Algebraic Analysis and Visualization of Multiple, Signed, and Two-mode Networks with ‘multiplex’ & ‘multigraph’

Antonio Rivero Ostoic

github.com/mplex

Sunbelt XXXVII ☆ Beijing, China ☆ 31 May 2017

Agenda

visualization and algebraic analyses . . .

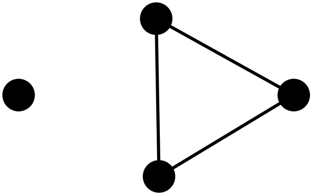
1. Introduction
2. Data sets
3. Relational Structure
4. Signed Networks
5. Affiliation Networks

(6. Miscellaneous)

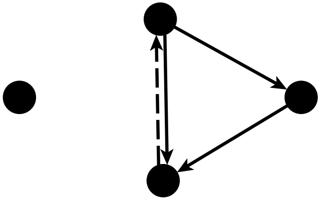
1. Introduction

Multiple Networks

- Social networks are typically characterized by a single relationship
- But social life is more complex and people are embedded in different types of relations that are **interlocked** to each other



graph depicting a **simple** network



multigraph depicting a **multiple** network

☛ *find the right methods to analyse multiple types of tie simultaneously*

Dyadic properties

multiple networks

Tie Entrainment

Asymmetric character

Tie Exchange

Mutual character

Mixed pattern

Full pattern

Mutual character

Create multivariate network data

```
# SET WORKING DIRECTORY (e.g.)
setwd("C:/sunbelt")
```

```
# INSTALL multigraph FROM CRAN (INSTALL multiplex AS WELL)
install.packages("multigraph")

# BETA VERSION FOR LOOPS
library("devtools")
devtools::install_github("mplex/multigraph", ref = "beta")
install.packages("multiplex")
```

Create multivariate network data

```r
# CREATE PSEUDO RANDOM DATA
library("stats")
set.seed(123); arr1 <-  array(runif(9), c(3, 3, 1))
set.seed(321); arr2 <-  array(runif(9), c(3, 3, 1))
```

```r
# CREATE 3D ARRAY OF MULTIPLE NETWORK 'arr'
library("multiplex")
arr <-  zbind(arr1, arr2)
arr <-  dichot(arr, c = .5)
```

```
, , 1

     [,1] [,2] [,3]
[1,]    0    1    1
[2,]    1    1    1
[3,]    0    0    1

, , 2

     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    1    0    0
[3,]    0    0    0
```

Create multivariate network data

With the `transf` function:

```
# v2.6+
transf(list(c("1, 2", "1, 3", "2, 1", "2, 2", "2, 3", "3, 1"),
+            c("1, 1", "2, 1")), type = "toarray")
```

```
, , 1

     [,1] [,2] [,3]
[1,]    0    1    1
[2,]    1    1    1
[3,]    0    0    1

, , 2

     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    1    0    0
[3,]    0    0    0
```

Bundle Patterns

```
# FUNCTION summaryBundles() REQUIRES A "Rel.Bundles" CLASS OBJECT
summaryBundles(bundles(arr))
```

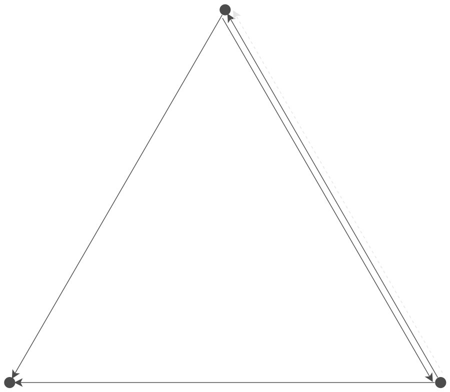
```
                   Bundles
Asym1         ->{1} (1, 3)
Asym2         ->{1} (2, 3)
Mixd  <->{1} <-{2}  (1, 2)
```

```
bundle.census(arr)
```

```
      BUNDLES NULL ASYMM RECIP T.ENTR T.EXCH MIXED FULL
TOTAL       3    0     2     0      0      0     1    0
```

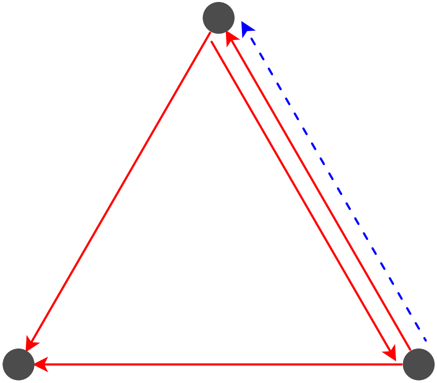
Network visualization

```
# LOOK AT THE MULTIGRAPH OF NETWORK 'arr'
library("multigraph")
multigraph(arr)
```



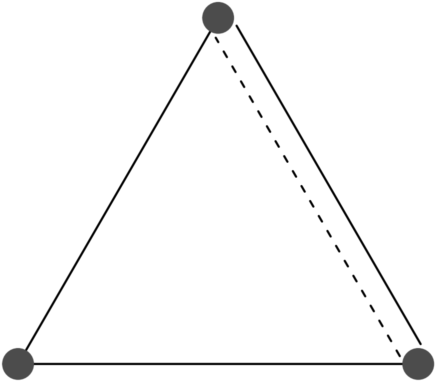
Network visualization

```
# ADD VERTEX / EDGE / GRAPH CHARACTERISTICS
multigraph(arr, cex = 6, lwd = 3, ecol = c('red','blue'))
```



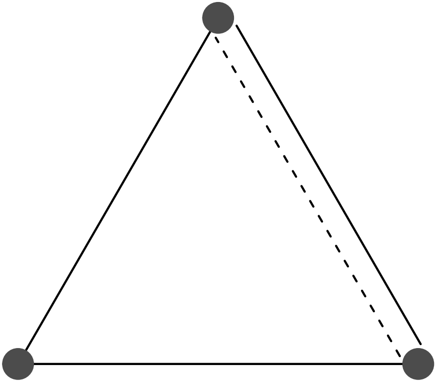
Network visualization

```
# ADD VERTEX / EDGE / GRAPH CHARACTERISTICS
multigraph(arr, cex = 6, lwd = 3, ecol = 1, directed = FALSE)
```



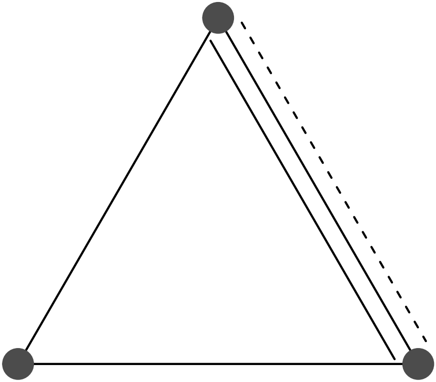
Network visualization

```r
# DEFINE A 'list' OF VERTEX / EDGE / GRAPH CHARACTERISTICS
otl <- list(cex = 6, lwd = 3, ecol = 1, directed = FALSE)
multigraph(arr, outline = otl)
```



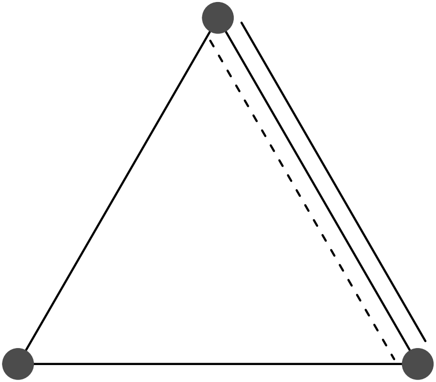
Network visualization

```r
# DO NOT COLLAPSE RECIPROCATED TIES (UNDIRECTED ONLY)
multigraph(arr, outline = otl, collRecip = FALSE)
```



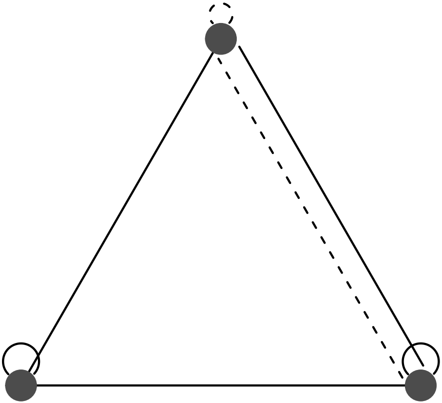
Network visualization

```
# ALSO SWAP BUNDLE TIES...
multigraph(arr, outline = otl, collRecip = FALSE, swp = TRUE)
```



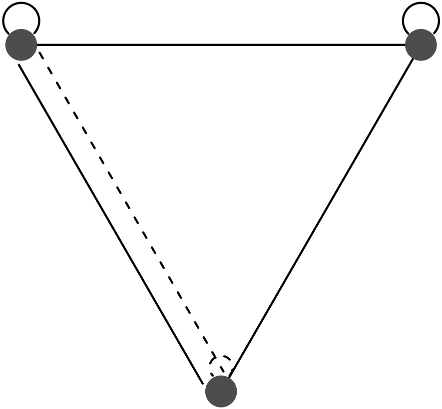
Network visualization

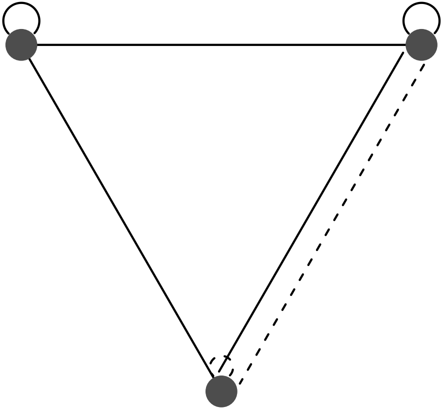
```
# SHOW LOOPS WITH THE COSTUMIZED CHARACTERISTICS
multigraph(arr, outline = otl, loops = TRUE)
```



Network visualization

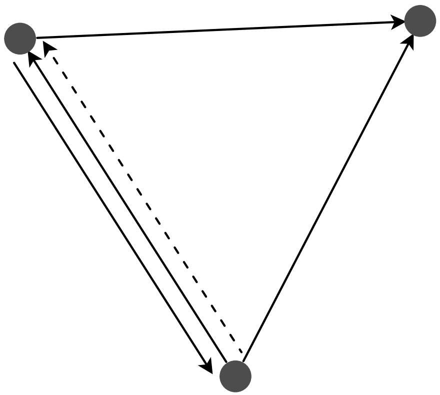
```r
# TWO WAYS TO MAKE A TRANSFORMATION OF THE GRAPH
multigraph(arr, outline = otl, loops = TRUE, rot = 180)
multigraph(arr, outline = otl, loops = TRUE, mirrorY = TRUE)
```





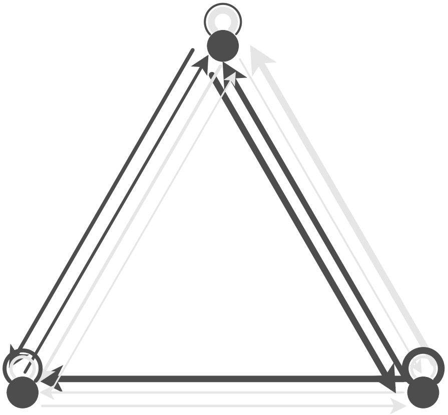
Network visualization

```
# APPLY A FORCE DIRECTED LAYOUT TO THE DIGRAPH
multigraph(arr, outline = c(otl, directed = TRUE), layout = "force", seed = 1)
```



Weighted network visualization

```
# WEIGHTED MULTIGRAPH WITH LOOPS  —  Obtained from zbind()
multigraph(zbind(arr1*10, arr2*10), cex = 6, weighted = TRUE, loops = TRUE)
```



2. Data sets

Incubator network

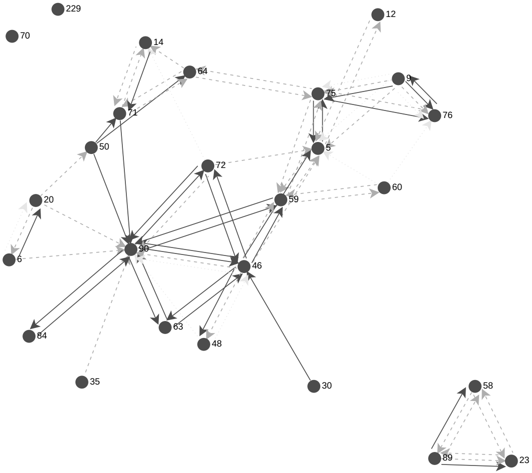
```r
# 'INCUBATOR A' NETWORK DATA SET
data("incubA")
str(incubA)
```

```
List of 2
 $ net: num [1:26, 1:26, 1:5] 0 0 0 0 0 0 0 0 0 1 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ : chr [1:26] "5" "6" "9" "12" ...
  .. ..$ : chr [1:26] "5" "6" "9" "12" ...
  .. ..$ : chr [1:5] "C" "F" "K" "A" ...
 $ IM : num [1:4, 1:4, 1:7] 1 1 1 0 0 1 0 0 1 0 ...
...
```

```r
# RECORD NETWORK AND ACTOR ATTRIBUTES
netA <- incubA$net[,,1:3]
attA <- incubA$net[,,4:5]
```

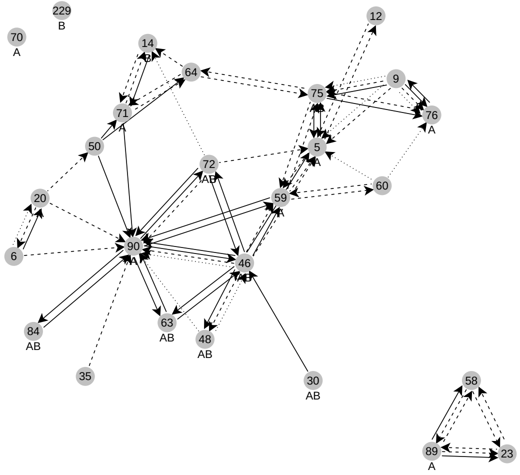
Incubator network

```
multigraph(netA, layout = "force", seed = 1)
```



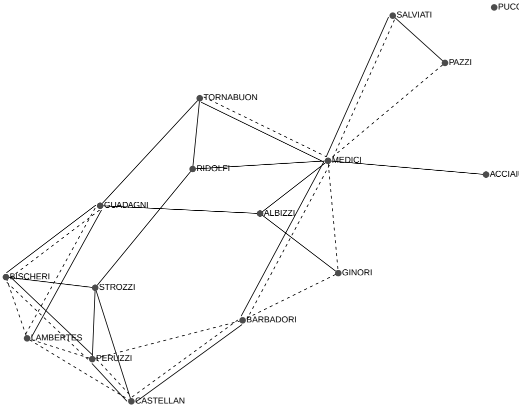
Incubator network

```
otlA <-  list(ecol = 1, vcol = "#C0C0C0", cex = 3, tcex = .8, pos = 0, bwd = .5)
multigraph(netA, layout = "force", seed = 1, outline = otlA, att = attA)
```



Florentine families, Padgett (undirected network)

```
# FLORENTINE FAMILIES DATA SET AS A UCINET DL FILE
flf <-  read.dl(file = "http://moreno.ss.uci.edu/padgett.dat")
multigraph(flf, directed = FALSE, layout = "force", seed = 1, ecol = 1)
```



Or locally `read.dl(file ="data/padgett.dl")`

Florentine families, Padgett (undirected network)

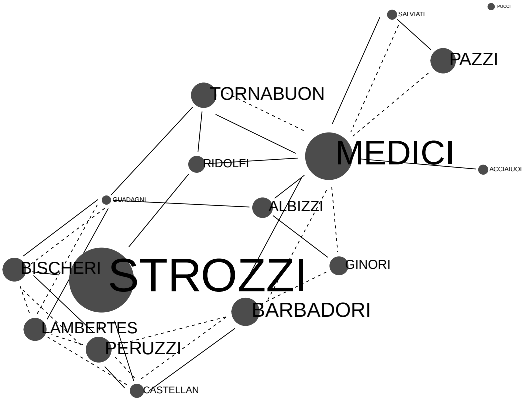
```r
# ACTOR ATTRIBUTES
flfa <- read.dl(file = "http://moreno.ss.uci.edu/padgw.dat")
flfa <- flfa[order(rownames(flfa)), ]
```

```
          WEALTH #PRIORS #TIES
ACCIAIUOL     10      53     2
ALBIZZI       36      65     3
BARBADORI     55       0    14
BISCHERI      44      12     9
CASTELLAN     20      22    18
GINORI        32       0     9
GUADAGNI       8      21    14
LAMBERTES     42       0    14
MEDICI       103      53    54
PAZZI         48       0     7
PERUZZI       49      42    32
PUCCI          3       0     1
RIDOLFI       27      38     4
SALVIATI      10      35     5
STROZZI      146      74    29
TORNABUON     48       0     7
```

Or locally `read.dl(file ="data/padgw.dl")`

Florentine families, Padgett (undirected network)

```
# PLOTTING WITH ACTOR ATTRIBUTES
multigraph(flf, directed = FALSE, "force", seed = 1, ecol = 1, cex = flfa[,1])
```



G20 Countries (weighted network)

```
# LOAD DATA G20 COUNTRIES TRADE MILK AND HONEY 2017
load("data/fmlkhny2015.Rdata")

# LOOK AT THE THREE FIRST ACTORS
fmlkhny2015[1:3, 1:3, ]
```

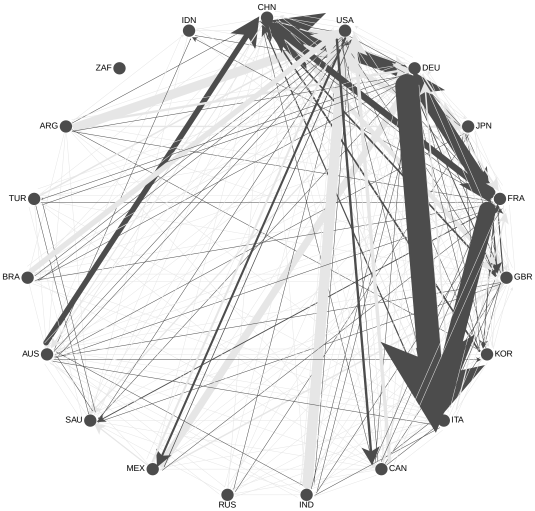
```
, , M

          CHN   USA   DEU
CHN         0 53982     0
USA   1831891     0 13586
DEU 144173827 88390     0

, , H

        CHN     USA      DEU
CHN       0   10140 12987385
USA 1119310       0    13945
DEU 2964219 2948088        0
```

☛ Or obtain the data from Github

```
load(url("https://github.com/mplex/sunbelt2017/raw/master/data/fmlkhny2015.Rdata"))
```

G20 Countries (weighted network)

```r
multigraph(fmlkhny2015, weighted = TRUE)
```



G20 Countries (weighted network)

```
# LOAD DATA G20 ECONOMIC FACTS
load("data/g20dat.Rdata")
```

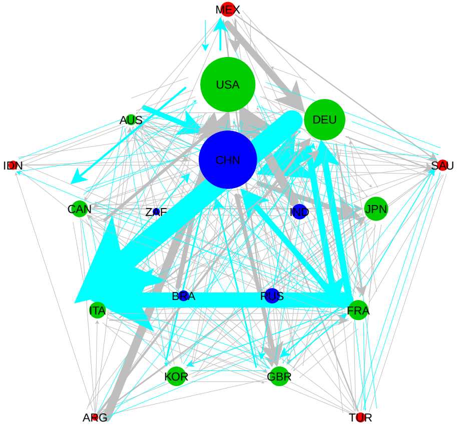
Or `load(url("https://github.com/mplex/sunbelt2017/raw/master/data/g20dat.Rdata"))`

```
      Trade Nom._GDP PPP_GDP_  GDP_PC PPP_GDP_PC   HDI Population     Area Eco_class
CHN 4201000 10982829 19510000   7990       14107 0.727 1367520000  9572900     BRICS
USA 3944000 17947000 17947000  55805       55805 0.915  318523000  9526468  Advanced
DEU 2866600  3357614  3842000  40997       46893 0.916   80940000   357114  Advanced
JPN 1522400  4123258  4658000  32486       38054 0.891  127061000   377930  Advanced
FRA 1212300  2421560  2647000  37675       41181 0.888   63951000   640679  Advanced
GBR 1189400  2849345  2660000  43771       41159 0.907   64511000   242495  Advanced
KOR 1170900  1376868  1849000  27195       36511 0.898   50437000   100210  Advanced
ITA  948600  1815757  2174000  29867       35708 0.873   60665551   301336  Advanced
CAN  947200  1552386  1632000  43332       44967 0.913   35467000  9984670  Advanced
IND  850600  2090706  7965000   1617        6162 0.609 1259695000  3287263     BRICS
RUS  844200  1324734  3471000   9055       25411 0.798  146300000 17098242     BRICS
MEX  813500  1144334  2220000   9009       17534 0.756  119581789  1964375  Emerging
SAU  521600   653219  1683000  20813       53624 0.837   30624000  2149690  Emerging
AUS  496700  1223887  1489000  50962       47389 0.935   23599000  7692024  Advanced
BRA  484600  1772589  3166000   8670       16155 0.755  202768000  8515767     BRICS
TUR  417000   733642  1589000   9437       20438 0.761   77324000   783562  Emerging
ARG  142370   585623   964300  13589       22554 0.836   42961000  2780400  Emerging
ZAF  200100   312957   723518   5695       13165 0.666   53699000  1221037     BRICS
IDN  346100   858953  2839000   3362       11126 0.684  251490000  1904569  Emerging
```

G20 Countries (weighted network)

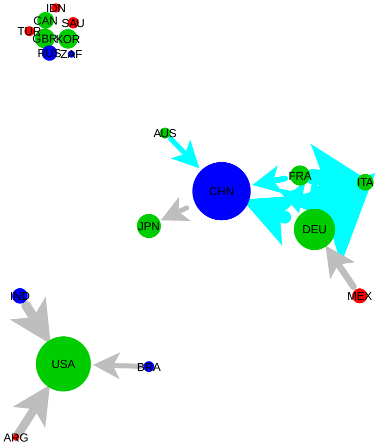
```
# DEFINE CHARACTERISTICS
cls <- g20dat[,9]
otlg20 <- list(vcol = 4:2, clu = cls, pos = 0, cex = g20dat[,1], tcex = 1, ecol = c(5,8), weighted = TRUE)
```

```
multigraph(fmlkhny2015, layout = "conc", nr = cls, outline = otlg20)
```



G20 Countries (weighted network)

```
# DROP TRADE UNDER 50000000 IN FORCE DIRECTED PLOT
multigraph(fmlkhny2015, layout = "force", seed = 2, outline = otlg20,
+     drp = 50000000)
```



G20 Countries (affiliation network)

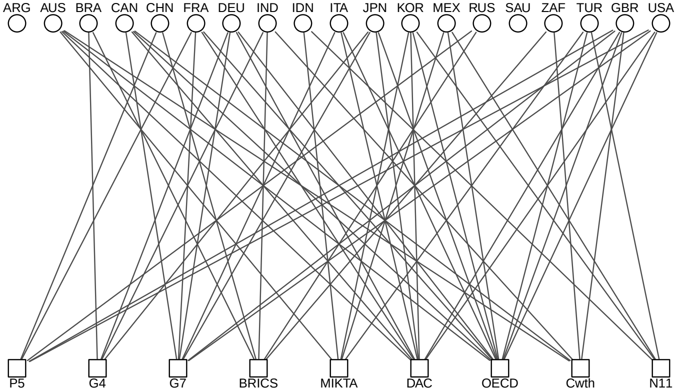
```r
# LOAD DATA G20 ECONOMIC FACTS
load("data/G20.Rdata")
```

Or `load(url("https://github.com/mplex/sunbelt2017/raw/master/data/G20.Rdata"))`

```
    P5 G4 G7 BRICS MIKTA DAC OECD Cwth N11
ARG  0  0  0     0     0   0    0    0   0
AUS  0  0  0     0     1   1    1    1   0
BRA  0  1  0     1     0   0    0    0   0
CAN  0  0  1     0     0   1    1    1   0
CHN  1  0  0     1     0   0    0    0   0
FRA  1  0  1     0     0   1    1    0   0
DEU  0  1  1     0     0   1    1    0   0
IND  0  1  0     1     0   0    0    1   0
IDN  0  0  0     0     1   0    0    0   1
ITA  0  0  1     0     0   1    1    0   0
JPN  0  1  1     0     0   1    1    0   0
KOR  0  0  0     0     1   1    1    0   1
MEX  0  0  0     0     1   0    1    0   1
RUS  1  0  0     1     0   0    0    0   0
SAU  0  0  0     0     0   0    0    0   0
ZAF  0  0  0     1     0   0    0    1   0
TUR  0  0  0     0     1   0    1    0   1
GBR  1  0  1     0     0   1    1    1   0
USA  1  0  1     0     0   1    1    0   0
```

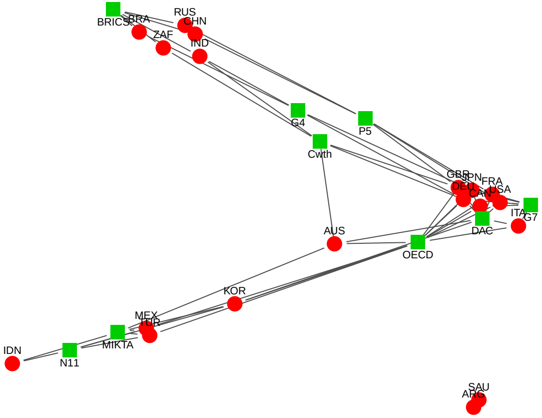
G20 Countries (affiliation network)

```r
# BIPARTITE GRAPH OF 'G20'
bmgraph(G20, rot = 90, mirrorX = TRUE)
```



G20 Countries (affiliation network)

```
# APPLY CORRESPONDENCE ANALYSIS TO THE PLOT
bmgraph(G20, layout = "CA", rot = 99, vcol = 2:3, pch = c(19, 15), jitter = .1)
```



3. Relational structure

Tie Interlock

- **Social structure** = Ties between actors

- **Relational structure** = Interrelations between relations

- **Role structure** = Relational system of aggregated relations

☛ we benefit from algebraic structures to represent relational systems

Semigroup

Algebraic structure

A **semigroup** is an algebraic structure with a set of elements with an associative operation attached to it:

$$\langle\ S,\ \circ\ \rangle$$

- S is the underlying set, closed under the operation
- $\circ$ is a binary operation on an ordered pair, i.e. $\circ\colon S \times S \to S$ that, for all $x, y, z \in S$ satisfies the associative law:

$$x \circ (y \circ z) \ = \ (x \circ y) \circ z$$

➠ $\circ$ is called the ‘composition’ operation

Semigroup of Relations

- In a **semigroup of relations** $S(R)$, 'x' and 'y' are **generators** (or primitives), whereas '$x \circ y$' constitutes a **compound** relation

- $S(R)$ represents the relational structure in multiplex networks

- The elements in $S(R)$ are the *unique* representative strings made of generator(s) and –most likely– compounds relations as well

- Unique strings are obtained after **equating** the occurring ties in the system, i.e. the Axiom of Quality (Boorman & White, 1976)

Partial Order

- A **partial order** is defined by an *inclusion* relation $\leq$ among $x, y \in S$ with the rule:

$$S^{\leq}_{x,y} = \begin{cases} 1 & \text{iff relation } x \text{ is contained in relation } y \\ 0 & \text{otherwise} \end{cases}$$

where 'contained' implies that all ties in x are occurring in y as well

➟ A **partially ordered semigroup** (Pattison, 1993) is S with a partial order

String relations

```r
# SOME REPRESENTATIVE STRINGS IN 'arr'
strings(arr)
```

```
$wt

...
, , 11

     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    0    0    1

, , 21

     [,1] [,2] [,3]
[1,]    0    1    1
[2,]    0    1    1
[3,]    0    0    0

, , 211

     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    0    0    0

...
```

Semigroup of relations

```
# SEMIGROUP OF RELATIONS IN NUMERICAL FORMAT
semigroup(arr)$S
```

```

  1 2 3 4 5
1 3 2 3 4 5
2 4 2 5 4 5
3 3 2 3 4 5
4 5 2 5 4 5
5 5 2 5 4 5

...
```

Semigroup of relations

```
# SEMIGROUP OF RELATIONS IN SYMBOLIC FORMAT
semigroup(arr, type = "symbolic")$S
```

```
      1 2   11 21 211
1    11 2   11 21 211
2    21 2  211 21 211
11   11 2   11 21 211
21  211 2  211 21 211
211 211 2  211 21 211

...
```

Equations in the semigroup

```r
# EQUATIONS OF COMPOUNDS UNTIL LENGTH 3
strings(arr, equat = TRUE, k = 3)$equat
```

```
$`2`
[1] "2"   "22"  "12"  "122" "112" "222" "212"

$`11`
[1] "11"  "111"

$`21`
[1] "21"  "221" "121"
```

Partial Order

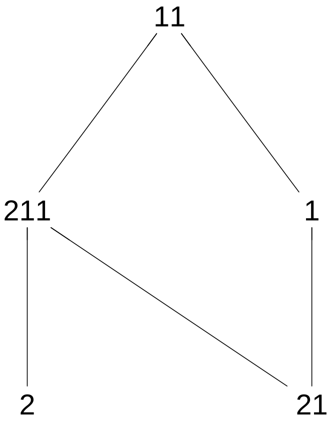
```
# PARTIAL ORDER TABLE OF STRING RELATIONS
partial.order(strings(arr))
```

```
    1 2 11 21 211
1   1 0  1  0   0
2   0 1  1  0   1
11  0 0  1  0   0
21  1 0  1  1   1
211 0 0  1  0   1
attr(,"class")
[1] "Partial.Order" "strings"
```

Hasse Diagram

Visualization of partial order structures

```
# FUNCTION diagram() PLOTS POSETS. REQUIRES "Rgraphviz" package
diagram(partial.order(strings(arr)))
```



Issues with the Semigroup Structure

- Modelling a multiple network by $S(R)$ results in a quite large and complex structure, even if the system is small

- An important task is to reduce complexity of the network
  ➟ by grouping different classes of actors

- *Blockmodeling* is an effective way to reduce the network and keeping the essential structure of the system

➥ **Positional Analysis**

☛ But it needs to preserve the network multiplicity of ties

Positional Analysis

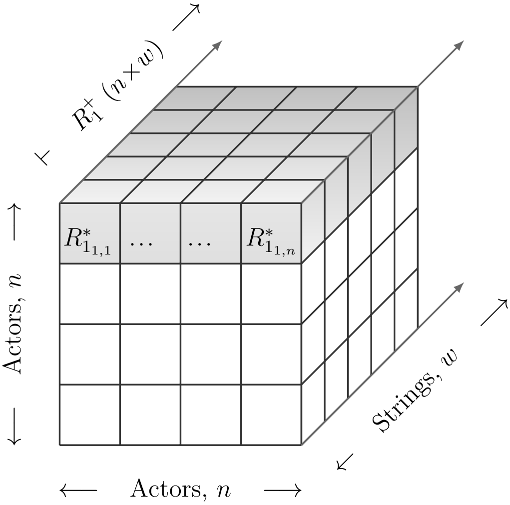
Equivalences types

Each *type* of graph homomorphism (a structure-preserving mapping) induces to a particular kind of equivalence

➟ which represents a system of *positions* and *roles* of the network

- Equivalences from a **global perspective**:
  - Structural (Lorrain & White, 1971)
  - Automorphic (Winship & Mandel, 1983; Everett, 1985)
  - Regular (Sailer [Boyd], 1978; White & Reitz, 1983)
  - Generalized (Batagelj et al, 1992; Doreian et al, 1994)
- Equivalences from a **local perspective**:
  - Local Role (Winship & Mandel, 1983; Mandel, 1983)
  - **Compositional** (Breiger & Pattison, 1986; Mandel, 1978)

Compositional Equivalence: Relation-Box



Compositional Equivalence: Person Hierarchies

Person Hierarchies

- Builds on the ordering among the actors' Role Relations in a particular Relation Plane (shadow part in the Relation-Box)

- All perceived inclusions in R_l^+ represents the **Person Hierarchy** H_l defined for $l, i, j \in \mathscr{X}$ and relation x as:

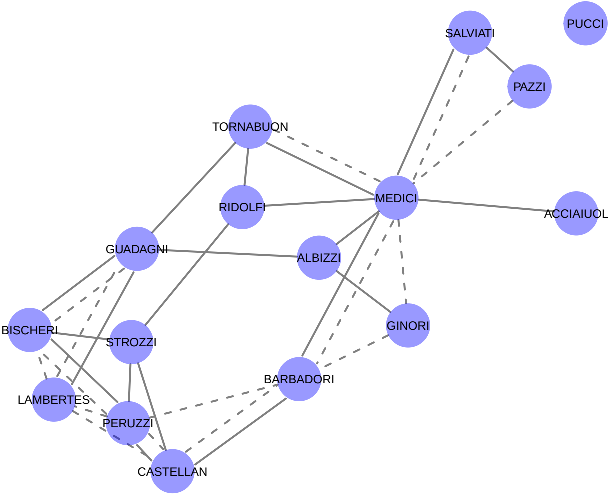
$$H_{l_{ij}} = \begin{cases} 1 & \text{iff} \;\; R^*_{l_{xi}} \leq R^*_{l_{xj}} \\ 1 & \text{iff} \;\; R^*_{l_{xi}} = R^*_{l_{xj}} \\ 0 & \text{iff} \;\; R^*_{l_{xi}} \nleq R^*_{l_{xj}} \\ 0 & \text{iff} \;\; \sum R^*_{l_{xi}} = 0 \end{cases}$$

- A **Cumulated Person Hierarchy** $\mathscr{H}$ matrix is based on the union of all person hierarchies with transitive closure

  ➟ the establishment of roles and positions are from the perspectives of individual actors, but it also considers common relational features

Compositional Equivalence: Undirected Networks

Florentine Families. Solid: Marriage. Dashed: Business



Florentine Families

```r
# INSPECT THE NETWORK RELATIONAL SYSTEM
rel.sys(flf, bonds = "full")$incl
```

```
[1] "BARBADORI" "BISCHERI"  "CASTELLAN" "GUADAGNI"  "LAMBERTES" "MEDICI"    "PERUZZI"
[8] "SALVIATI"  "TORNABUON"
```

```r
# WHO IS NOT LINKED AT BOTH LEVELS
rel.sys(flf, bonds = "full")$excl
```

```
[1] "ACCIAIUOL" "ALBIZZI"   "GINORI"    "PAZZI"     "PUCCI"     "RIDOLFI"   "STROZZI"
```

Compositional Equivalence: Relation-Box

```
# FUNCTION TO CONSTRUCT THE RELATION—BOX
formals("rbox")
```

```
$w


$transp
[1] FALSE

$smpl
[1] FALSE

$k
[1] 3

$tlbs
```

Compositional Equivalence: Cumulated Person Hierarchy

```
# FUNCTION cph() TO CONSTRUCT THE CUMULATED PERSON HIERARCHY
# INPUT MUST BE A "Rel.Box" CLASS. OUTPUT IS A "Partial.Order" "CPH" CLASS
cph(rbox(flf))
```

```
          ACCIAIUOL ALBIZZI BARBADORI BISCHERI CASTELLAN GINORI GUADAGNI LAMBERTES MEDICI PAZZI
ACCIAIUOL         1       1         1        1         1      1        1         1      1     1
ALBIZZI           1       1         1        1         1      1        1         1      1     1
BARBADORI         1       1         1        1         1      1        1         1      1     1
BISCHERI          1       1         1        1         1      1        1         1      1     1
CASTELLAN         1       1         1        1         1      1        1         1      1     1
GINORI            1       1         1        1         1      1        1         1      1     1
GUADAGNI          1       1         1        1         1      1        1         1      1     1
LAMBERTES         1       1         1        1         1      1        1         1      1     1
MEDICI            1       1         1        1         1      1        1         1      1     1
PAZZI             1       1         1        1         1      1        1         1      1     1
PERUZZI           1       1         1        1         1      1        1         1      1     1
PUCCI             0       0         0        0         0      0        0         0      0     0
RIDOLFI           1       1         1        1         1      1        1         1      1     1
SALVIATI          1       1         1        1         1      1        1         1      1     1
STROZZI           1       1         1        1         1      1        1         1      1     1
TORNABUON         1       1         1        1         1      1        1         1      1     1
attr(,"class")
[1] "Partial.Order" "CPH"
```

(Extract)

Compositional Equivalence: Cumulated Person Hierarchy

```r
cph(rbox(flf, k = 4))
```

```
          ACCIAIUOL ALBIZZI BARBADORI BISCHERI CASTELLAN GINORI GUADAGNI LAMBERTES MEDICI PAZZI
ACCIAIUOL         1       1         1        1         1      1        1         1      1     1
ALBIZZI           1       1         1        1         1      1        1         1      1     1
BARBADORI         1       1         1        1         1      1        1         1      1     1
BISCHERI          1       1         1        1         1      1        1         1      1     1
CASTELLAN         1       1         1        1         1      1        1         1      1     1
GINORI            1       1         1        1         1      1        1         1      1     1
GUADAGNI          1       1         1        1         1      1        1         1      1     1
LAMBERTES         1       1         1        1         1      1        1         1      1     1
MEDICI            1       1         1        1         1      1        1         1      1     1
PAZZI             1       1         1        1         1      1        1         1      1     1
PERUZZI           1       1         1        1         1      1        1         1      1     1
PUCCI             0       0         0        0         0      0        0         0      0     0
RIDOLFI           1       1         1        1         1      1        1         1      1     1
SALVIATI          1       1         1        1         1      1        1         1      1     1
STROZZI           1       1         1        1         1      1        1         1      1     1
TORNABUON         1       1         1        1         1      1        1         1      1     1
attr(,"class")
[1] "Partial.Order" "CPH"
```

(Extract)

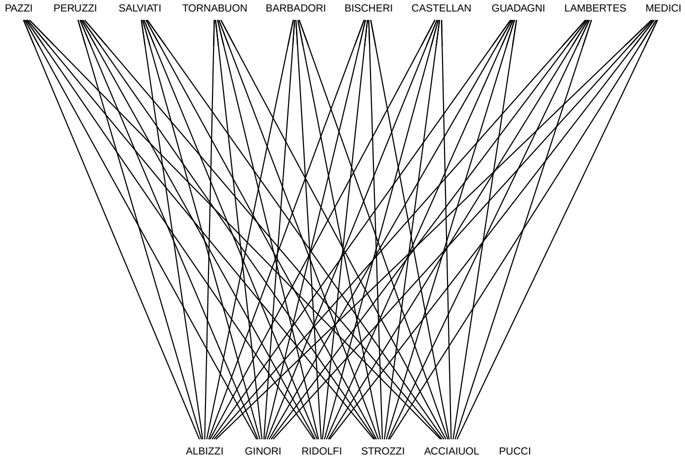
Compositional Equivalence: Cumulated Person Hierarchy

```
# TEST OBJECTS FOR EXACT EQUALITY
identical(cph(rbox(flf, k = 3)), cph(rbox(flf, k = 4)))

[1] TRUE
```

```
identical(cph(rbox(flf, k = 4)), cph(rbox(flf, k = 5)))

[1] FALSE
```

Visualization of the Poset

```
# CPH IS A POSET
diagram(cph(rbox(flf, k = 5)))
```



Compositional Equivalence: Positional Analysis

```
# LEVELS IN THE PLOTED HASSE DIAGRAM
diagram.levels(cph(rbox(flf, k = 5)))
```

```
          2       2         1         1          1      2         1          1      1     1
1 ACCIAIUOL ALBIZZI BARBADORI BISCHERI CASTELLAN GINORI GUADAGNI LAMBERTES MEDICI PAZZI
        1     3       2         1       2          1
1 PERUZZI PUCCI RIDOLFI SALVIATI STROZZI TORNABUON
```

```
# OBTAIN THE CLUSTERING WITH perm ARGUMENT
diagram.levels(cph(rbox(flf, k = 5)), perm = TRUE)$clu
```

```
[1] 2 2 1 1 1 2 1 1 1 1 1 3 2 1 2 1
```

☛ However, levels in the plotted Hasse diagram are not always the best criteria for classifying the actors

Compositional Equivalence: Positional Analysis

```r
# FIRST RECORD THE CLUSTERING VECTOR
flfclu <- diagram.levels(cph(rbox(flf, k = 5)), perm = TRUE)$clu

# APPLY CLUSTERING TO PRODUCE A POSITIONAL SYSTEM WITH FUNCTION reduc()
flfps <- reduc(flf, clu = flfclu)
```

```
, , PADGM

     [,1] [,2] [,3]
[1,]    1    1    0
[2,]    1    1    0
[3,]    0    0    0

, , PADGB

     [,1] [,2] [,3]
[1,]    1    1    0
[2,]    1    0    0
[3,]    0    0    0
```

Compositional Equivalence: Role Structure

```
# THE SEMIGROUP OF THE POSITIONAL SYSTEM IN DEFAULT FORMAT
semigroup(flfps)
```

```
$dim
[1] 3

$gens

...

$ord
[1] 2

$st
[1] "PADGM" "PADGB"

$S
  1 2
1 1 1
2 1 1

attr(,"class")
[1] "Semigroup" "numerical"
```

Compositional Equivalence: Role Structure

```r
# FOR SEMIGROUP IN SYMBOLIC FORMAT WE NEED TO ARRANGE THE TIE LABELS
semigroup(flfps, type = "symbolic", lbs = c("M", "B"))
```

```
$dim
[1] 3

$gens

...

$ord
[1] 2

$st
[1] "M" "B"

$S
  M B
M M M
B M M

attr(,"class")
[1] "Semigroup" "symbolic"
```

Compositional Equivalence: with Actor Attributes

For a given attribute defined in α, and for $i = x_1, x_2, \ldots, x_n$, attribute information is analyzed in relational terms where pair of vectors are element of an indexed matrix $\mathbf{A}^\alpha$ as:

$$a^\alpha_{ij} =; \delta_{ij},$$

Here

$$c_i = \begin{cases} 1 & \text{if the corresponding attribute is tied to actor } i \\ 0 & \text{otherwise.} \end{cases}$$

And δ_{ij} is defined for nodes $i, j = x_1, x_2, \ldots, x_n$ in $\mathscr{X}$ by the Kronecker delta function as:

$$\delta_{ij} = \begin{cases} 1 & \text{for } \; i = j \\ 0 & \text{for } \; i \neq j. \end{cases}$$

☛ That is, $\mathbf{A}^\alpha$ is a *diagonal matrix*.

Actor Attributes in Relational Structures

```
          WEALTH #PRIORS #TIES
ACCIAIUOL     10      53     2
ALBIZZI       36      65     3
BARBADORI     55       0    14
BISCHERI      44      12     9
CASTELLAN     20      22    18
GINORI        32       0     9
GUADAGNI       8      21    14
LAMBERTES     42       0    14
MEDICI       103      53    54
PAZZI         48       0     7
PERUZZI       49      42    32
PUCCI          3       0     1
RIDOLFI       27      38     4
SALVIATI      10      35     5
STROZZI      146      74    29
TORNABUON     48       0     7
```

```
# FUNCTION read.srt() TRANSFORMS DATA FRAMES INTO ARRAYS
read.srt(flfa, attr = TRUE, rownames = TRUE)

# SPLIT RICH ACTORS FROM THE VERY RICH ONES AND BIND IT TO THE NETWORK
fw <-  dichot(read.srt(flfa, attr = TRUE, rownames = TRUE)[, , 1], c = 40)
flfw <-  zbind(flf, fw)
```

Compositional Equivalence: CPH with Actor Attributes

```
# TEST OBJECTS FOR EXACT EQUALITY
identical(cph(rbox(flfw, k = 2)), cph(rbox(flfw, k = 3)))
```

```
[1] TRUE
```

```
identical(cph(rbox(flfw, k = 3)), cph(rbox(flfw, k = 4)))
```

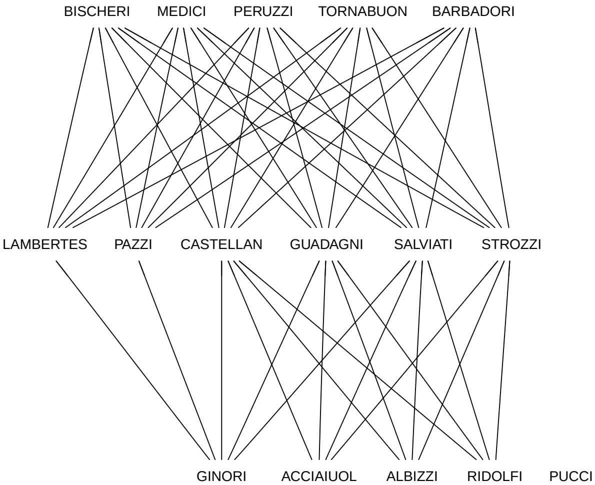
```
[1] TRUE
```

```
identical(cph(rbox(flfw, k = 4)), cph(rbox(flfw, k = 5)))
```

```
[1] FALSE
```

Hasse Diagram of $\mathscr{H}$ with Actor Attributes

```
diagram(cph(rbox(flfw, k = 5)))
```



Positional Analysis with Actor Attributes

```
# POSITIONAL SYSTEM WITH THE CLUSTERING INFO OF THE HASSE DIAGRAM
flfwclu <- diagram.levels(cph(rbox(flfw, k = 5)), perm = TRUE)$clu
flfwps <- reduc(flfw, clu = flfwclu)
```

```
, , PADGM

     [,1] [,2] [,3] [,4]
[1,]    1    1    1    0
[2,]    1    1    1    0
[3,]    1    1    1    0
[4,]    0    0    0    0

, , PADGB

     [,1] [,2] [,3] [,4]
[1,]    1    1    1    0
[2,]    1    1    0    0
[3,]    1    0    0    0
[4,]    0    0    0    0

, , 3

     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    0    0
[4,]    0    0    0    0
```

Algebraic Constraint: Role Table

Role Structure with Actor Attributes

```
# SEMIGROUP OF ROLE RELATIONS WITH COSTUMIZED LABELS
semigroup(flfwps, type = "symbolic", lbs = c("M", "B", "W"))$S

# OR EVEN BETTER...
dimnames(flfwps)[3][[1]] <-  c("M", "B", "W")
semigroup(flfwps, type = "symbolic")$S
```

```
      M  B   W  MW  BW WM WB WMW
M     M  M  MW  MW  MW  M  M  MW
B     M  M  BW  MW  MW  M  M  MW
W    WM WB   W WMW WMW WM WB WMW
MW    M  M  MW  MW  MW  M  M  MW
BW    M  M  BW  MW  MW  M  M  MW
WM   WM WM WMW WMW WMW WM WM WMW
WB   WM WM WMW WMW WMW WM WM WMW
WMW  WM WM WMW WMW WMW WM WM WMW
```

Algebraic Constraint: Set of Equations

Role Structure with Actor Attributes

```r
# FUNCTION strings() SERVES TO FIND EQUATIONS AMONG RELATIONS
strings(flfwps, equat = TRUE, k = 3)$equat
```

```
$M
 [1] "M"   "MM"  "BB"  "MB"  "BM"  "MMM" "BBM" "MBB" "MMB" "BBB" "BMM" "BMB" "MBM" "MWM"
[15] "BWB" "MWB" "BWM"

$W
[1] "W"   "WW"  "WWW"

$MW
[1] "MW"  "MWW" "MMW" "BBW" "MBW" "BMW"

$BW
[1] "BW"  "BWW"

$WM
[1] "WM"  "WWM" "WMM" "WBB" "WMB" "WBM"

$WB
[1] "WB"  "WWB"

$WMW
[1] "WMW" "WBW"
```

Algebraic Constraint: Partial Ordering

Role Structure with Actor Attributes

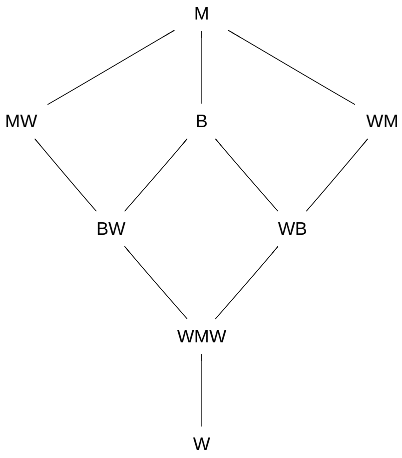
```
# PARTIAL ORDERING OF STRING RELATIONS
partial.order(strings(flfwps), type = "strings")
```

```
    M B W MW BW WM WB WMW
M   1 0 0  0  0  0  0   0
B   1 1 0  0  0  0  0   0
W   1 1 1  1  1  1  1   1
MW  1 0 0  1  0  0  0   0
BW  1 1 0  1  1  0  0   0
WM  1 0 0  0  0  1  0   0
WB  1 1 0  0  0  1  1   0
WMW 1 1 0  1  1  1  1   1
attr(,"class")
[1] "Partial.Order" "strings"
```

Hasse Diagram for the Partial Order of String Relations

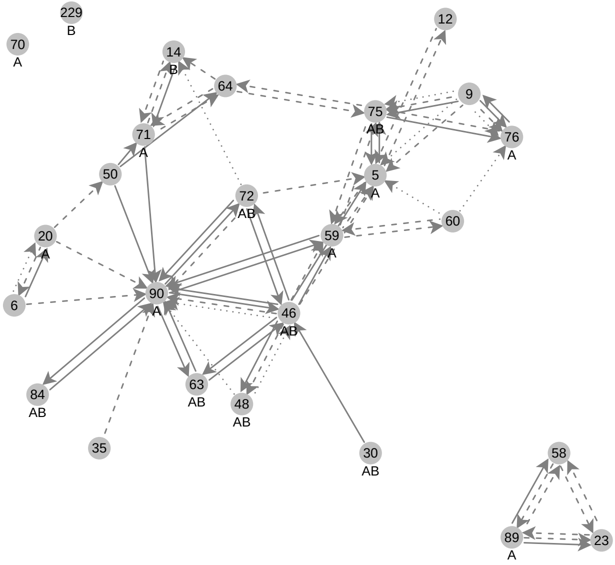
Role Structure with Actor Attributes

```
diagram(partial.order(strings(flfwps), type = "strings"))
```



Compositional Equivalence: Directed Networks

Incubator A. Solid: Collaboration. Dotted: Friendship. Dashed: Competition



Positional Analysis: Directed Networks

- Compositional equivalence with directed networks performs better by including **relational contrast** in the modeling
  - ➟ This is operationalized through the *transpose* of the primitive ties

```
netAat <- zbind(netA, attA)
dimnames(netAat)[3][[1]]

[1] "C" "F" "K" "A" "B"
```

- Function `rbox` can generate tie transposes

- However, since actor attributes are represented by diagonal matrices, it does not make any sense to include the transposes in the modeling
  - ➟ we control the labeling of transposes through argument `tlbs`

```
rbox(netAat, transp = TRUE, tlbs = c("D","G","L",NA,NA))
```

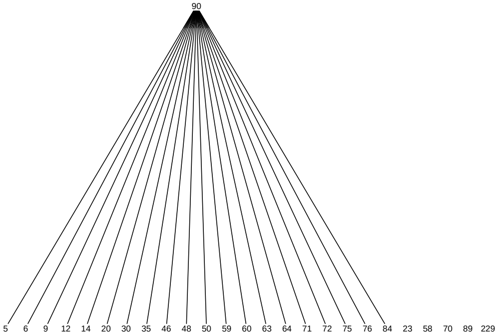
Cumulated Person Hierarchy

```
cph(rbox(netAat, transp = TRUE, tlbs = c("D","G","L",NA,NA), k = 2))


    5 6 9 12 14 20 23 30 35 46 48 50 58 59 60 63 64 70 71 72 75 76 84 89 90 229
5   1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
6   1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
9   1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
12  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
14  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
20  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
23  0 0 0  0  0  0  1  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0   0
30  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
35  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
46  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
48  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
50  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
58  0 0 0  0  0  0  1  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0   0
59  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
60  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
63  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
64  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
70  0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0   0
71  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
72  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
75  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
76  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
84  1 1 1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  0  1  1  1  1  1  0  1   0
89  0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0   0
90  0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1   0
229 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   1
...
```

Cumulated Person Hierarchy

Compositional equivalence with directed networks

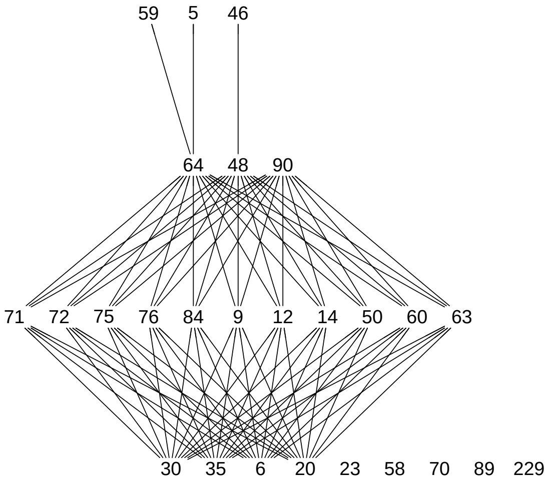
```
diagram(cph(rbox(netAat, transp = TRUE, tlbs = c("D","G","L",NA,NA), k = 2)))
```



☛ not an optimal structure

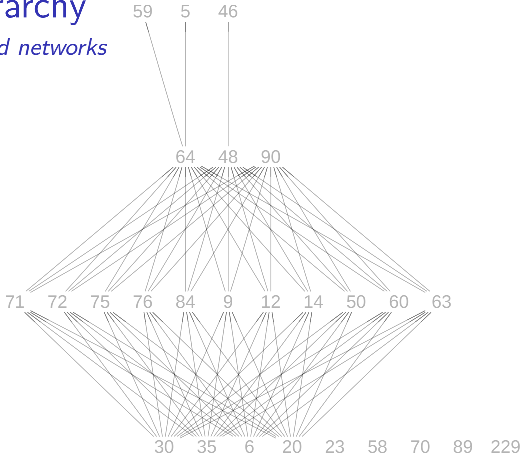
Cumulated Person Hierarchy

```
diagram(cph(rbox(netAat, transp = TRUE, tlbs = c("D","G","L",NA,NA), k = 3)))
```



Cumulated Person Hierarchy

Compositional equivalence, directed networks




```
as.table(rbind(dimnames(netAat)[1][[1]],
+     c(3,2,1,1,1,2,4,2,2,3,3,1,4,3,1,1,3,4,1,1,1,1,1,4,3,4)))

  A B C D  E  F  G  H  I  J  K  L  M  N  O  P  Q  R  S  T  U  V  W  X  Y  Z
A 5 6 9 12 14 20 23 30 35 46 48 50 58 59 60 63 64 70 71 72 75 76 84 89 90 229
B 3 2 1 1  1  2  4  2  2  3  3  1  4  3  1  1  3  4  1  1  1  1  1  4  3  4
```

Positional System for the Incubator A

```
netArb <-  rbox(netAat, transp = TRUE, tlbs = c("D","G","L",NA,NA), k = 3)
netAatps <-  reduc(netArb$W[ , , 1:8],
+     clu = c(3,2,1,1,1,2,4,2,2,3,3,1,4,3,1,1,3,4,1,1,1,1,1,4,3,4))
```

C

1	0	1	0
0	1	1	0
1	0	1	0
0	0	0	1

F

1	0	1	0
1	1	1	0
1	0	1	0
0	0	0	1

K

1	0	1	0
0	1	0	0
0	0	1	0
0	0	0	0

A

1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

D

1	0	1	0
0	1	0	0
1	1	1	0
0	0	0	1

G

1	1	1	0
0	1	0	0
1	1	1	0
0	0	0	1

L

1	0	0	0
0	1	0	0
1	0	1	0
0	0	0	0

B

1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Role Structure of Incubator A

Three **algebraic constraints** of the Role Structure:

```
# ROLE TABLE
semigroup(netAatps, type = "symbolic")
```

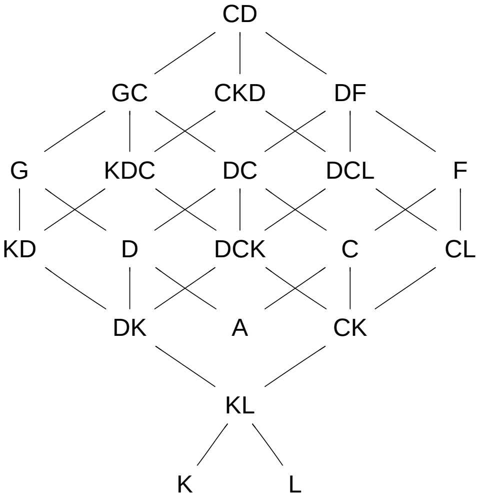
```
# SET OF EQUATIONS
netAatst <-  strings(netAatps, equat = TRUE, k = 3)
netAatst$equat
```

```
# SET OF INCLUSIONS
partial.order(netAatst, type = "strings")
```

Hasse Diagram with the Set of Inclusions

Role structure of Incubator A

```
diagram(partial.order(netAatst, type = "strings"))
```



Decomposition of Relational Structures

Subdirect representation

- An **aggregated** role structure is obtained by means of synthesis rules of the relational system
  ➟ synthesis rules can be a direct or a subdirect representation

- *Subdirect* representations imply finding **congruence** relations, which are correspondences that preserve the operation
  ➟ certain overlapping is required with this synthesis rule

Function `cngr` computes the congruence relations in semigroups

Function `decomp` performs the decomposition of relational structures

Decomposition of Relational Structures

```
# FIRST RECORD THE ROLE AND PARTIAL ORDER TABLES
netAatrt <-  semigroup(netAatps, type = "symbolic")
netAatpo <-  partial.order(netAatst, type = "strings")
```

```
## CONGRUENCES IN THE ABSTRACT SEMIGROUP
cngr(netAatrt)
```

```
# UNIQUE CONGRUENCES IN THE PARTIALLY ORDERED SEMIGROUP
netAatcg <-  cngr(S = netAatrt, PO = netAatpo, unique = TRUE)
```

```
# DECOMPOSITION OF ROLE TABLES BASED ON CONGRUENCE CLASSES
decomp(netAatrt, netAatcg, type = "cc")
```

```
# DECOMPOSITION WITH THE REDUCTION OPTION
decomp(netAatrt, netAatcg, type = "cc", reduc = TRUE)
```

Example: Decomposition of Role Structures Incubator A

```
netAatdc <- decomp(netAatrt, netAatcg, type = "cc", reduc = TRUE)
```

```
# ADDITIONAL SET OF EQUATIONS
netAatdc$eq[2]
```

```
[[1]]
[[1]][[1]]
 [1] C   F   CK  CD  CL  KD  DC  DF  DK  GC  CKD KDC DCK DCL

[[1]][[2]]
[1] KL
...
```

```
# IMAGE MATRIX
netAatdc$IM[2]
```

```
[[1]]
   C KL  K  A D  L
C  C  C  C  C C  C
KL C KL KL KL C KL
K  C KL  K  K C KL
A  C KL  K  A D  L
D  C  C  C  D D  C
L  C KL KL  L C  L
```

Decomposition of Relational Structures

- **Aggregated** role tables are homomorphic images of the role structure

- They provide the **logics** in the interlock of the role relations

  ➟ make a more transparent substantial interpretation

- It is possible to decompose the relational system by *induced inclusions* of the full factorization option from PACNET (Pattison et al, 2000)

  ➟ use function `pacnet` to import this outcome

- To compare relational structures are different strategies: the joint homomophic reduction or JNTHOM (Boorman & White, 1976), and the common structure semigroup or CSS (Bonacich, 1980)

  ➟ this is an extension that is out of scope

4. Signed networks

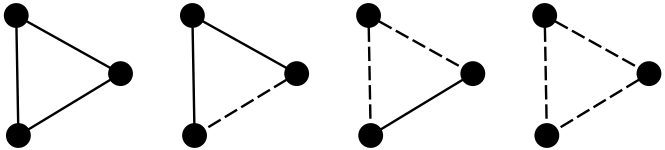
Structural Balance

- Simmel (1950) studied "conflict as a mechanism for integration" in triadic relations
- Heider (1958) developed the **Structural Balance** theory as a special cases of transitivity
- Structural Balance theory applies to networks to see whether the system has an inherent equilibrium or not

"all positive ties within groups; all negative ties between groups"

Structural Balance

- A balanced structure is represented by a **signed network**
  ➟ a special case of a multiple network



- Paths in signed graphs are positive when they have an even number of negative edges; otherwise negative

☛ **extension**: a path/semipath is ambivalent iff contains at least one ambivalent edge

Structures in Balance Theory

balanced → **clusterable** → '*weak*' clusterable

(Cartwright & Harary, 1956) (Davis, 1967)

$\circ$	p	n
p	p	n
n	n	p

Classical

$\circ$	p	n	a
p	p	n	a
n	n	a	a
a	a	a	a

Extended

p → positive n → negative a → ambivalent

Semiring

Algebraic structure

A **semiring** is an object set endowed with a pair operations, multiplication and addition, together with two neutral elements:

$$\langle\, Q,\ +,\ \cdot\ ,\ 0,\ 1\,\rangle$$

properties:

- closed, associative, and commutative under addition
- multiplication distributes over addition, i.e. for all $\mathsf{p}, \mathsf{n}, \mathsf{a} \in Q$:

$$\mathsf{p}\ \cdot\ (\mathsf{n} + \mathsf{a}) = (\mathsf{p}\ \cdot\ \mathsf{n}) + (\mathsf{p}\ \cdot\ \mathsf{a}) \quad \text{and} \quad (\mathsf{p} + \mathsf{n})\ \cdot\ \mathsf{a} = (\mathsf{p}\ \cdot\ \mathsf{a}) + (\mathsf{n}\ \cdot\ \mathsf{a})$$

☛ *Semirings help us to evaluate the relational system in terms of balance theory by looking at paths and semipaths*

Semiring Operations

$\cdot$	o	n	p	a
o	o	o	o	o
n	o	p	n	a
p	o	n	p	a
a	o	a	a	a

$+$	o	n	p	a
o	o	n	p	a
n	n	n	a	a
p	p	a	p	a
a	a	a	a	a

Balance

$\cdot$	o	n	p	a	q
o	o	o	o	o	o
n	o	q	n	n	q
p	o	n	p	a	q
a	o	n	a	a	q
q	o	q	q	q	q

$+$	o	n	p	a	q
o	o	n	p	a	q
n	n	n	a	a	n
p	p	a	p	a	p
a	a	a	a	a	a
q	q	n	p	a	q

Clustering

Semiring function

```
# ARGUMENTS IN FUNCTION semiring()
formals("semiring")


$x


$type
c("balance", "cluster")

$symclos
[1] TRUE

$transclos
[1] TRUE

$k
[1] 2

$lbs
```

Balanced Structures

Example as in Doreian, et al (2005)

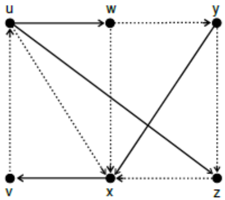


Figure 10.3. An example from Roberts.

Table 10.4. The Value Matrix and Its Closure for Roberts's Example

	u	v	w	x	y	z
u	0	0	p	n	0	p
v	n	0	0	0	0	0
w	0	0	0	n	n	0
x	0	p	0	0	0	0
y	0	0	0	p	0	n
z	0	0	0	n	0	0

	u	v	w	x	y	z
u	**p**	n	**p**	n	n	**p**
v	n	**p**	n	**p**	**p**	n
w	**p**	n	**p**	n	n	**p**
x	n	**p**	n	**p**	**p**	n
y	n	**p**	n	**p**	**p**	n
z	**p**	n	**p**	n	n	**p**

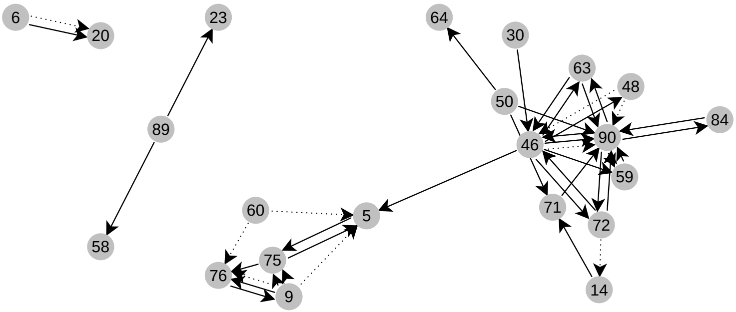
Balance Semiring

```
# CREATE MATRIX DATA TYPE
mat <- matrix(nrow=6, ncol=6)
rownames(mat) <- letters[21:26]
colnames(mat) <- rownames(mat)

# ASSING VALUES
mat[1,] <- c(0,0,1,-1,0,1)
mat[2,] <- c(-1,0,0,0,0,0)
mat[3,] <- c(0,0,0,-1,-1,0)
mat[4,] <- c(0,1,0,0,0,0)
mat[5,] <- c(0,0,0,1,0,-1)
mat[6,] <- c(0,0,0,-1,0,0)


   u  v  w  x  y  z
u  0  0  1 -1  0  1
v -1  0  0  0  0  0
w  0  0  0 -1 -1  0
x  0  1  0  0  0  0
y  0  0  0  1  0 -1
z  0  0  0 -1  0  0
```

```
# BALANCE SEMIRING STRUCTURE
semiring(as.signed(mat), type="balance")


$val
[1] 1  0  -1

$s
   1 2 3  4  5  6
1  0 0 1 -1  0  1
2 -1 0 0  0  0  0
3  0 0 0 -1 -1  0
4  0 1 0  0  0  0
5  0 0 0  1  0 -1
6  0 0 0 -1  0  0

$Q
  1 2 3 4 5 6
1 p n p n n p
2 n p n p p n
3 p n p n n p
4 n p n p p n
5 n p n p p n
6 p n p n n p

$k
[1] 2

attr(,"class")
[1] "Rel.Q"   "balance"
```

Incubator network

```
# COOPERATION AND COMPETITION TIES IN 'netA' WITHOUT ISOLATES
netAck <-  rm.isol(netA[ , , c(1,3)])

# PLOT THE MULTIGRAPH BY REUSING THE OUTLINE
multigraph(netAck, outline = otlA, lty = c(1,3), layout = "force", seed = 9)
```



Signed Network *C and K in Incubator A*

```
# FUNCTION signed() CREATES A "Signed" CLASS OBJECT FROM 2 MATRICES
netAsg <-  signed(netAck)
```

```
$val
[1] p o n a

$s
   5 6 9 14 20 23 30 46 48 50 58 59 60 63 64 71 72 75 76 84 89 90
5  o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  p  o  o  o  o
6  o o o  o  a  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o
9  n o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  a  a  o  o  o
14 o o o  o  o  o  o  o  o  o  o  o  o  o  o  p  o  o  o  o  o  o
20 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o
23 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o
30 o o o  o  o  o  o  p  o  o  o  o  o  o  o  o  o  o  o  o  o  o
46 p o o  o  o  o  o  o  p  o  o  p  o  p  o  o  p  o  o  o  o  a
48 o o o  o  o  o  o  n  o  o  o  o  o  o  o  o  o  o  o  o  o  n
50 o o o  o  o  o  o  o  o  o  o  o  o  o  p  p  o  o  o  o  o  p
58 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o
59 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  p
60 n o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  n  o  o  o
63 o o o  o  o  o  o  p  o  o  o  o  o  o  o  o  o  o  o  o  o  p
64 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o
71 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  p
72 o o o  n  o  o  o  p  o  o  o  o  o  o  o  o  o  o  o  o  o  p
75 p o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  p  o  o  o
76 o o p  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o
84 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  p
89 o o o  o  o  p  o  o  o  o  p  o  o  o  o  o  o  o  o  o  o  o
90 o o o  o  o  o  o  p  o  o  o  p  o  p  o  o  p  o  o  p  o  o
```

Semiring structures

```
# BALANCE SEMIRING 2—PATHS (DEAFULT)
semiring(netAsg, type = "balance")

# 3—PATHS
semiring(netAsg, type = "balance", k = 3)

# 2—SEMIPATHS
semiring(netAsg, type = "balance", symclos = FALSE)
# ...
```

```
# CLUSTER SEMIRING 2—PATHS (DEAFULT)
semiring(netAsg, type = "cluster")

# 3—PATHS
semiring(netAsg, type = "cluster", k = 3)

# 2—SEMIPATHS
semiring(netAsg, type = "cluster", symclos = FALSE)
# ...
```

Checking for Balance

```
identical(
+   semiring(netAsg, type = "balance", k = 3)$Q,
+   semiring(netAsg, type = "balance", k = 2)$Q )

[1] FALSE
```

```
identical(
+   semiring(netAsg, type = "balance", k = 3)$Q,
+   semiring(netAsg, type = "balance", k = 4)$Q )

[1] FALSE
```

```
identical(
+   semiring(netAsg, type = "balance", k = 4)$Q,
+   semiring(netAsg, type = "balance", k = 5)$Q )

[1] TRUE
```

Checking for Balance (Cluster)

```
identical(
+   semiring(netAsg, type = "cluster", k = 3)$Q,
+   semiring(netAsg, type = "cluster", k = 2)$Q )

[1] FALSE
```

```
identical(
+   semiring(netAsg, type = "cluster", k = 3)$Q,
+   semiring(netAsg, type = "cluster", k = 4)$Q )

[1] FALSE
```

```
identical(
+   semiring(netAsg, type = "cluster", k = 4)$Q,
+   semiring(netAsg, type = "cluster", k = 5)$Q )

[1] TRUE
```

Weak Balance Structure

```r
# BALANCE WITH SEMIPATHS
netAQb <-  semiring(netAsg, type = "balance", k = 4)

perm(netAQb$Q, clu = c(1,6,1,2,6,6,3,2,2,3,6,2,4,2,5,2,2,1,1,2,6,2))
```

```
   5 9 75 76 14 46 48 59 63 71 72 84 90 30 50 60 64 6 20 23 58 89
5  a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
9  a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
75 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
76 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
14 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
46 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
48 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
59 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
63 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
71 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
72 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
84 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
90 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
30 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
50 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
60 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
64 a a a  a  a  a  a  a  a  a  a  a  a  a  a  a  a  o o  o  o  o
6  o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  a o  o  o  o
20 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o a  o  o  o
23 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  p  p  o
58 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  p  p  o
89 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  p
```

Weak Balance Structure

```
# BALANCE WITH PATHS
netAQbp <-  semiring(netAsg, type = "balance", symclos = FALSE, k = 4)

perm(netAQbp$Q, clu = c(1,6,1,2,6,6,3,2,2,3,6,2,4,2,5,2,2,1,1,2,6,2))

   5 9 75 76 14 46 48 59 63 71 72 84 90 30 50 60 64 6 20 23 58 89
5  a a a  a  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
9  a a a  a  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
75 a a a  a  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
76 a a a  a  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
14 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
46 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
48 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
59 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
63 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
71 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
72 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
84 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
90 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
30 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
50 a a a  a  a  a  a  a  a  a  a  a  a  o  o  o  o  o o  o  o  o
60 a a a  a  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
64 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
6  o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
20 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
23 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
58 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
89 o o o  o  o  o  o  o  o  o  o  o  o  o  o  o  o  o o  o  o  o
```

Weak Balance Structure

Main component of Incubator A

```
# FUNCTION comps() FINDS COMPONENTS AND ISOLATES
comps(netAck)
```

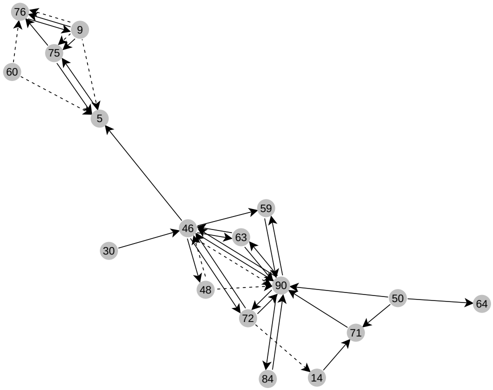
```
$com
$com[[1]]
 [1] "5"  "50" "59" "60" "63" "64" "71" "72" "75" "76" "84" "90" "9"  "14" "30" "46" "48"

$com[[2]]
[1] "58" "89" "23"

$com[[3]]
[1] "6"  "20"


$isol
character(0)
```

```
# RECORD TIES FROM MAIN COMPONENT OF netAck
netAc1 <-  rel.sys(incubA$net, "toarray", sel = comps(netAck)$com[[1]])
```

Weak Balance Structure

Main component of Incubator A

```r
# PLOT NETWORK RELATIONS 'C' AND 'K' IN THE COMPONENT
multigraph(netAc1[,,c(1,3)], outline = otlA, layout = "force", seed = 6)
```



Weak Balance Structure

Outline

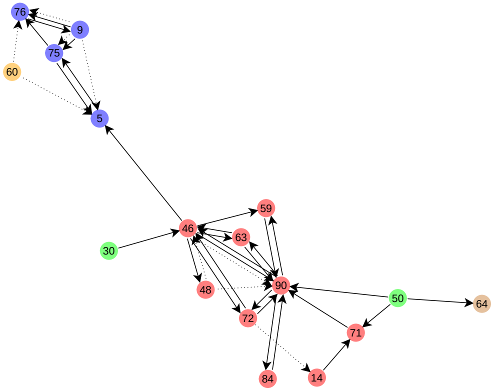
```
# MAKE OUTLINE WITH INFO FROM WEAK BALANCE STRUCTURE OF PATHS
otlAck <-  list(lty = c(1,3), clu = c(1,1,2,3,2,2,3,2,4,2,5,2,2,1,1,2,2),
+    vcol = c("blue","red","green","orange","peru"), alpha = .5)
```

```
c(otlAck,otlA)


$lty
[1] 1 3

$clu
 [1] 1 1 2 3 2 2 3 2 4 2 5 2 2 1 1 2 2

$vcol
[1] "blue"   "red"    "green"  "orange" "peru"

$alpha
[1] 0.5

$ecol
[1] 1

$vcol
[1] "#C0C0C0"

...
```

Weak Balance Structure

Main component of Incubator A

```
# ARGUMENT NAMES CAN BE OMITTED OUTSIDE outline
multigraph(netAc1[,,c(1,3)], outline = c(otlA, otlAck), "force", seed = 6)
```

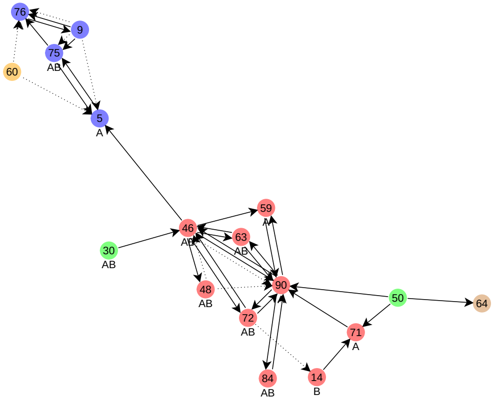


Otherwise `outline = c(otlA, otlAck, layout="force")`

Weak Balance Structure

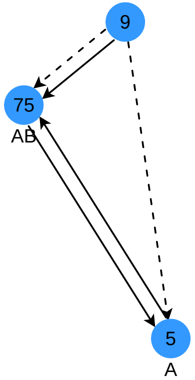
for social influence through comparison

```
multigraph(netAc1[,,c(1,3)], att = netAc1[,,4:5], layout = "force", seed = 6,
+   outline = c(otlA, otlAck))
```



Balance semiring *(Signed triad)*

```
# 2—Paths  (9, 75)
"n, p" "o, a" "a, o"

# multiplication
"n" "o" "o"

# addition
n
```



	5	9	75
5	o	o	p
9	n	o	a
75	p	o	o

t^{α}

	5	9	75
5	p	o	o
9	a	o	n
75	o	o	p

t^{α} paths, $k > 1$

	5	9	75
5	p	a	a
9	a	a	n
75	a	n	a

t^{α} semipaths, $k = 2$

	5	9	75
5	a	a	a
9	a	a	a
75	a	a	a

t^{α} semipaths, $k > 2$

$5.$ Two-mode networks

Two-mode networks

- Ties between two sets of entities represent two-mode, bipartite, or **affiliations networks**

  ➟ like the duality between *"people and groups"*, *"person and events"*, *"actors and their attributes"*

- In a 2-mode matrix data the domain and the codomain are not equal

  ➟ serves to represent affiliations networks

- An algebraic approach to affiliation networks is found in **Formal Concept Analysis**

Formal Concept Analysis

(Ganter & Wille, 1996)

- Formal Concept Analysis is an analytical framework for the study of affiliation networks

- Elements in the domain and codomain are called *Objects* and *Attributes* resp.

- A set of Objects G and a set of Attributes M are associated with an incident relation $I \subseteq G \times M$ in a **formal context**

- The **formal concept** of a formal context is a pair of sets of maximally contained objects A and attributes B

  ➟ (i.e. maximal rectangles in the formal context)

A and B are said to be the *extent* and *intent* of the formal concept

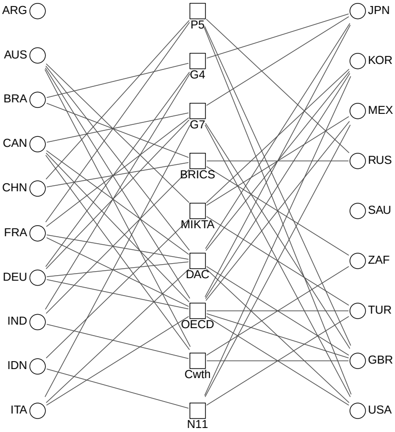
G20 countries affiliation network

```
# OBJECT G20 IS A DATA FRAME THAT REPRESENTS A FORMAL CONTEXT
G20
```

```
    P5 G4 G7 BRICS MIKTA DAC OECD Cwth N11
ARG  0  0  0     0     0   0    0    0   0
AUS  0  0  0     0     1   1    1    1   0
BRA  0  1  0     1     0   0    0    0   0
CAN  0  0  1     0     0   1    1    1   0
CHN  1  0  0     1     0   0    0    0   0
FRA  1  0  1     0     0   1    1    0   0
DEU  0  1  1     0     0   1    1    0   0
IND  0  1  0     1     0   0    0    1   0
IDN  0  0  0     0     1   0    0    0   1
ITA  0  0  1     0     0   1    1    0   0
JPN  0  1  1     0     0   1    1    0   0
KOR  0  0  0     0     1   1    1    0   1
MEX  0  0  0     0     1   0    1    0   1
RUS  1  0  0     1     0   0    0    0   0
SAU  0  0  0     0     0   0    0    0   0
ZAF  0  0  0     1     0   0    0    1   0
TUR  0  0  0     0     1   0    1    0   1
GBR  1  0  1     0     0   1    1    1   0
USA  1  0  1     0     0   1    1    0   0
```

G20 countries affiliation network

```r
# BIPARTITE GRAPH WITH THREE COLUMNS
bmgraph(G20, layout = "bip3", cex = 3, tcex = 1)
```



Galois Derivations

- A **Galois derivation** between G and M is defined for any subsets $A \subseteq G$ and $B \subseteq M$ by

$$\begin{aligned} A' &= m \in M \mid (g, m) \in I \quad \text{(for all } g \in A) \\ B' &= g \in G \mid (g, m) \in I \quad \text{(for all } m \in B) \end{aligned}$$

– A' is the set of attributes common to all the objects in the intent

– B' the set of objects possessing the attributes in the extent

```
formals("galois")

$x


$labeling
c("full", "reduced")
```

Galois derivations in G20

```
galois(G20)


$P5
[1] "CHN, FRA, GBR, RUS, USA"

$G4
[1] "BRA, DEU, IND, JPN"

$`DAC, G7, OECD`
[1] "CAN, DEU, FRA, GBR, ITA, JPN, USA"

$BRICS
[1] "BRA, CHN, IND, RUS, ZAF"

$MIKTA
[1] "AUS, IDN, KOR, MEX, TUR"

$`DAC, OECD`
[1] "AUS, CAN, DEU, FRA, GBR, ITA, JPN, KOR, USA"

$OECD
[1] "AUS, CAN, DEU, FRA, GBR, ITA, JPN, KOR, MEX, TUR, USA"

$Cwth
[1] "AUS, CAN, GBR, IND, ZAF"

$`MIKTA, N11`
[1] "IDN, KOR, MEX, TUR"

$`BRICS, Cwth, DAC, G4, G7, MIKTA, N11, OECD, P5`
character(0)

...
```

Galois derivations in G20 – Reduced labeling

```
g20gc <- galois(G20, labeling = "reduced")
```

```
$reduc
$reduc$P5
character(0)

$reduc$G4
character(0)

$reduc$G7
[1] "ITA"

$reduc$BRICS
character(0)

$reduc$MIKTA
character(0)

$reduc$DAC
character(0)

$reduc$OECD
character(0)

$reduc$Cwth
character(0)
```

```
$reduc$N11
[1] "IDN"

$reduc[[10]]
character(0)

$reduc[[11]]
[1] "FRA, USA"

$reduc[[12]]
[1] "CHN, RUS"

$reduc[[13]]
[1] "GBR"

$reduc[[14]]
[1] "DEU, JPN"

$reduc[[15]]
[1] "BRA"

$reduc[[16]]
[1] "IND"

$reduc[[17]]
[1] "CAN"
```

```
$reduc[[18]]
[1] "ZAF"

$reduc[[19]]
[1] ""

$reduc[[20]]
character(0)

$reduc[[21]]
[1] "AUS"

$reduc[[22]]
character(0)

$reduc[[23]]
[1] "KOR"

$reduc[[24]]
[1] "MEX, TUR"

$reduc[[25]]
[1] "ARG, SAU"
```

Partial ordering of the Concepts

A *hierarchy* of concepts is given by the sub–superconcept relation

$$(A, B) \leq (A_2, B_2) \quad \Leftrightarrow \quad A_1 \subseteq A_2 \qquad (\Leftrightarrow \quad B_1 \subseteq B_2)$$

Concept Lattice of the Context

- built from the hierarchy structure of concepts
- The greatest lower bound of the meet and the least upper bound of the join are defined for an index set T as

$$\bigwedge_{t \in T} \; (A_t, B_t) \; = \; \Big(\bigcap_{t \in T} A_t, \; (\bigcup_{t \in T} B_t)'' \Big)$$

$$\bigvee_{t \in T} \; (A_t, B_t) \; = \; \Big((\bigcup_{t \in T} A_t)'', \; \bigcap_{t \in T} B_t \Big)$$

Partial order of concepts

```
# FUNCTION partial.order() CONSTRUCTS HIERARCHY OF CONCEPTS
g20gcpo <-  partial.order(g20gc, type = "galois")

                {P5} {} {G4} {} {G7} {ITA} {BRICS} {} {MIKTA} {} {DAC} {} {OECD} {} {Cwth} {}
{P5} {}               1       0            0          0           0        0         0         0
{G4} {}               0       1            0          0           0        0         0         0
{G7} {ITA}            0       0            1          0           0        1         1         0
{BRICS} {}            0       0            0          1           0        0         0         0
{MIKTA} {}            0       0            0          0           1        0         0         0
{DAC} {}              0       0            0          0           0        1         1         0
{OECD} {}             0       0            0          0           0        0         1         0
{Cwth} {}             0       0            0          0           0        0         0         1
{N11} {IDN}           0       0            0          0           1        0         0         0
10                    1       1            1          1           1        1         1         1
{} {FRA, USA}         1       0            1          0           0        1         1         0
{} {CHN, RUS}         1       0            0          1           0        0         0         0
{} {GBR}              1       0            1          0           0        1         1         1
{} {DEU, JPN}         0       1            1          0           0        1         1         0
{} {BRA}              0       1            0          1           0        0         0         0
{} {IND}              0       1            0          1           0        0         0         1
{} {CAN}              0       0            1          0           0        1         1         1
{} {ZAF}              0       0            0          1           0        0         0         1
19                    0       0            0          0           1        1         1         0
20                    0       0            0          0           1        0         1         0
{} {AUS}              0       0            0          0           1        1         1         1
22                    0       0            0          0           0        1         1         1
{} {KOR}              0       0            0          0           1        1         1         0
{} {MEX, TUR}         0       0            0          0           1        0         1         0
{} {ARG, SAU}         0       0            0          0           0        0         0         0
```

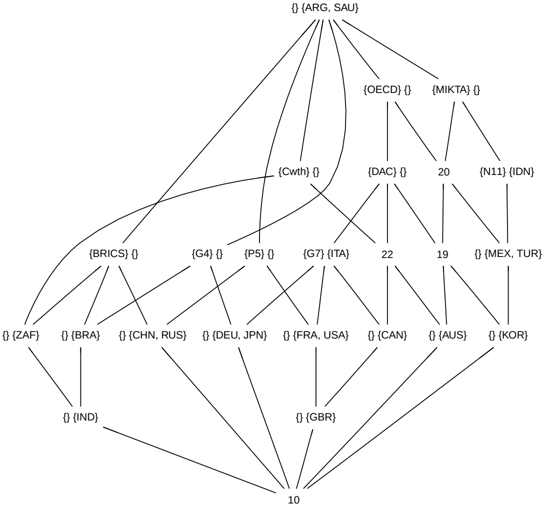
(Extract)

Galois derivations and partial ordering

```
# STRUCTURE OF g20gc OBJECT CREATED WITH A REDUCED LABELING
str(g20gc)

List of 2
 $ full :List of 25
  ..$ P5                                    : chr "CHN, FRA, GBR, RUS, USA"
  ..$ G4                                    : chr "BRA, DEU, IND, JPN"
  ..$ DAC, G7, OECD                         : chr "CAN, DEU, FRA, GBR, ITA, JPN, USA"
  ..$ BRICS                                 : chr "BRA, CHN, IND, RUS, ZAF"
  ..$ MIKTA                                 : chr "AUS, IDN, KOR, MEX, TUR"
  ..$ DAC, OECD                             : chr "AUS, CAN, DEU, FRA, GBR, ITA, JPN,
  ..$ OECD                                  : chr "AUS, CAN, DEU, FRA, GBR, ITA, JPN,
  ..$ Cwth                                  : chr "AUS, CAN, GBR, IND, ZAF"
  ..$ MIKTA, N11                            : chr "IDN, KOR, MEX, TUR"
  ..$ BRICS, Cwth, DAC, G4, G7, MIKTA, N11, OECD, P5: chr(0)
...

..- attr(*, "class")= chr [1:2] "Galois" "full"
 $ reduc:List of 25
  ..$ P5    : chr(0)
  ..$ G4    : chr(0)
  ..$ G7    : chr "ITA"
  ..$ BRICS: chr(0)
  ..$ MIKTA: chr(0)
  ..$ DAC   : chr(0)
  ..$ OECD  : chr(0)
  ..$ Cwth  : chr(0)
  ..$ N11   : chr "IDN"
  ..$       : chr(0)
...
```

Concept lattice of the context

```r
# PLOT HIERARCHY OF CONCEPTS AS LATTICE DIAGRAM
diagram(g20gcpo)
```



Set of inclusions in the partial ordering

```
# FUNCTION transf() TRANSFORMS MATRIX TO LIST OF PAIRWISE RELATIONS
transf(g20gcpo, type = "tolist", lb2lb = TRUE)


  [1] "{} {ARG, SAU}, {} {ARG, SAU}" "{} {AUS}, {} {ARG, SAU}"
  [3] "{} {AUS}, {} {AUS}"           "{} {AUS}, {Cwth} {}"
  [5] "{} {AUS}, {DAC} {}"           "{} {AUS}, {MIKTA} {}"
  [7] "{} {AUS}, {OECD} {}"          "{} {AUS}, 19"
  [9] "{} {AUS}, 20"                 "{} {AUS}, 22"
 [11] "{} {BRA}, {} {ARG, SAU}"      "{} {BRA}, {} {BRA}"
 [13] "{} {BRA}, {BRICS} {}"         "{} {BRA}, {G4} {}"
 [15] "{} {CAN}, {} {ARG, SAU}"      "{} {CAN}, {} {CAN}"
 [17] "{} {CAN}, {Cwth} {}"          "{} {CAN}, {DAC} {}"
 [19] "{} {CAN}, {G7} {ITA}"         "{} {CAN}, {OECD} {}"
 [21] "{} {CAN}, 22"                 "{} {CHN, RUS}, {} {ARG, SAU}"
 [23] "{} {CHN, RUS}, {} {CHN, RUS}" "{} {CHN, RUS}, {BRICS} {}"
 [25] "{} {CHN, RUS}, {P5} {}"       "{} {DEU, JPN}, {} {ARG, SAU}"
 [27] "{} {DEU, JPN}, {} {DEU, JPN}" "{} {DEU, JPN}, {DAC} {}"
 [29] "{} {DEU, JPN}, {G4} {}"       "{} {DEU, JPN}, {G7} {ITA}"
 [31] "{} {DEU, JPN}, {OECD} {}"     "{} {FRA, USA}, {} {ARG, SAU}"
 [33] "{} {FRA, USA}, {} {FRA, USA}" "{} {FRA, USA}, {DAC} {}"
 [35] "{} {FRA, USA}, {G7} {ITA}"    "{} {FRA, USA}, {OECD} {}"
 [37] "{} {FRA, USA}, {P5} {}"       "{} {GBR}, {} {ARG, SAU}"
 [39] "{} {GBR}, {} {CAN}"           "{} {GBR}, {} {FRA, USA}"
 [41] "{} {GBR}, {} {GBR}"           "{} {GBR}, {Cwth} {}"
 [43] "{} {GBR}, {DAC} {}"           "{} {GBR}, {G7} {ITA}"
 [45] "{} {GBR}, {OECD} {}"          "{} {GBR}, {P5} {}"
 [47] "{} {GBR}, 22"                 "{} {IND}, {} {ARG, SAU}"

...

[135] "22, 22"
```

Filters and Ideals

formal definition

- Let $(P, \leq)$ be an ordered set, and a, b are elements in P
- A non-empty subset U [resp. D] of P is an upset [resp. downset] called a **filter** [resp. **ideal**] if, for all $a \in P$ and $b \in U$ [resp. D]

$$b \leq a \quad \text{implies} \quad a \in U \qquad\qquad [\text{ resp. } a \leq b \quad \text{implies} \quad a \in D\ ]$$

- The upset $\uparrow x$ formed for all the upper bounds of $x \in P$ is called a **principal filter** generated by x
- Dually, $\downarrow x$ is a **principal ideal** with all the lower bounds of $x \in P$

☛ filters and ideals not coinciding with P are called *proper*

Filters and Ideals

```
# fltr() FINDS PRINCIPAL FILTERS IN THE PARTIAL ORDER OF THE CONTEXT
formals("fltr")


$x


$PO


$rclos
[1] TRUE

$ideal
[1] FALSE
```

Principal Filters

```
# PRINCIPAL FILTER OF THE FIRST CONCEPT IN g20gcpo
fltr(1, g20gcpo)
```

```
$`1`
[1] "{P5} {}"

$`25`
[1] "{} {ARG, SAU}"
```

```
# ANOTHER OPTION IS TO USE INTENT LABELS OF DIFFERENT CONCEPTS
fltr(c("P5", "BRICS"), g20gcpo)
```

```
$`1`
[1] "{P5} {}"

$`4`
[1] "{BRICS} {}"

$`25`
[1] "{} {ARG, SAU}"
```

Principal Ideals

```
# PRINCIPAL IDEAL OF THE FIRST CONCEPT IN g20gcpo
fltr("P5", g20gcpo, ideal = TRUE)

$`1`
[1] "{P5} {}"

$`10`
[1] "10"

$`11`
[1] "{} {FRA, USA}"

$`12`
[1] "{} {CHN, RUS}"

$`13`
[1] "{} {GBR}"
```

☛ Beware that *ideals* in groups and semigroups have a different meaning

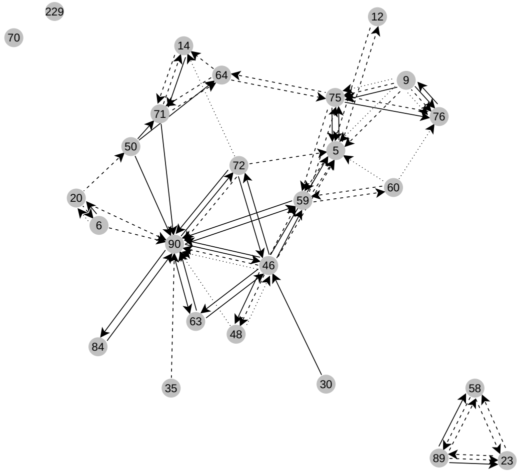
6. Miscellaneous

Costumized coordinated system

```
# FUNCTION frcd() PROVIDES COORDINATES FOR THE FORCE DIRECTED LAYOUT
frcd(netA, seed = 1)


            V1          V2
5   0.65267103 -0.28223505
6   0.17257937 -0.43406002
9   0.79536079 -0.13969918
12  0.73711345 -0.01060475
14  0.34446352 -0.06955628
20  0.12662053 -0.37858449
23  1.00000000 -0.91130240
30  0.63293893 -0.75622763
35  0.31886852 -0.76427853
46  0.51657424 -0.51546610
48  0.45059885 -0.65494211
50  0.23722657 -0.27412538
58  0.93472721 -0.76427853
59  0.58590570 -0.38352210
60  0.76960028 -0.35724634
63  0.36873411 -0.62902125
64  0.42137241 -0.12903575
70  0.00000000 -0.05321605
71  0.29624989 -0.20766673
72  0.45630701 -0.31248805
75  0.65140506 -0.17440605
76  0.86214500 -0.21322180
84  0.17061150 -0.68040030
89  0.86214500 -0.90619560
90  0.32575712 -0.47210522
229 0.08260838  0.00000000
```

Costumized coordinated system

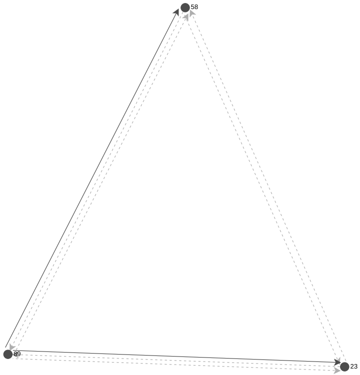
```
# THESE OPTIONS ARE THEN EQUIVALENT
multigraph(netA, outline = otlA, layout = "force", seed = 1)
multigraph(netA, outline = otlA, coord = frcd(netA, seed = 1))
```



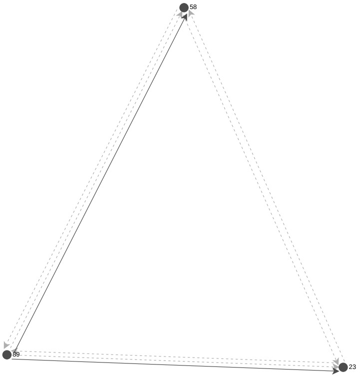
Costumized coordinated system

```
# RECORD DATA TRIADIC CONFIGURATION of netA
cdsA <-  frcd(netA, seed = 1)[c(7,13,24), ]
netAc2 <-  rel.sys(netA, type = "toarray", sel = comps(netA)$com[[2]])
```

```
multigraph(netAc2, coord=cdsA)
```

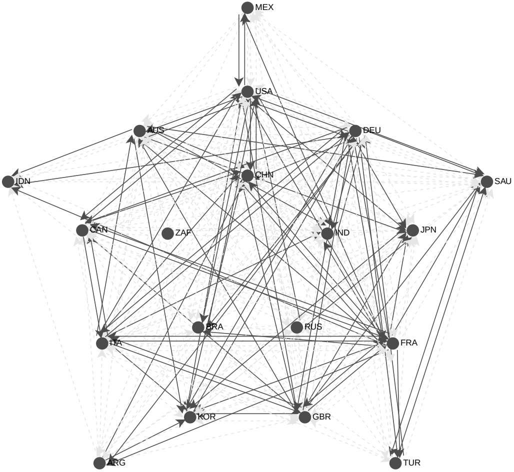



```
multigraph(netAc2, coord=cdsA, swp=TRUE)
```



Costumized coordinated system

```
# SIMILAR EQUIVALENCE WITH THE CONCENTRIC LAYOUT...
multigraph(fmlkhny2015, layout = "conc", nr = cls)
multigraph(fmlkhny2015, coord = conc(fmlkhny2015, nr = cls))
```



Decomposition with meet complements

Recall...

```
# PARTIAL ORDER TABLE
netAatpo <- partial.order(netAatst, type = "strings")
```

```
## CONGRUENCES IN THE ABSTRACT SEMIGROUP
cngr(netAatrt)
```

```
# UNIQUE CONGRUENCES IN THE PARTIALLY ORDERED SEMIGROUP
netAatcg <- cngr(S = netAatrt, PO = netAatpo, unique = TRUE)
```

```
# DECOMPOSITION OF ROLE TABLES BASED ON CONGRUENCE CLASSES
decomp(netAatrt, netAatcg, type = "cc")
```

```
# DECOMPOSITION WITH THE REDUCTION OPTION
decomp(netAatrt, netAatcg, type = "cc", reduc = TRUE)
```

Decomposition with meet complements

```
# FUNCTION pacnet() READS OUTPUT OF PACNET PROGRAM (FULL FACTORIZATION OPTION)
pacnet(file = "https://github.com/mplex/sunbelt2017/raw/master/data/netAdecmp.out")
```

```
# OR WITH LOCAL DIRECTORY
# setwd("C:/sunbelt")
netApac <- pacnet(file = "data/netAdecmp.out")
```

```
# THEN PI—RELATIONS ARE CREATED WITH THE PARTIAL ORDER STRUCTURE
netAatpi <- pi.rels(netApac, netAatpo)
```

Decomposition with meet complements

```
# DECOMPOSITION WITH PI RELATIONS
decomp(netAatrt, netAatpi, type = "pi", reduc = TRUE)$clu

[[1]]
 [1]  1  1  2  3  4  5  6  7  8  7  9 10 11 11 12  8 13 13 14 14
 ...

 [[104]]
 [1] 1 2 3 3 4 4 4 1 2 2 4 4 1 2 4 1 2 1 1 2
```

```
# DECOMPOSITION WITH MEET COMPLEMENTS
decomp(netAatrt, netAatpi, type = "mc", reduc = TRUE)$clu

[[1]]
 [1] 1 1 1 2 3 4 2 1 4 1 4 1 3 3 3 4 4 4 3 3

[[2]]
 [1] 1 2 3 3 4 4 4 1 2 2 4 4 1 2 4 1 2 1 1 2

[[3]]
 [1] 1 1 2 1 1 1 2 2 1 2 2 2 1 1 2 1 2 2 2 2
```

Finally, some terminology...

- **Multiplex** $\leftrightarrow$ **Monoplex** *structure*

  ➟ System with several $\leftrightarrow$ single or collapsed levels in the set of relations

- **multiplex** $\leftrightarrow$ **uniplex** *edge*

  ➟ A relationship with multiple $\leftrightarrow$ single or collapsed levels

- **Multimodal** network

  ➟ Same as Multiplex, but most used with flows or transportation modes

- **Multilevel** network

  ➟ A structure with individual and group levels; i.e. affiliation networks, but *where both level entities are interrelated*

- **Multilayer** network

  ➟ Cascade structure with multiple subsystems and layers of connectivity

Multilevel...

TBD

References

- Pattison, P. *Algebraic Models for Social Networks*. Cambridge Univ. Press. 1993
- Doreian, P., V. Batagelj and A. Ferligoj *Generalized Blockmodeling*. Cambridge Univ. Press, 2004
- Harary, F, Z. Norman, and D. Cartwright *Structural Models: An Introduction to the Theory of Directed Graphs*. John Wiley & Sons. 1965.
- Ganter, B. and R. Wille *Formal Concept Analysis – Mathematical Foundations*. Springer. 1996
- Ostoic, J.A.R. "Creating context for social influence processes in multiplex networks." *Network Science*, 5(1), 1-29.

- R Development Core Team, *R: A language and environment for statistical computing*, v 3.3.1
- Gentry, J, et al `Rgraphviz`*: Provides plotting capabilities for R graph objects*. R pkg v 2.12.2
- Ostoic, J.A.R. `multiplex`*: Algebraic Tools for the Analysis of Multiple Social Networks*. R pkg v 2.6
- Ostoic, J.A.R. `multigraph`*: Plot and Manipulate Multigraphs*. R pkg devel v 0.60 (beta)

謝謝

http://CRAN.R-Project.org/package=multiplex

CRAN in views: Psychometrics, SocialSciences

http://CRAN.R-Project.org/package=multigraph

Package ‘multiplex’

August 28, 2013

Type Package

Title Analysis of Multiple Social Networks with Algebra

Version 1.0

Depends R (>= 3.0.1)

Date 2013-08-28

Author J. Antonio Rivero Ostoic

Maintainer Antonio Rivero O…

Description multiplex - Analysis of Multiple Social Networks with Algebra is a package for the study of social systems made of different types of relationships. It is possible to create and manipulate multivariate network data with different formats, and there are … treat multiple networks with routines that combine algebraic systems like the partially ordered semigroup or the semiring structure together with the relational bundles occurring in different types of multivariate network data sets.

License GPL-3

Suggests Rgraphviz

Encoding latin1

Collate
’as.semigroup.R’ ’as.strings.R’ ’bundle.census.R’ ’bundles.R’’cngr.R’ ’convert.R’ ’cph.R’

NeedsCompilation no

Repository CRAN

Date/Publication 2013-08-28 13:53:11

R topics documented: