package com.twitter.recos.hose.common

import com.twitter.finagle.stats.{Stat, StatsReceiver}

import com.twitter.logging.Logger

import com.twitter.recos.internal.thriftscala.RecosHoseMessage

import java.util.concurrent.Semaphore

/\*\*

\* This class reads a buffer of edges from the concurrently linked queue

\* and inserts each edge into the recos graph.

\* If the queue is empty the thread will sleep for 100ms and attempt to read from the queue again.

\*/

case class BufferedEdgeWriter(

queue: java.util.Queue[Array[RecosHoseMessage]],

queuelimit: Semaphore,

edgeCollector: EdgeCollector,

statsReceiver: StatsReceiver,

isRunning: () => Boolean)

extends Runnable {

val logger = Logger()

private val queueRemoveCounter = statsReceiver.counter("queueRemove")

private val queueSleepCounter = statsReceiver.counter("queueSleep")

def running: Boolean = {

isRunning()

}

override def run(): Unit = {

while (running) {

val currentBatch = queue.poll

if (currentBatch != null) {

queueRemoveCounter.incr()

queuelimit.release()

var i = 0

Stat.time(statsReceiver.stat("batchAddEdge")) {

while (i < currentBatch.length) {

edgeCollector.addEdge(currentBatch(i))

i = i + 1

}

}

} else {

queueSleepCounter.incr()

Thread.sleep(100L)

}

}

logger.info(this.getClass.getSimpleName + " is done")

}

}