package com.twitter.search.core.earlybird.index.inverted;

import java.io.IOException;

import com.google.common.annotations.VisibleForTesting;

import org.apache.lucene.index.PostingsEnum;

import com.twitter.search.common.util.io.flushable.DataDeserializer;

import com.twitter.search.common.util.io.flushable.DataSerializer;

import com.twitter.search.common.util.io.flushable.FlushInfo;

import com.twitter.search.common.util.io.flushable.Flushable;

public class MultiPostingLists extends OptimizedPostingLists {

@VisibleForTesting

public static final int DEFAULT\_DF\_THRESHOLD = 1000;

private final OptimizedPostingLists lowDF;

private final OptimizedPostingLists highDF;

private final int dfThreshold;

/\*\*

\* Given the number of postings in each term (in this field), sum up the number of postings in

\* the low df fields.

\* @param numPostingsPerTerm number of postings in each term in this field.

\* @param dfThreshold the low/high df threshold.

\*/

private static int numPostingsInLowDfTerms(int[] numPostingsPerTerm, int dfThreshold) {

int sumOfAllPostings = 0;

for (int numPostingsInATerm : numPostingsPerTerm) {

if (numPostingsInATerm < dfThreshold) {

sumOfAllPostings += numPostingsInATerm;

}

}

return sumOfAllPostings;

}

/\*\*

\* Creates a new posting list delegating to either lowDF or highDF posting list.

\* @param omitPositions whether positions should be omitted or not.

\* @param numPostingsPerTerm number of postings in each term in this field.

\* @param maxPosition the largest position used in all the postings for this field.

\*/

public MultiPostingLists(

boolean omitPositions,

int[] numPostingsPerTerm,

int maxPosition) {

this(

new LowDFPackedIntsPostingLists(

omitPositions,

numPostingsInLowDfTerms(numPostingsPerTerm, DEFAULT\_DF\_THRESHOLD),

maxPosition),

new HighDFPackedIntsPostingLists(omitPositions),

DEFAULT\_DF\_THRESHOLD);

}

private MultiPostingLists(

OptimizedPostingLists lowDF,

OptimizedPostingLists highDF,

int dfThreshold) {

this.lowDF = lowDF;

this.highDF = highDF;

this.dfThreshold = dfThreshold;

}

@Override

public int copyPostingList(PostingsEnum postingsEnum, int numPostings)

throws IOException {

return numPostings < dfThreshold

? lowDF.copyPostingList(postingsEnum, numPostings)

: highDF.copyPostingList(postingsEnum, numPostings);

}

@Override

public EarlybirdPostingsEnum postings(int postingsPointer, int numPostings, int flags)

throws IOException {

return numPostings < dfThreshold

? lowDF.postings(postingsPointer, numPostings, flags)

: highDF.postings(postingsPointer, numPostings, flags);

}

@SuppressWarnings("unchecked")

@Override

public FlushHandler getFlushHandler() {

return new FlushHandler(this);

}

@VisibleForTesting

OptimizedPostingLists getLowDfPostingsList() {

return lowDF;

}

@VisibleForTesting

OptimizedPostingLists getHighDfPostingsList() {

return highDF;

}

public static class FlushHandler extends Flushable.Handler<MultiPostingLists> {

private static final String DF\_THRESHOLD\_PROP\_NAME = "dfThresHold";

public FlushHandler() {

super();

}

public FlushHandler(MultiPostingLists objectToFlush) {

super(objectToFlush);

}

@Override

protected void doFlush(FlushInfo flushInfo, DataSerializer out)

throws IOException {

MultiPostingLists objectToFlush = getObjectToFlush();

flushInfo.addIntProperty(DF\_THRESHOLD\_PROP\_NAME, objectToFlush.dfThreshold);

objectToFlush.lowDF.getFlushHandler().flush(

flushInfo.newSubProperties("lowDFPostinglists"), out);

objectToFlush.highDF.getFlushHandler().flush(

flushInfo.newSubProperties("highDFPostinglists"), out);

}

@Override

protected MultiPostingLists doLoad(FlushInfo flushInfo,

DataDeserializer in) throws IOException {

OptimizedPostingLists lowDF = new LowDFPackedIntsPostingLists.FlushHandler()

.load(flushInfo.getSubProperties("lowDFPostinglists"), in);

OptimizedPostingLists highDF = new HighDFPackedIntsPostingLists.FlushHandler()

.load(flushInfo.getSubProperties("highDFPostinglists"), in);

return new MultiPostingLists(

lowDF,

highDF,

flushInfo.getIntProperty(DF\_THRESHOLD\_PROP\_NAME));

}

}

}