

Graphs in the **gRbase** package

Søren Højsgaard

January 18, 2011

Contents

1	Introduction	1
2	Graphs	2
3	Plotting graphs	3
4	Graph queries	3
5	More advanced graph operations	4
6	Coercion	5
7	Time and space considerations	6
7.1	Space	6
7.2	Space	7

1 Introduction

For the R community, the packages **igraph**, **graph**, **RBGL** and **Rgraphviz** are extremely useful tools for graph operations, manipulation and layout. The **gRbase** package adds some additional tools to these fine packages. The most important tools are:

1. Undirected and directed acyclic graphs can be specified using formulae or an adjacency list using the functions **ug()** and **dag()**. This gives graphs represented as **graphNEL** objects, which is one of the graph representations available in the **graph** package. (The 'NEL' in **graphNEL** stands for “node–edge–list”).
2. Similarly, graphs represented as adjacency matrices can be specified as formula or an adjacency list using **ugMAT()** and **dagMAT()**.

3. Some graph algorithms are implemented in **gRbase**. These can be applied to graphs represented as **graphNELs** or as matrices.

The most important algorithms are: **mcs()**, **mcsMAT()** (maximum cardinality search) **moralize()**, **moralizeMAT()** (moralization of directed acyclic graph), **rip()**, **ripMAT()** (RIP ordering of cliques of triangulated undirected graph), **triangulate()**, **triangulateMAT()** (triangulate undirected graph). For example **mcs()** can be applied to a **graphNEL** object whereas **mcsMAT** is to be applied to an adjacency matrix.

Furthermore corresponding to some of the functions in the **graph** and **RBGL** packages there are corresponding matrix versions of these implemented in **gRbase**. These are: **maxCliqueMAT()**.

2 Graphs

An undirected graph represented as a **graphNEL** object is created by the **ug()** function. The graph can be specified by a formula, a list of formulas or a list of vectors. Thus the following two forms are equivalent:

```
> ug11 <- ug(~a*b*c, ~c*d, ~d*e, ~e*a, ~f*g)
> ug12 <- ug(~a*b*c + c*d + d*e + a*e + f*g)
> ug13 <- ug(c("a", "b", "c"), c("c", "d"), c("d", "e"), c("a", "e"), c("f", "g"))
> ug13
```

A **graphNEL** graph with undirected edges

Number of Nodes = 7

Number of Edges = 7

Notice that a “.” can be used instead of “*” in the formula specifications above.

A representation as an adjacency matrix can be obtained with one of the following equivalent specifications:

```
> ug11m <- ugMAT(~a*b*c, ~c*d, ~d*e, ~e*a, ~f*g)
> ug12m <- ugMAT(~a*b*c + c*d + d*e + a*e + f*g)
> ug13m <- ugMAT(c("a", "b", "c"), c("c", "d"), c("d", "e"), c("a", "e"), c("f", "g"))
> ug13m
```

	a	b	c	d	e	f	g
a	0	1	1	0	1	0	0
b	1	0	1	0	0	0	0
c	1	1	0	1	0	0	0
d	0	0	1	0	1	0	0
e	1	0	0	1	0	0	0
f	0	0	0	0	0	0	1
g	0	0	0	0	0	1	0

A directed acyclic graph can be specified as a collection of formulas or as a list of vectors

```
> dag11 <- dag(~a, ~b*a, ~c*a*b, ~d*c*e, ~e*a, ~g*f)
> dag12 <- dag("a", c("b","a"), c("c","a","b"), c("d","c","e"),
+             c("e","a"),c("g","f"))
> dag12
```

A graphNEL graph with directed edges

Number of Nodes = 7

Number of Edges = 7

Here `~a` means that “a” has no parents while `~d*b*c` means that “d” has parents “b” and “c”. Notice that a “:” can be used instead of “*” the specification.

A representation as an adjacency matrix can be obtained with

```
> dag11m <- dagMAT(~a, ~b*a, ~c*a*b, ~d*c*e, ~e*a, ~g*f)
> dag12m <- dagMAT("a", c("b","a"), c("c","a","b"), c("d","c","e"),
+             c("e","a"),c("g","f"))
> dag12m

  a b c d e g f
a 0 1 1 0 1 0 0
b 0 0 1 0 0 0 0
c 0 0 0 1 0 0 0
d 0 0 0 0 0 0 0
e 0 0 0 1 0 0 0
g 0 0 0 0 0 0 0
f 0 0 0 0 0 1 0
```

3 Plotting graphs

Graphs (represented as `graphNEL` objects) are displayed with `plot()`, but this requires that the `Rgraphviz` package is installed. There is also an `iplot` function for graphs and this function does not require any additional software to be installed.

4 Graph queries

The `graph` and `RBGL` packages implement various graph operations for `graphNEL` objects. See the documentation for these packages. The `gRbase` implements a few additional functions, see Section 1. An additional function in `gRbase` for graph operations is `query-`

`graph()`. This function is intended as a wrapper for the various graph operations available in `gRbase`, `graph` and `RBGL`.

5 More advanced graph operations

A moralized directed acyclic graph is obtained with

```
> moralize(dag11)

A graphNEL graph with undirected edges
Number of Nodes = 7
Number of Edges = 8
```

Testing for whether a graph is triangulated is based on Maximum Cardinality Search. If `character(0)` is returned the graph is not triangulated. Otherwise a linear ordering of the nodes is returned.

```
> mcs(ug11)

character(0)
```

Triangulate an undirected graph by adding extra edges to the graph:

```
> tug1<-triangulate(ug11)

A graphNEL graph with undirected edges
Number of Nodes = 7
Number of Edges = 8
```

A RIP ordering of the cliques of a triangulated graph can be obtained as:

```

> r <- rip(tug1)
> r

cliques
  1 : c a b
  2 : e a c
  3 : d c e
  4 : g f
separators
  1 :
  2 : a c
  3 : c e
  4 :
parents
  1 : 0
  2 : 1
  3 : 2
  4 : 0

```

For graphs represented as matrices, the corresponding functions are `moralizeMAT()`, `mc-sMAT()`, `triangulateMAT()` and `ripMAT()`.

6 Coercion

Coercion between representations as a `graphNEL` object and an adjacency matrix can be done with the `as()` method from the `graph` package:

```

> as(ug11, "matrix")

  a b c d e f g
a 0 1 1 0 1 0 0
b 1 0 1 0 0 0 0
c 1 1 0 1 0 0 0
d 0 0 1 0 1 0 0
e 1 0 0 1 0 0 0
f 0 0 0 0 0 0 1
g 0 0 0 0 0 1 0

> as(ug11m, "graphNEL")

A graphNEL graph with undirected edges
Number of Nodes = 7
Number of Edges = 7

```

```

> as(dag11,"matrix")

  a b c d e g f
a 0 1 1 0 1 0 0
b 0 0 1 0 0 0 0
c 0 0 0 1 0 0 0
d 0 0 0 0 0 0 0
e 0 0 0 1 0 0 0
g 0 0 0 0 0 0 0
f 0 0 0 0 0 1 0

> as(dag11m, "graphNEL")

A graphNEL graph with directed edges
Number of Nodes = 7
Number of Edges = 7

```

7 Time and space considerations

7.1 Space

It is worth noticing that working with graphs represented as **graphNEL** objects is somewhat slower working with graphs represented as adjacency matrices. On the other hand, graph

Consider for example coercion from a **graphNEL** object. This can be obtained with `as()` as shown above or by using `as.adjMAT()` from **gRbase**. The timings are:

```

> system.time({for (ii in 1:200) as(ug11, "matrix")})

  user  system elapsed
 0.72    0.00    0.73

> system.time({for (ii in 1:200) as.adjMAT(ug11)})

  user  system elapsed
 0.03    0.00    0.03

```

Similarly, consider finding the cliques of an undirected graph represented as a **graphNEL** object or as a matrix:

```
> system.time({for (ii in 1:200) maxClique(ug11)})  
  
   user  system elapsed  
  0.25    0.00    0.25  
  
> system.time({for (ii in 1:200) maxCliqueMAT(ug11m)})  
  
   user  system elapsed  
  0.03    0.00    0.03
```

7.2 Space

On the other hand, the **graphNEL** representation is – at least – in principle more economic in terms of space requirements than the adjacency matrix representation (because the adjacency matrix representation uses a 0 to represent a “missing edge”. However, in practice the picture is not so clear. Consider the following examples

```

> V <- 1:100
> M <- 1:10
> ## Sparse graph
> ##
> g1 <- randomGraph(V, M, 0.05)
> length(edgeList(g1))

[1] 104

> object.size(g1)

76600 bytes

> object.size(as.adjMAT(g1))

87448 bytes

> ## More dense graph
> ##
> g1 <- randomGraph(V, M, 0.2)
> length(edgeList(g1))

[1] 1816

> object.size(g1)

870504 bytes

> object.size(as.adjMAT(g1))

87448 bytes

> ## Even more dense graph
> ##
> g1 <- randomGraph(V, M, 0.5)
> length(edgeList(g1))

[1] 4771

> object.size(g1)

2241624 bytes

> object.size(as.adjMAT(g1))

87448 bytes

```