How To use the clusterGraph and distGraph classes

May 2, 2019

Introduction

Graphs can be used to help explore different clustering algorithms. While they have not been extensively used in this regard that is probably due to the lack of software rather than for any other reasons. As we demonstrate below, one can easily and naturally compare different aspects of clustering algorithms using these tools.

clusterGraph

A *clusterGraph* is a graph defined on a set of nodes that have been clustered or grouped in some fashion. The grouping must form a partition of the nodes. In this graph all nodes within the same cluster are adjacent while there are no edges between clusters.

Thus, each cluster is a complete graph but there are no between cluster edges.

```
> library("graph")
> library("cluster")
> data(ruspini)
> pm <- pam(ruspini, 4)
> cG <- new("clusterGraph",
+      clusters = split(names(pm$clustering), pm$clustering))
> nodes(cG)
```

```
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14" "15"
[16] "16" "17" "18" "19" "20" "21" "22" "23" "24" "25" "26" "27" "28" "29" "30"
[31] "31" "32" "33" "34" "35" "36" "37" "38" "39" "40" "41" "42" "43" "44" "45"
[46] "46" "47" "48" "49" "50" "51" "52" "53" "54" "55" "56" "57" "58" "59" "60"
[61] "61" "62" "63" "64" "65" "66" "67" "68" "69" "70" "71" "72" "73" "74" "75"
```

We now have a graph that we could perform various operations on. For example, we could try a second clustering algorithm on the same data and see if the two largely agree.

```
> library(stats)
> km = kmeans(ruspini, 4)
> cG.km = new("clusterGraph",
+       clusters=split(as.character(1:75), km$cluster))
> inBoth = intersection(cG.km, cG)
```

The graph `inBoth` is of length 1 indicating that there are that many distinct groups. One could, compute various measures of correspondence between the two clustering algorithms using the graph representation.

distGraph

We use this same data to consider some potential uses for the *distGraph* class. Others have considered a similar structure for exploring clustering algorithms.

```
> d1 = dist(ruspini)
> dG = new("distGraph", Dist=d1)
> rl = NULL
> j=1
> for(i in c(40, 30, 10, 5) ){
+   nG = threshold(dG, i)
+   rl[[j]] = connComp(nG)
+   j=j+1
+ }
```

We can then examine the components of `rl` to see how the graph is being reduced.

```
> sapply(rl, length)

[1]  3  3 11 36

>
```

We see that when we remove all distances that are bigger than 5 units (the range of distances was from 1.414 to 154.496) there are still only 36 connected components - one of which is of size 11.