The graphs in `gRain` - the `gRash` !!

Søren Højsgaard

May 1, 2008

Contents

```
> options(width = 105)
```

1 Introduction

This note describes a simple the "graph system" used in the `gRain` package. We refer to these this graph system as `gRash`. Thus `gRash` is not an R package but a part of an `R` package.

For the `R` community, the three packages `graph`, `RBGL` and `Rgraphviz` are extremely useful tools for graph operations, manipulation and layout. The `gRash` system is not intended as a competitor for these fine packages. On the contrary, parts of the `gRash` functionality use these packages.

However, `gRain` implement some additional graph operations, (for example graph triangulations, maximum cardinality search and creating a RIP (running intersection property) ordering of the cliques of a decomposable graph). Another virtue of the `gRash` system is that graphs are specified in a way closer to normal text book representations. The same applies to some extent to the graph operations. Only undirected and directed acyclic graphs are implemented.

2 Graphs

2.1 Undirected graphs

An undirected graph is created by the `newug()` function. The graph can be specified by a formula (or a list of formulas): Thus the following two forms are equivalent:

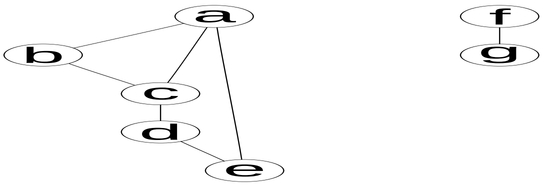
```
> ug1 <- newug(~a * b * c, ~c * d, ~d * e, ~e * a, ~f * g)

Undirected graph with 7 nodes and 7 edges

> ug12 <- newug(~a * b * c + c * d + d * e + a * e + f * g)

Undirected graph with 7 nodes and 7 edges
```

Instead of "`*`", a "`:`" can be used in the specification. Alternatively one can specify a graph as:

```
> ug13 <- newug(c("a", "b", "c"), c("c", "d"), c("d", "e"), c("a", "e"), c("f", "g"))

Undirected graph with 7 nodes and 7 edges
```

Graphs are displayed with `plot()`:

```
> plot(ug1)
```



2.2 Directed acyclic graphs

A directed acyclic graph can be specified as a collection of formulas:

```
> dag1 <- newdag(~a, ~b * a, ~c * a * b, ~d * c * e, ~e * a, ~g * f)

Directed graph with 7 nodes and 7 edges
```

Here `~a` means that "a" has no parents while `~d*b*c` means that "d" has parents "b" and "c".

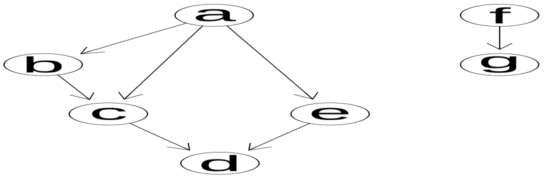
Instead of "`*`", a "`:`" can be used in the specification.

Alternatively one can specify a graph as:

```
> dag12 <- newdag("a", c("b", "a"), c("c", "a", "b"), c("d", "c", "e"), c("e", "a"),
+     c("g", "f"))

Directed graph with 7 nodes and 7 edges
```

As before, graphs are displayed with `plot()`:

```
> plot(dag1)
```



If a directed graph contains cycles, then NULL is returned:

```
> newdag(~a:b, ~b:c, ~c:a)

NULL
```

3 Operations on undirected graphs

3.1 Simple operations

Simple operations on undirected graphs are:

```
> nodes(ug1)

a b c d e f g

> edges(ug1)

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

3.2 Graph queries

Many features of a graph are obtained by asking queries using the `queryg` function:

3.2.1 Nodes

```
> queryg(ug1, "nodes")

a b c d e f g
```

3.2.2 Edges

```
> queryg(ug1, "edges")

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

3.2.3 Cliques

```
> queryg(ug1, "cliques")

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

3.2.4 Connected components

```
> queryg(ug1, "concomp")

a b c d e
f g
```

3.2.5 Closure

```
> queryg(ug1, "cl", "c")

c a b d
```

3.2.6 Adjacencies

```
> queryg(ug1, "adj", "c")

a b d
```

3.2.7 Simplicial nodes

Nodes whose boundary is complete.

```
> queryg(ug1, "simplicialNodes")

b f g
```

3.2.8 Is complete

Is the graph complete?

```
> queryg(ug1, "is.complete")

[1] FALSE
```

3.2.9 Is simplical

Is a node/set simplical?

```
> queryg(ug1, "is.simplicial", "a")

[1] FALSE

> queryg(ug1, "is.simplicial", c("a", "b", "d"))

[1] FALSE
```

3.2.10 Is triangulated

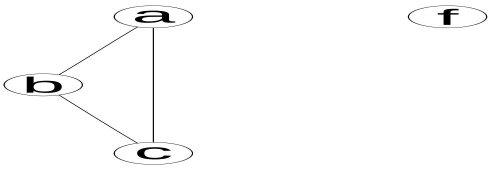
```
> queryg(ug1, "is.triangulated")

[1] FALSE
```

3.2.11 Is A and B separated by S

```
> queryg(ug1, "separates", c("a", "b"), c("d", "f"), "c")

[1] FALSE

> queryg(ug1, "separates", c("a", "b"), c("d", "f"), c("c", "e"))

[1] TRUE
```

3.2.12 Subgraph

```
> queryg(ug1, "subgraph", c("a", "b", "c", "f"))

Undirected graph with 4 nodes and 3 edges

> plot(queryg(ug1, "subgraph", c("a", "b", "c", "f")))
```



3.3 Triangulation and Maximum Cardinality Search

3.3.1 Maximum cardinality search

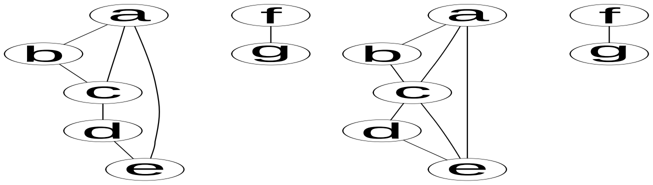
Testing for whether a graph is triangulated is based on Maximum Cardinality Search. If NULL is returned the graph is not triangulated. Otherwise a linear ordering of the nodes is returned.

```
> mcs(ug1)

NULL
```

3.3.2 Triangulation

```
> tug1 <- triangulate(ug1)

Undirected graph with 7 nodes and 8 edges
```

```
> par(mfrow = c(1, 2))
> plot(ug1)
> plot(tug1)
```



3.3.3 RIP (running intersection property) ordering of the cliques

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

```
> rip <- ripOrder(tug1)
> names(rip)

nodes cliques separators pa nLevels

> rip

Cliques
  1 c a b
  2 e a c
  3 d c e
  4 g f
Separators
  1 NA
  2 a c
  3 c e
  4 NA
Parents
  1 NA
  2 1
  3 2
  4 NA
```

4 Operations on directed acyclic graphs

4.1 Simple operations

Simple operations on directed acyclic graphs are:

```
> nodes(dag1)

a b c d e g f

> edges(dag1)

b a
c a
c b
d c
d e
e a
g f

> vpav(dag1)

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

4.2 Graph queries

Many features of a graph are obtained by asking queries using the `queryg` function as above:

4.2.1 Parents

```
> queryg(dag1, "pa", "d")

c e
```

4.2.2 Children

```
> queryg(dag1, "ch", "c")

d
```

4.2.3 Ancestral set

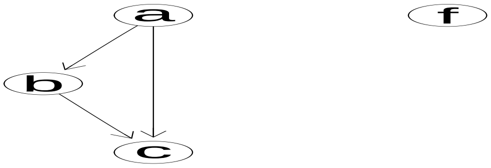
```
> queryg(dag1, "ancestralSet", c("b", "e"))

a b e
```

4.2.4 Ancestral graph

```
> queryg(dag1, "ancestralGraph", c("b", "e"))

Directed graph with 3 nodes and 2 edges
```

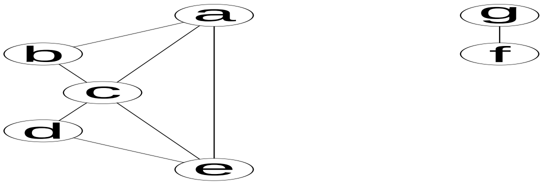
4.2.5 Subgraph

```
> queryg(dag1, "subgraph", c("a", "b", "c", "f"))

Directed graph with 4 nodes and 3 edges

> plot(queryg(dag1, "subgraph", c("a", "b", "c", "f")))
```



4.3 Moralization

```
> moralize(dag1)

Undirected graph with 7 nodes and 8 edges

> plot(moralize(dag1))
```



5 Conversion to different formats

A graph can be converted to 1) an adjacency matrix or 2) a `graphNEL` object (which is one of the formats of graphs used in the `graph` package).

```
> as.adjmat(ug1)

  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.graphNEL(ug1)

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

```
> as.adjmat(dag1)

  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.graphNEL(dag1)

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