package com.twitter.recos.user\_user\_graph

import com.twitter.recos.model.Constants

import com.twitter.recos.graph\_common.NodeMetadataLeftIndexedPowerLawMultiSegmentBipartiteGraphBuilder.GraphBuilderConfig

/\*\*

\* The class holds all the config parameters for recos graph.

\*/

object RecosConfig {

val maxNumSegments: Int = 5

val maxNumEdgesPerSegment: Int = 1 << 26 // 64M edges per segment

val expectedNumLeftNodes: Int = 1 << 24 // should correspond to 16M nodes storage

val expectedMaxLeftDegree: Int = 64

val leftPowerLawExponent: Double = 16.0 // steep power law as most nodes will have a small degree

val expectedNumRightNodes: Int = 1 << 24 // 16M nodes

val numRightNodeMetadataTypes = 1 // UUG does not have node metadata

val graphBuilderConfig = GraphBuilderConfig(

maxNumSegments = maxNumSegments,

maxNumEdgesPerSegment = maxNumEdgesPerSegment,

expectedNumLeftNodes = expectedNumLeftNodes,

expectedMaxLeftDegree = expectedMaxLeftDegree,

leftPowerLawExponent = leftPowerLawExponent,

expectedNumRightNodes = expectedNumRightNodes,

numRightNodeMetadataTypes = numRightNodeMetadataTypes,

edgeTypeMask = new UserEdgeTypeMask()

)

println("RecosConfig - maxNumSegments " + maxNumSegments)

println("RecosConfig - maxNumEdgesPerSegment " + maxNumEdgesPerSegment)

println("RecosConfig - expectedNumLeftNodes " + expectedNumLeftNodes)

println("RecosConfig - expectedMaxLeftDegree " + expectedMaxLeftDegree)

println("RecosConfig - leftPowerLawExponent " + leftPowerLawExponent)

println("RecosConfig - expectedNumRightNodes " + expectedNumRightNodes)

println("RecosConfig - numRightNodeMetadataTypes " + numRightNodeMetadataTypes)

println("RecosConfig - salsaRunnerConfig " + Constants.salsaRunnerConfig)

}