package com.twitter.simclusters\_v2.scio

package multi\_type\_graph.multi\_type\_graph\_sims

import com.spotify.scio.ScioContext

import com.spotify.scio.coders.Coder

import com.spotify.scio.values.SCollection

import com.twitter.beam.io.dal.DAL

import com.twitter.beam.io.fs.multiformat.DiskFormat

import com.twitter.beam.io.fs.multiformat.PathLayout

import com.twitter.beam.job.DateRangeOptions

import com.twitter.dal.client.dataset.SnapshotDALDataset

import com.twitter.scio\_internal.coders.ThriftStructLazyBinaryScroogeCoder

import com.twitter.scio\_internal.job.ScioBeamJob

import com.twitter.scrooge.ThriftStruct

import com.twitter.simclusters\_v2.scio.multi\_type\_graph.common.MultiTypeGraphUtil

import com.twitter.simclusters\_v2.thriftscala.RightNode

import com.twitter.simclusters\_v2.thriftscala.RightNodeSimHashSketch

import com.twitter.util.Duration

import com.twitter.wtf.dataflow.cosine\_similarity.SimHashJob

import java.time.Instant

trait RightNodeSimHashScioBaseApp extends ScioBeamJob[DateRangeOptions] with SimHashJob[RightNode] {

override implicit def scroogeCoder[T <: ThriftStruct: Manifest]: Coder[T] =

ThriftStructLazyBinaryScroogeCoder.scroogeCoder

override val ordering: Ordering[RightNode] = MultiTypeGraphUtil.rightNodeOrdering

val isAdhoc: Boolean

val rightNodeSimHashSnapshotDataset: SnapshotDALDataset[RightNodeSimHashSketch]

val simsHashJobOutputDirectory: String = Config.simsHashJobOutputDirectory

override def graph(

implicit sc: ScioContext,

): SCollection[(Long, RightNode, Double)] =

MultiTypeGraphUtil.getTruncatedMultiTypeGraph(noOlderThan = Duration.fromDays(14))

override def configurePipeline(sc: ScioContext, opts: DateRangeOptions): Unit = {

implicit def scioContext: ScioContext = sc

// DAL.Environment variable for WriteExecs

val dalEnv = if (isAdhoc) DAL.Environment.Dev else DAL.Environment.Prod

val sketches = computeSimHashSketchesForWeightedGraph(graph)

.map {

case (rightNode, sketch, norm) => RightNodeSimHashSketch(rightNode, sketch, norm)

}

// Write SimHashSketches to DAL

sketches

.saveAsCustomOutput(

name = "WriteSimHashSketches",

DAL.writeSnapshot(

rightNodeSimHashSnapshotDataset,

PathLayout.FixedPath(

((if (!isAdhoc)

MultiTypeGraphUtil.RootThriftPath

else

MultiTypeGraphUtil.AdhocRootPath)

+ simsHashJobOutputDirectory)),

Instant.ofEpochMilli(opts.interval.getEndMillis - 1L),

DiskFormat.Thrift(),

environmentOverride = dalEnv

)

)

}

}