Generation of multi-level directed graphs

Boris Petelin

```
library(knitr)
```

```
knitr::opts_chunk$set(cache=FALSE)
```

```
import sys
import time
mldgpath = "C:/Users/petelin/Documents/MLDG"
sys.path.insert(0, mldgpath)
import mldg

model = 'mfs'
modelpath = mldgpath + "/models/" + model
meshfile = modelpath + "/" + "meshmask.nc"

projspath = mldgpath + "/" + "projects"
nx = 60
ny = 30
project = model + "_" + "%2dx%2d" % (nx,ny)
projpath = projspath + '/' + project

geompath = projpath + '/geometry'
vertfile = geompath + '/' + model + '_vertices_' + "%dx%d" % (nx,ny) + '.shp'

start_time = time.time()
(verts,name,long_name,lon_wmc,lat_wmc) = mldg.load_vertices(vertfile)
elapsed_time = time.time() - start_time
m, s = divmod(elapsed_time, 60)
h, m = divmod(m, 60)
print 'Load vertices elapsed time: ' + '%d hours %02d minutes %02d seconds' % (h, m, s)

trajpath = projpath + '/trajectories'
trajfile = trajpath + '/mfs_ariane_1999_2011.nc'
baseyear = 1999
basemonth = 1
baseday = 2
yearmin = 1999
yearmax = 2011
monthmin = 1
monthmax = 12
depthmin = 0
depthmax = -20.
timeinter = 6
sampinter = 3

exampath = projpath + '/examples'
examfile = exampath + '/' + model + '_examples_' + ('%dx%d' % (nx,ny)) + '_' + \
           ('%04d' % timeinter) + 'days_years_' + ('%d_%d' % (yearmin, yearmax)) + \
           '_' + 'depth' + '_' + ('%04d_%04d' % (depthmin*(-1.),depthmax*(-1.))) + '.txt'
```

```
start_time = time.time()
# (nout,nerr) = mldg.generate_examples(verts,name,trajfile,baseyear,basemonth,baseday, \
#                                       yearmin,monthmin,yearmax,monthmax,depthmin,depthmax, \
#                                       examfile,timeinter,sampinter)
nout = 0
nerr = 0
print 'Number of examples generated: %d \tNumber of errors: %d' % (nout, nerr)
(h, m, s) = mldg.calc_elapsed_time(start_time)
print 'Generate examples elapsed time: ' + '%d hours %02d minutes %02d seconds' % (h, m, s)

minSupport = 0.000001
minConf = 0.0001
maxIS = 1000000

rulepath = projpath + '/rules'
rulefile = rulepath + '/' + model + '_rules_' + ('%dx%d' % (nx,ny)) + '_' + \
           '%04d' % timeinter + 'days_years_' + ('%d_%d' % (yearmin, yearmax)) + \
           '_' + 'depth' + '_' + ('%04d_%04d' % (depthmin*(-1.),depthmax*(-1.))) + '.txt'
nrules = 0
nmatch = 0
start_time = time.time()
#(nrules,nmatch) = mldg.generate_assoc_rules(vertfile,examfile,minSupport,minConf,maxIS,rulefile)
print 'Number of assoc. rules generated: %d \tNumber of matched rules: %d' % (nrules, nmatch)
(h, m, s) = mldg.calc_elapsed_time(start_time)
print 'Generate assoc. rules elapsed time: ' + '%d hours %02d minutes %02d seconds' % (h, m, s)

graphpath = projpath + '/graphs'
statfile = graphpath + '/' + model + '_graph_statistics_' + ('%dx%d' % (nx,ny)) + '_' + \
           '%04d' % timeinter + 'days_years_' + ('%d_%d' % (yearmin, yearmax)) + \
           '_' + 'depth' + '_' + ('%04d_%04d' % (depthmin*(-1.),depthmax*(-1.))) + '.txt'
start_time = time.time()
ngraph = mldg.generate_graphs(model, vertfile, yearmin, yearmax, monthmin, monthmax, \
                              timeinter, depthmin, depthmax, graphpath, rulefile, statfile)
(h, m, s) = mldg.calc_elapsed_time(start_time)
print 'Generate graphs elapsed time: ' + '%d hours %02d minutes %02d seconds' % (h, m, s)
```

```
## Load vertices elapsed time: 0 hours 00 minutes 00 seconds
## Number of examples generated: 0  Number of errors: 0
## Generate examples elapsed time: 0 hours 00 minutes 00 seconds
## Number of assoc. rules generated: 0  Number of matched rules: 0
## Generate assoc. rules elapsed time: 0 hours 00 minutes 00 seconds
## Generate graphs elapsed time: 0 hours 07 minutes 00 seconds
```