package com.twitter.servo.util

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

import scala.collection.mutable

/\*\*

\* Maintains a frequency counted circular buffer of objects.

\*/

class FrequencyCounter[Q](

size: Int,

threshold: Int,

trigger: Q => Unit,

statsReceiver: StatsReceiver = NullStatsReceiver) {

require(threshold > 1) // in order to minimize work for the common case

private[this] val buffer = new mutable.ArraySeq[Q](size)

private[this] var index = 0

private[this] val counts = mutable.Map[Q, Int]()

private[this] val keyCountStat = statsReceiver.scope("frequencyCounter").stat("keyCount")

/\*\*

\* Adds a new key to the circular buffer and updates frequency counts.

\* Runs trigger if this key occurs exactly `threshold` times in the buffer.

\* Returns true if this key occurs at least `threshold` times in the buffer.

\*/

def incr(key: Q): Boolean = {

// TOOD(aa): maybe write lock-free version

val count = synchronized {

counts(key) = counts.getOrElse(key, 0) + 1

Option(buffer(index)) foreach { oldKey =>

val countVal = counts(oldKey)

if (countVal == 1) {

counts -= oldKey

} else {

counts(oldKey) = countVal - 1

}

}

buffer(index) = key

index = (index + 1) % size

counts(key)

}

keyCountStat.add(count)

if (count == threshold) {

trigger(key)

}

count >= threshold

}

}