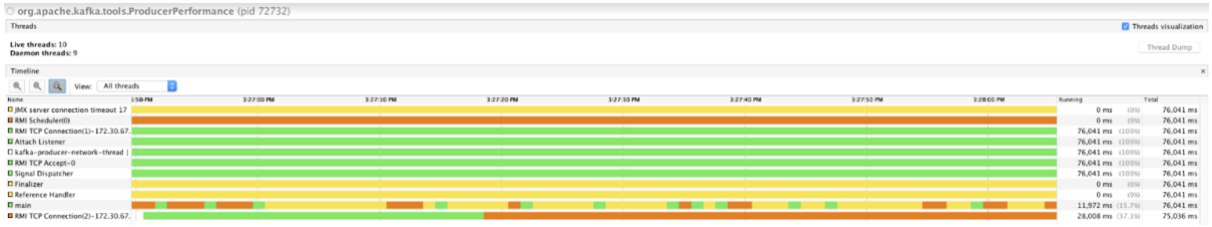
Mivel a késleltetések vizsgálata közben különös, vissza-visszatérő kiugrásokra lettünk figyelmesek, újabb méréseket végeztünk ezek okának kiderítésére.

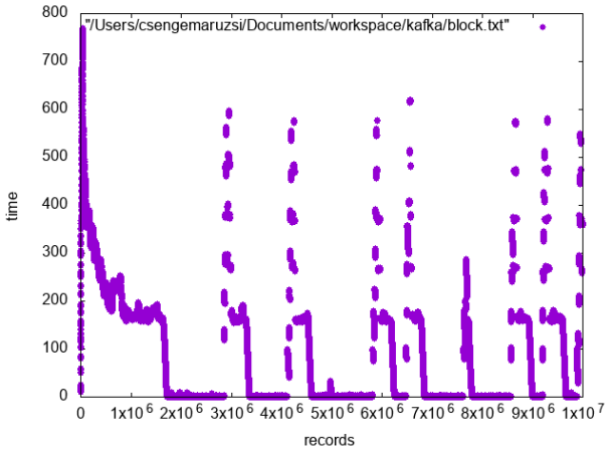
Ezúttal lokális méréseket végeztünk egy laptopon, és a program működését a tesztek közben VisualVM segítségével vizsgáltuk. A következő következtetést vontam le:



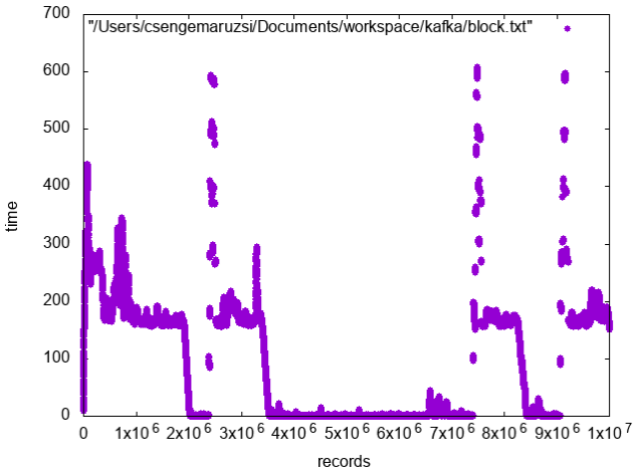


Throughput = 100000

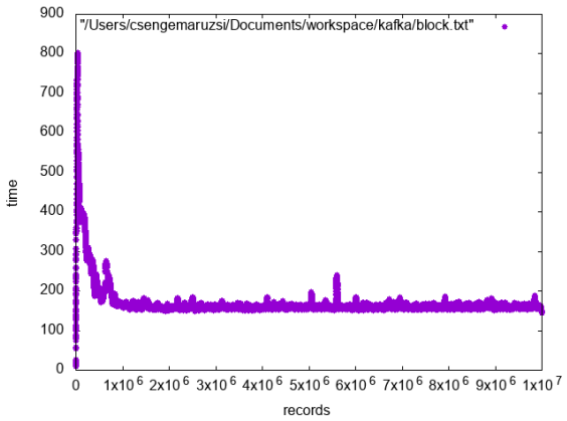




Throughput = 130000



Throughput = 150000



Throughput = 200000

A nagy késleltetés oka valószínűleg összefügg a main függvény futási szálának blokkolásával, ott időzik el sokat a folyamat. A producer blokkolása akkor áll elő, ha a rekordok továbbítása megakad, ekkor a buffer megtelik, és a producer nem veszi át az elküldésre szánt csomagot.

Összegzés

A vizsgálatok legfontosabb tanulsága, hogy a realisztikus mérések érdekében olyan terhelést kell alkalmaznunk, ami mellett nem fordul elő a fő végrehajtási szál blokkolása. Az általunk használt determinisztikus időközönként generált determinisztikus méretű csomagok erre a célra nem voltak alkalmasak, vagy észrevehetetlenül kicsi terhelést okoztak a brókernek, vagy azonnal túlterhelték azt. A jövőben sztochasztikus viselkedésű forgalommal folytatjuk a rendszer teljesítményvizsgálatát, melyben a megfelelően megválasztott méretű buffer a börsztök okozta túlterhelési időszakokat képes kezelni.