Algebraic analyses and visualization of complex networks

· online workshop ·

Antonio Rivero Ostoic
CESU, University of San Simón

Sunbelt XLIII Conference ☆ 23 June 2023

Analysis of complex networks with algebra

Agenda

1. Visualization of multigraphs
   - ➟ Example 1a: Roman Empire (RE) network
   - ➟ Example 2: Shipwrecks network in the Mediterranean
2. Elementary group structures
   - ➟ Example 3: Dihedral groups
   - ➟ Example 4: Kariera kinship
3. Positional analysis and Role structure
   - ➟ Example 1b: RE network with Compositional equivalence
   - ➟ Example 1c: RE network with Formal Concept Analysis
4. Signed networks
   - ➟ Example 5: Incubator network
5. Valued, many-valued & multilevel systems
   - ➟ Example 6: Group of Twenty trade network

0. Introduction

network analysis using R

‘multiplex’ *10th anniversary!*

for computations of multiple networks

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 Ostoic <multiplex@post.com>

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 effective ways available to 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'

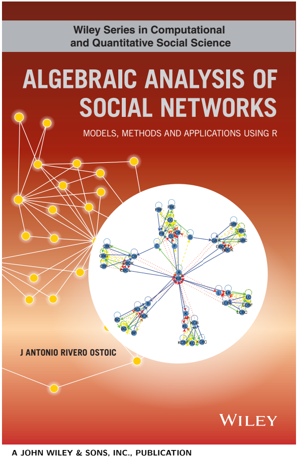
NeedsCompilation no

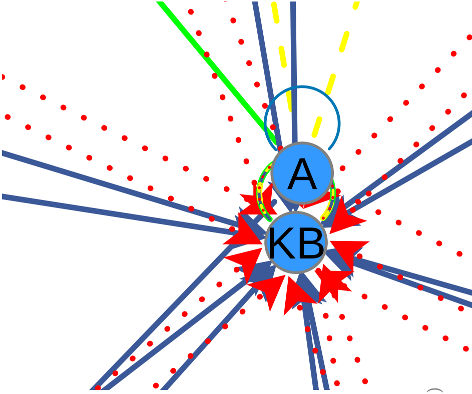
Repository CRAN

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

R topics documented:

‘multigraph’ *to plot multiplex networks*





(Cayley graph)

Getting started

In the **R** console, **R** IDE, or notebook with **R** kernel:

```r
# install packages from CRAN
install.packages("multiplex", "multigraph")
# or GitHub versions
devtools::install_github("mplex/multiplex")
devtools::install_github("mplex/multigraph")
```

```r
# load packages
library("multigraph")
# Loading required package: multiplex
```

```r
packageVersion("multigraph")
```

```
[1] '0.99'
```

```r
packageVersion("multiplex")
```

```
[1] '3.0.0'
```

Transforming and plotting network data

$(1, 2)$

```r
multiplex::transf("1, 2", type = "toarray")
```

```
  1 2
1 0 1
2 0 0
```

```r
multigraph::multigraph("1, 2", cex = 18, lwd = 20, rot = -90, pos = 0)
```

```r
scp <- list(cex = 18, lwd = 20, rot = -90, pos = 0, vedist = -2)
multigraph("1, 2", scope = scp)
```

Representation of network data

Undirected

$\{1, 2\}$

```r
matrix(c(0,1,1,0), nrow = 2, ncol = 2)
```

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

```r
# R native pipe
"1, 2" |>
  multigraph(directed = FALSE, scope = scp)
```

Representation of network data

Multiplex

$(1, 2); (2, 1)$

```r
transf(list("1, 2", "2, 1"))
```

```
, , 1

  1 2
1 0 1
2 0 0

, , 2

  1 2
1 0 0
2 1 0
```

```r
multigraph(list("1, 2", "2, 1"), scope = scp, ecol = 1, bwd = .7)
```

Representation of network data

Multiplex

$(1, 2); (2, 1)$

```
, , 1

  1 2
1 0 1
2 0 0

, , 2

  1 2
1 0 0
2 1 0
```

```r
net <- list("1, 2", "2, 1")
multigraph(net, scope = scp, ecol = 1, bwd = .7, swp = TRUE)
```

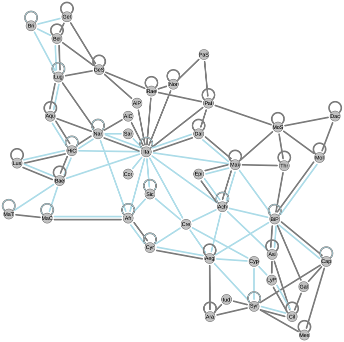
1. Visualization of multigraphs

Example 1a: Roman Empire network

Roman Empire (RE) transport network: Multiplex and undirected

ca. AD 125

routes based on Rodrigue (2013)



(in press Springer Nature)

```r
sdam::plot.map(type = "rp")
```

RE transport network

```r
# array of RE transport network
load(file = "./data/REtn.rda")
```

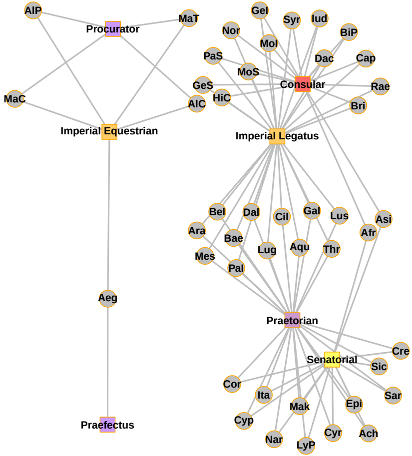
```r
# look at network 'dimensions'
dim(REtn)
```

```
[1] 45 45  2
```

```r
# plot multigraph with scope
scp <- list(cex = 2, pos = 0, lty = 1, bwd = .5, lwd = 3, fsize = 8, col = "gray",
+             ecol = c("#808080","#AFDDE9"))
REtn |>
  multigraph(directed = FALSE, layout = "force", scope = scp)
```

Roman provinces political affiliations: Bipartite graph

ca. AD 117



(*in press* Springer Nature)

RE political affiliations of provinces

```r
# data frame of RE goverment types
load(file = "./data/REgt.rda")
```

```r
# first three entries
REgt |>
  head(3)
```

```
    Senatorial Imperial Legatus Imperial Equestrian Consular Praetorian Praefectus Procurator
Ach          1                0                   0        0          1          0          0
Aeg          0                0                   1        0          0          1          0
Afr          1                0                   0        1          0          0          0
```

RE political affiliations of provinces

Depicting Bipartite graphs

```
# clustering for two types of vertices
list(rep(1, nrow(REgt)), c(1,2,2,3,4,5,5))
```

```
[[1]]
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

[[2]]
[1] 1 2 2 3 4 5 5
```

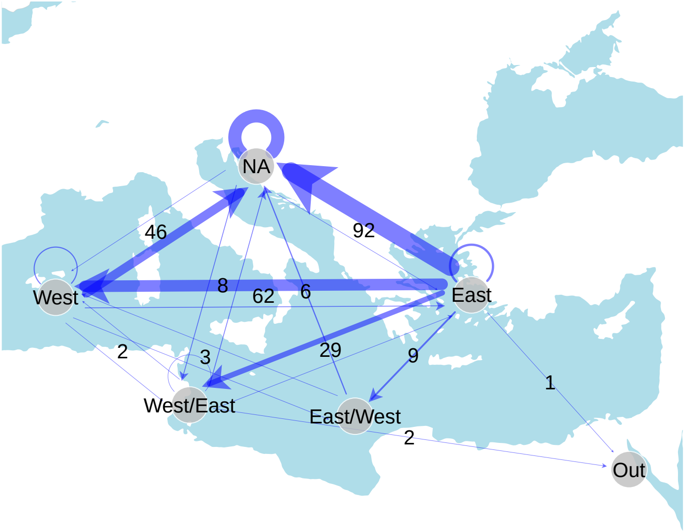
```
# scope for bipartite graph (v0.99+ for vertex clustering)
scpb <- list(cex = 3, pos = 0, fsize = 8, fstyle = "bold", vcol0 = "orange",
+             vcol = c(8,"#FFFF66","#FFCC66","#FF6666","#CC99CC","#CC99FF"),
+             vclu = list(NULL, c(1,2,2,3,4,5,5)), ecol = 8, lwd = 2)
REgt |>
  multigraph::bmgraph(layout = "force", seed = 4, scope = scpb)
```

$1b$ Graphs on cartographic maps

Example 2: Shipwrecks network in the Mediterranean

Shipwrecks traffic between aggregated sea regions in the Mediterranean Basin

Graph representation with cartographical map



Graph representation with cartographical map, coordinated system & missing data

```r
# aggregated sea regions in the Mediterranean
load(file = "./data/elsr.rda")
```

```r
# entries in data frame
head(elsr)
```

```
  Place of origin Place of destination from   to fra til
1           Egypt       Northern Italy    E  L,D   2  12
2             Cos       Northern Italy    A  L,D   2  12
3          Rhodes       Northern Italy    A  L,D   2  12
4          Aegean       Northern Italy    A  L,D   2  12
5            <NA>                 <NA> <NA> <NA>   0   0
6          Rhodes       Northern Italy    A  L,D   2  12
```

➟ Columns `fra` and `til` make the *edge list* of aggregated relations

Graph representation with cartographical map and coordinated system

```
# valued graph with loop scale in scope
scpv <- list(cex = 5, pos = 0, ecol = 4, col = "gray", lscl = .15)
```

```
# first cartographical map as background
sdam::plot.map(type = "med", new = TRUE)
# new layer with graph of relations 5 and 6 in edge list
multiplex::transf(elsr[,5:6], type = "toarray", na.rm = FALSE,
+     lbs = c("NA","West","West/East","East","East/West","Out")) |>
  multigraph::multigraph(valued = TRUE, values = TRUE, loops = TRUE,
+        scope = scpv, new = TRUE, coord = read.table(header = FALSE, text = "
+        0.3499740 0.52921756
+        0.0000000 0.30046916
+        0.2333714 0.11265614
+        0.7252469 0.30461617
+        0.5221496 0.09305772
+        1.0000000 0.00000000 "))
```

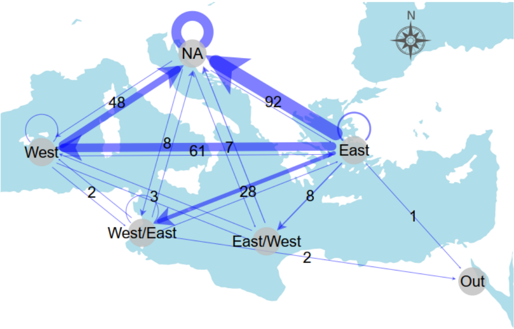


Figure 3.5: Shipwrecks traffic between aggregated sea regions in the Mediterranean Basin.
Graph representation with cartographical map.

2. Elementary structures

Example 3: Dihedral groups

Typology of multiple network structures

Simple networks:

- *(Simple) graphs*, *matrices*

  ➠ for relations between actors

Multiplex networks:

- *Multigraphs*, *arrays*

  ➠ for (types of) relations between actors

- *Cayley graphs*, *tables*

  ➠ for relationships between relations

☛ *Different types of algebraic structures are represented by tables*

Algebraic representation of multiplex networks

Typology

Type of structure	Algebraic object
Elementary	*Group*
Complex	*Semigroup*, *Semiring*, *Lattice*, etc.

Group: Elementary structure

A *group* is an algebraic structure with an *element set* and an endowed *operation*:

$$\langle\, G,\ \cdot \,\rangle$$

That for all a, b, c, and a neutral element $e \in G$ satisfies axioms:

$$\begin{aligned}
\text{Identity:}\quad & a \cdot e = e \cdot a = a \\
\textit{Inversion}\text{:}\quad & a \cdot a^{-1} = a^{-1} \cdot a = e \\
\text{Associativity:}\quad & (a \cdot b) \cdot c = a \cdot (b \cdot c) \\
\text{Closure:}\quad & a \cdot b \in G \qquad \text{(for all} \quad a, b\text{)}
\end{aligned}$$

Group structure by permutations

Theorem (Cayley)

All of group theory can be found in permutations.

➟ we focus on permutation symmetry

A *permutation* operator is represented by a *permutation matrix*

➟ having one entry in each row and in each column, and 0 elsewhere

$$\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \rightarrow \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}$$

Group Structures

Definition (Permutation Group on X)

The *permutation group on* X is the set of all permutations S_X on X

Definition (Symmetric Group of order n, S_n)

The *symmetric group* on a n-element set $\{1, 2, \ldots, n\}$ is the set of all permutations with $n!$ bijections σ, $S_n = \{\sigma_1, \sigma_2, \ldots, \sigma_{n!}\}$.

- If $X = \{1, 2, \ldots, n\}$ then $S_X = S_n$
  - ➟ the symmetric groups on n-elements are permutation groups

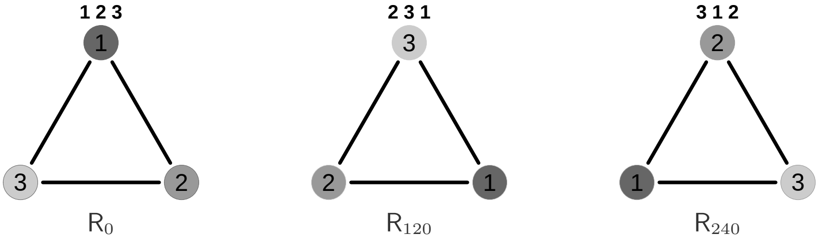
Definition (Dihedral Group of degree n, D_n, $n > 2$)

The set of all permutations which are symmetries on a regular n-sided polygon and the composition operation $\circ$ makes the *dihedral group* $(D_n, \circ)$.

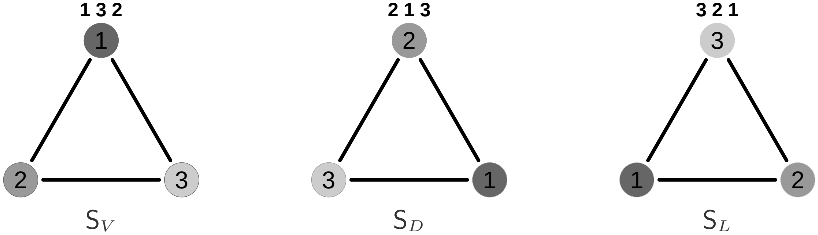
- the order of a dihedral group is twice its degree

Group of symmetries of the equilateral triangle (Dihedral group, D_3)

Rotations



Reflections



Cayley table of D_3

$\mathsf{R}_\circ$ rotations; S_V: mirror; S_D and S_L are diagonals

$\circ$	R_0	R_{120}	R_{240}	S_V	S_D	S_L
R_0	R_0	R_{120}	R_{240}	S_V	S_D	S_L
R_{120}	R_{120}	R_{240}	R_0	S_D	S_L	S_V
R_{240}	R_{240}	R_0	R_{120}	S_L	S_V	S_D
S_V	S_V	S_L	S_D	R_0	R_{240}	R_{120}
S_D	S_D	S_V	S_L	R_{120}	R_0	R_{240}
S_L	S_L	S_D	S_V	R_{240}	R_{120}	R_0

Generators of D_3

Define dihedral group family generators as permutation matrices

```r
# transform and sort for a lexicographic order
D3 <- list(F = c("1, 3","2, 1","3, 2"), G = c("1, 1","2, 3","3, 2")) |>
  transf(type = "toarray", sort = TRUE)
```

```
, , F

  1 2 3
1 0 0 1
2 1 0 0
3 0 1 0

, , G

  1 2 3
1 1 0 0
2 0 0 1
3 0 1 0
```

String relations in D_3

```r
# unique strings as word tables
D3 |>
  multiplex::strings()
```

```
$wt
, , F         , , FF        , , GF

  1 2 3         1 2 3         1 2 3
1 0 0 1       1 0 1 0       1 0 0 1
2 1 0 0       2 0 0 1       2 0 1 0
3 0 1 0       3 1 0 0       3 1 0 0

, , G         , , FG        , , GG

  1 2 3         1 2 3         1 2 3
1 1 0 0       1 0 1 0       1 1 0 0
2 0 0 1       2 1 0 0       2 0 1 0
3 0 1 0       3 0 0 1       3 0 0 1
```

Equations in group structure, D_3 $(k = 3)$

Argument `equat` from `strings()` to find group equations & identity

```
D3 |> strings(equat = TRUE, k = 3)
```

```
$equat
$equat$F
[1] "F"   "GGF" "FGG"

$equat$G
[1] "G"   "GGG" "FGF"

$equat$FF
[1] "FF"  "GFG"

$equat$FG
[1] "FG"  "GFF"

$equat$GF
[1] "GF"  "FFG"

$equat$GG
[1] "GG"  "FFF"


$equate
$equate$e
[1] "e"   "GG"  "FFF"
```

Group structure, D_3

```r
# semigroup with numerical format
D3 |> multiplex::semigroup()
```

```
...

$st
[1] "F"  "G"  "FF" "FG" "GF" "GG"

$S
  1 2 3 4 5 6
1 3 4 6 5 2 1
2 5 6 4 3 1 2
3 6 5 1 2 4 3
4 2 1 5 6 3 4
5 4 3 2 1 6 5
6 1 2 3 4 5 6

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

Group structure, D_3

symbolic format

Function `semigroup()` allows finding the group structure

```
# semigroup structure with symbolic format
D3 |> semigroup(type = "symbolic") |> getElement("S")
```

```
    F  G FF FG GF GG
F  FF FG GG GF  G  F
G  GF GG FG FF  F  G
FF GG GF  F  G FG FF
FG  G  F GF GG FF FG
GF FG FF  G  F GG GF
GG  F  G FF FG GF GG
```

Permutation of the group structure, D_3

`perm()` for rearrangement of elements' group structure

```
D3S <- D3 |> semigroup() |> multiplex::perm(clu = c(2,4,3,5,6,1))
```

```
  6 1 3 2 4 5
6 6 1 3 2 4 5
1 1 3 6 4 5 2
3 3 6 1 5 2 4
2 2 5 4 6 3 1
4 4 2 5 1 6 3
5 5 4 2 3 1 6
```

This comes from the string labels where `GG` is the identity element

```
..
$st
[1] "F"  "G"  "FF" "FG" "GF" "GG"
...
```

Depiction of group structure: Cayley graph

Definition (Cayley graph)

The *Cayley graph* Γ of a group G with respect to a generating set $C \subseteq G$:

$$\Gamma = \Gamma(G, C).$$

- G is the node set in Γ
- A generator $c \in C$ connects two nodes $a, b \in G$ whenever $b = ca$
  - ➟ i.e. all pairs of the form $(a, c \cdot b)$ make the edge set in Γ

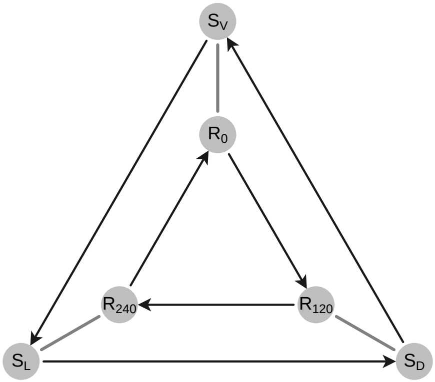
Cayley colour graph

Example (Cayley graph, integers under addition $\mathbb{Z}_2$)

```
      e x               e=ee  => solid loop
    e e x               e=xx  => solid loop
    x x e
                        x=ex  => dashed arc
                        x=xe  => dashed arc
```

Cayley graph of Dihedral group D_3

Group of symmetries of the equilateral triangle



Depiction of the group structure, D_3

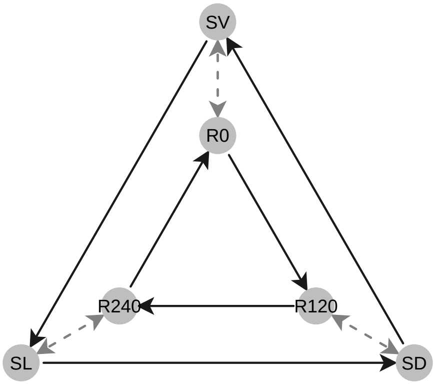
```r
# relabel strings in semigroup specifying generators
lbs3 <- c("R0", "R120", "R240", "SV", "SD", "SL")
D3S <- D3S |>
  multiplex::as.semigroup(gens = c(2, 4), lbs = lbs3)
```

```
...
$st
[1] "R0"   "R120" "R240" "SV"   "SD"   "SL"

$gens
[1] "R120" "SV"

$S
       R0 R120 R240   SV   SD   SL
R0     R0 R120 R240   SV   SD   SL
R120 R120 R240   R0   SD   SL   SV
R240 R240   R0 R120   SL   SV   SD
SV     SV   SL   SD   R0 R240 R120
SD     SD   SV   SL R120   R0 R240
SL     SL   SD   SV R240 R120   R0

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

Depiction of group structure

Cayley graph, D_3

```
# plot Cayley colour graph with a 2-radii concentric layout
scpD3 <- list(cex = 7, lwd = 3, pos = 0, col = 8, fsize = 16)
D3S |> multigraph::ccgraph(conc = TRUE, nr = 2, scope = scpD3)
```



Workflow for D_3 and D_4

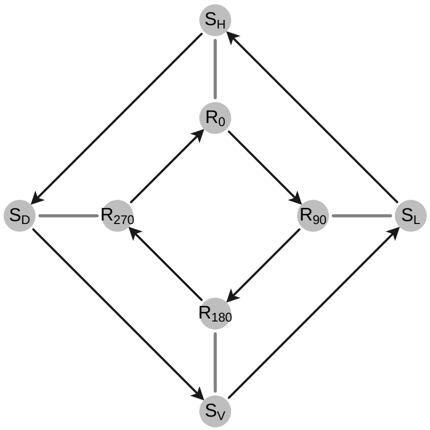
Depiction of group structure

```
# D_3
scpD3 <- list(cex = 7, lwd = 3, pos = 0, col = 8, fsize = 16, lty = 1)
D3 |>
  semigroup() |>
  perm(clu = c(2,4,3,5,6,1)) |>
  as.semigroup(gens = c(2,4),lbs = c("R0","R120","R240","SV","SD","SL")) |>
  multigraph::ccgraph(scope = scpD3, conc = TRUE, nr = 2)
```

```
# D_4
scpD4 <- list(cex = 6, vcol = 8, lwd = 3, pos = 0, ecol = 6, lty = 1)
lbs4 <- c("R0", "R90", "R180", "R270", "SH", "SL", "SV", "SD")
list(R = c("2, 1","3, 2","4, 3","1, 4"),
+    S = c("1, 3","2, 2","3, 1","4, 4")) |>
  transf(type = "toarray", sort = TRUE) |>
  semigroup() |> perm(clu = c(2,5,3,6,8,1,4,7)) |>
  as.semigroup(gens = c(2,5), lbs = lbs4) |>
  multigraph::ccgraph(scope = scpD4, conc = TRUE, nr = 2, undRecip = TRUE)
```

Group of symmetries of the square

Cayley graph of dihedral group D_4



$2b.$ Group structure in social networks

Example 4: Kariera kinship

Kariera society kinship system and group structure

- Despite the symmetry, algebraic groups can model human societies

- Some primitive societies like the *Kariera* from Western Australia have kinship networks that follow the rules of a group structure

  ➟ where *primitive* means “first of its class”

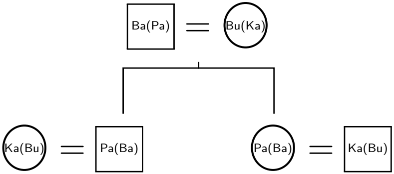
- The Karieras have four clans with specific rules of marriage & descent: *Banaka*, *Burung*, *Karimera*, and *Palyeri*.

  ➟ data collected by Radcliffe-Brown, analysed by White (1963)

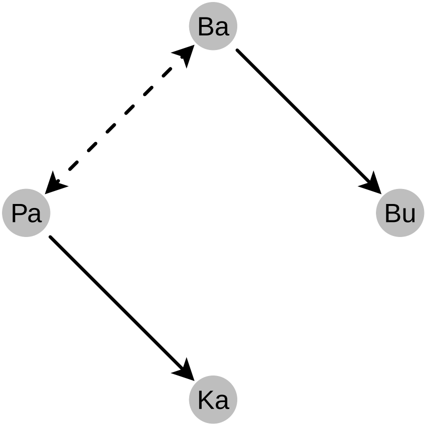
Kariera rules for marriage & descent (I)

Clans: Banaka (Ba), Burung (Bu), Karimera (Ka), Palyeri (Pa)

Two types of descent rules among Banaka and Palyeri (ego male)



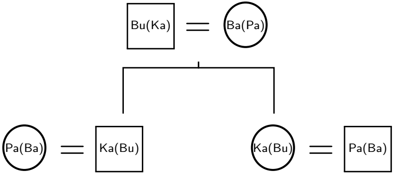
square: male, circle: female



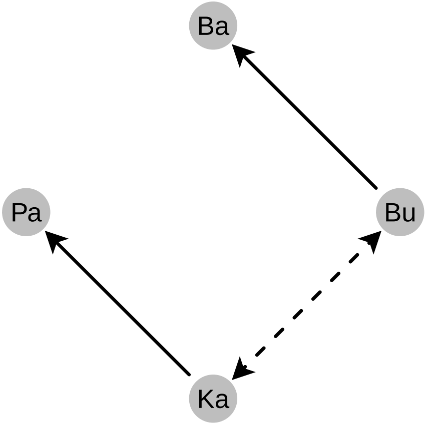
Kariera rules for marriage & descent (II)

Clans: Banaka (Ba), Burung (Bu), Karimera (Ka), Palyeri (Pa)

Two types of descent rules among Burung and Karimera (ego male)



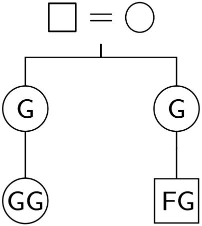
square: male, circle: female



Parallel-cousins marriages in kinship networks

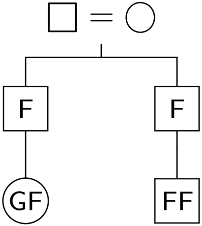
identifiers, F for male and G for female, are with right multiplication

FG = GG



(a) Matrilineal

GF = FF

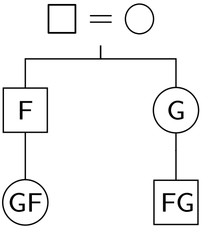


(b) Patrilineal

Cross-cousins marriages in kinship networks

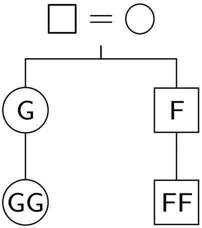
identifiers, F for male and G for female, are with right multiplication

FG = GF



(a) Matrilineal

FF = GG



(b) Patrilineal

Kariera kinship system

```r
# create permutation matrices with marriage & descent rules
kks <- transf(list(F = c("1, 2", "3, 4", "2, 1", "4, 3"),
+                  G = c("1, 4", "2, 3", "3, 2", "4, 1")))
```

```
, , F

  1 2 3 4
1 0 1 0 0
2 1 0 0 0
3 0 0 0 1
4 0 0 1 0

, , G

  1 2 3 4
1 0 0 0 1
2 0 0 1 0
3 0 1 0 0
4 1 0 0 0
```

Group structure as multiplication table

Kariera kinship system

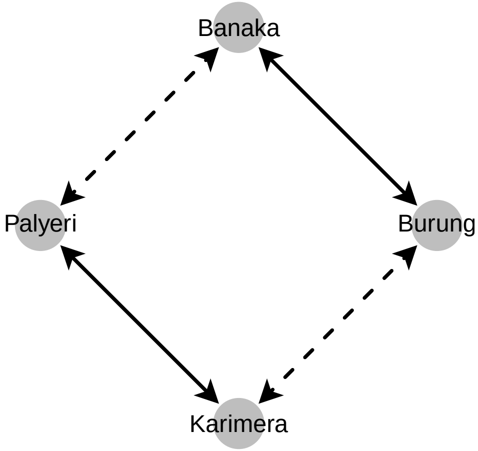
The *multiplication table* reflects the group structure of the clan system

```
# Group structure with a symbolic format
semigroup(kks, type = "symbolic")
```

```
$dim
[1] 4
...

$ord
[1] 4

$st
[1] "F"  "G"  "FF" "FG"

$S
    F  G FF FG
F  FF FG  F  G
G  FG FF  G  F
FF  F  G FF FG
FG  G  F FG FF

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

Rules of marriage & descent in Kariera kinship system

```r
# visualize marriage & descent rules in the Kariera
multigraph(kks, scope = scpD3, ecol = 1, collRecip = TRUE,
+          lbs = c("Banaka", "Burung", "Karimera", "Palyeri"))
```



Set of equations to identify cross- and parallel-cousins marriages

The *set of equations* to detect allowed marriage types by commutation

```r
# equations allow finding marriage types in 'kks'
kks |>
  strings(equat = TRUE)
```

```
...
$st
[1] "F"  "G"  "FF" "FG"

$equat
$equat$FF
[1] "FF" "GG"

$equat$FG
[1] "FG" "GF"


$equate
$equate$e
[1] "e"  "FF" "GG"
```

☛ *Both cross-cousins marriages are permitted in the Kariera*

Algebraic constraints in group structures

Two *algebraic constraints* for the analysis of the elementary structures:

- *Multiplication table* with relations between the different types of tie
- *Set of equations* among different types of tie

☛ *Complex structures have additional algebraic constraints*

3. Positional analysis and Role structure

Example 1b-c: RE transport network & political affiliations

Affiliation 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*"

- *Domain* and *codomain* are *constituent* parts in a 2-mode matrix data

  ➠ connected by constituent relations

- Positional analyses of affiliation networks

  ➠ Composition equivalence

  ➠ Formal Concept Analysis

Positional analysis with Compositional equivalence

Incorporation of node attributes from government types

```
colnames(REgt)
```

```
 [1] "Senatorial"          "Imperial Legatus"    "Imperial Equestrian"
 [4] "Consular"            "Praetorian"          "Praefectus"
 [7] "Procurator"
```

```
# Senatorial/Imperial edge list
rpsi <- cbind(REgt[,1], (REgt[,2]+REgt[,3]))
rownames(rpsi)<-rownames(REgt); colnames(rpsi)<-c("Senatorial","Imperial")
```

```
rpsi |> head(4)
```

```
    Senatorial Imperial
Ach          1        0
Aeg          0        1
Afr          1        0
AlC          0        1
```

Positional analysis with Compositional equivalence

Incorporation of node attributes from government types

```
# edge list into diagonal matrices with nodal attributes
sinet <- multiplex::edgel(rpsi, attr = TRUE, rownames = TRUE)
# network 'dimensions'
dim(sinet)
```

```
[1] 45 45  2
```

```
# Senatorial/Imperial in diagonal matrices
rbind(diag(sinet[,,1]), diag(sinet[,,2]))
```

```
     Ach Aeg Afr AlC AlP Aqu Ara Asi Bae Bel BiP Bri Cap Cil Cor Cre Cyp Cyr Dac Dal
[1,]   1   0   1   0   0   0   0   1   0   0   0   0   0   0   1   1   1   1   0   0
[2,]   0   1   0   1   1   1   1   0   1   1   1   1   1   1   0   0   0   0   1   1
     Epi Gal GeI GeS HiC Ita Iud Lug Lus LyP MaC Mak MaT Mes MoI MoS Nar Nor PaI PaS
[1,]   1   0   0   0   0   1   0   0   0   1   0   1   0   0   0   0   1   0   0   0
[2,]   0   1   1   1   1   0   1   1   1   0   1   0   1   1   1   1   0   1   1   1
     Rae Sar Sic Syr Thr
[1,]   0   1   1   0   0
[2,]   1   0   0   1   1
```

Positional analysis with Compositional equivalence

RE transport network and senatorial/imperial provinces

```r
# bind the two networks
REtnsi <- multiplex::zbind(REtn, sinet)
# network 'dimensions'
dim(REtnsi)
```

```
[1] 45 45  4
```

With compounds of length $k = 6$, first non-trivial stable partition of the CPH

```r
# workflow: net -> RelBox -> CPH -> clu
cluCPH6 <- REtnsi |>
  multiplex::rbox(k = 6) |>
  multiplex::cph() |>
  multiplex::diagram.levels(perm = TRUE) |> getElement("clu")
```

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

Role structure: Word tables

Positional analysis with Compositional equivalence

```r
# otherwise, load vector with clustering
load(file = "./data/cluCPH6.rda")
# use string label relations in the reduction
REtnsi |>
  multiplex::reduc(clu = cluCPH6, slbs = c("t","m","s","i")) |>
  strings()
```

```
$wt
, , t                 , , ti

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

, , s                 , , it

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

, , i

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

Role structure: Equations in string relations

Positional analysis with Compositional equivalence

```r
# obtain a set of equations from the reduction
REtnsi |>
  reduc(clu = cluCPH6, slbs = c("t","m","s","i")) |>
  strings(equat = TRUE)
```

```
$equat
$equat$t
[1] "t"  "m"  "tt" "ts" "st"

$equat$s
[1] "s"  "ss"

$equat$i
[1] "i"  "ii" "si" "is"


$equate
$equate$e
[1] "e"  "s"  "ss"
```

Role structure and Green's relations

Positional analysis with Compositional equivalence

```
# semigroup role structure with principal ideals
REtnsi |>
  reduc(clu = cluCPH6, slbs = c("t","m","s","i")) |>
  semigroup(type = "symbolic") |>
  multiplex::green.rel()
```

```
$S
    t  s  i ti it
t   t  t ti ti  t
s   t  s  i ti it
i  it  i  i  i it
ti  t ti ti ti  t
it it it  i  i it

...

$D
   t  it   s    i  ti

t  t  t  | t  | ti ti
ti t  t  | ti | ti ti
   -  -  - -  - -  -
s  t  it | s  | i  ti
   -  -  - -  - -  -
i  it it | i  | i  i
it it it | it | i  i
```

Multilevel role structure

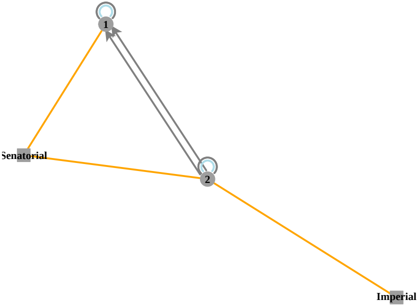
Two domains of the multilevel structure are the reductions of `REtn` and `rpsi`

```
# multilevel with binomial projection of domains
REtn |> reduc(clu = cluCPH6) |>
  multiplex::mlvl(y = reduc(rpsi,clu = cluCPH6,row = TRUE), type = "bpn")
```

```
$mlnet
, , roads

           1 2 Senatorial Imperial
1          1 1          0        0
2          1 1          0        0
Senatorial 0 0          0        0
Imperial   0 0          0        0

, , shipr

           1 2 Senatorial Imperial
1          1 1          0        0
2          1 1          0        0
Senatorial 0 0          0        0
Imperial   0 0          0        0

, , 3

           1 2 Senatorial Imperial
1          0 0          1        0
2          0 0          1        1
Senatorial 0 0          0        0
Imperial   0 0          0        0

...
$modes
[1] "1M" "1M" "2M"
...
```

Visualization of multilevel role structure

Positional analysis with Compositional equivalence, $k = 6$




```
# scope multilevel with circles and squares as 'pch'
scpML <- list(cex = 3.3, pos = 0, vcol = 8, vcol0 = 8, bwd = .6, lty = 1, lwd = 3,
+             ffamily = "serif", fstyle = "bold", hds = .3,
+             ecol = c("#808080","#AFDDE9","orange"))
REtn |>
  reduc(clu = cluCPH6) |>
  mlvl(y = reduc(rpsi,clu = cluCPH6,row = TRUE), type = "bpn") |>
  multigraph::mlgraph(layout = "force", scope = scpML)
```

Positional analysis with Formal Concept Analysis: Galois connections

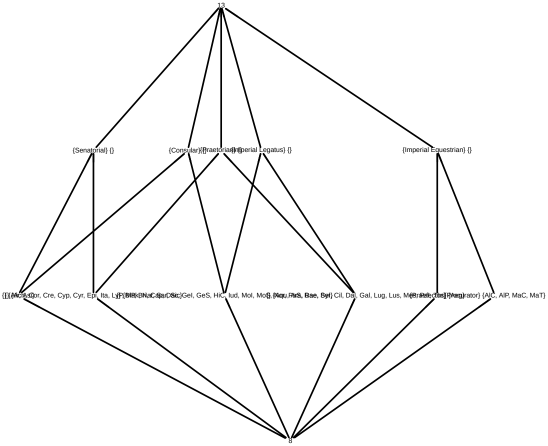
```
# government types of three Roman provinces
REgt |> head(3)
```

```
    Senatorial Imperial Legatus Imperial Equestrian Consular Praetorian Praefectus Procurator
Ach          1                0                   0        0          1          0          0
Aeg          0                0                   1        0          0          1          0
Afr          1                0                   0        1          0          0          0
```

```
# reduced labeling in Galois connection (gc)
REgt |> multiplex::galois(labeling = "reduced") |> getElement("gc")
```

```
$reduc
...
$reduc$Praefectus
[1] "Aeg"
...
$reduc[[9]]
[1] "Afr, Asi"

$reduc[[10]]
[1] "Ach, Cor, Cre, Cyp, Cyr, Epi, Ita, LyP, Mak, Nar, Sar, Sic"
...
```

Positional analysis with Formal Concept Analysis: Concept diagram




```r
require("Rgraphviz")
REgt |> multiplex::galois(labeling = "reduced") |>
  multiplex::partial.order(type = "galois") |>
  multiplex::diagram(fsize = 24, lwd = 2)
```

Levels in Concept diagram (produced by graphviz)

Positional analysis with Formal Concept Analysis

```r
require("Rgraphviz")
REgt |> galois(labeling = "reduced") |>
  partial.order(type = "galois") |>
  diagram.levels()
```

```
$`2`
[1] "{Senatorial} {}"          "{Imperial Legatus} {}"    "{Imperial Equestrian} {}"
[4] "{Consular} {}"            "{Praetorian} {}"

$`3`
[1] "{Praefectus} {Aeg}"
[2] "{Procurator} {AlC, AlP, MaC, MaT}"
[3] "{} {Afr, Asi}"
[4] "{} {Ach, Cor, Cre, Cyp, Cyr, Epi, Ita, LyP, Mak, Nar, Sar, Sic}"
[5] "{} {BiP, Bri, Cap, Dac, GeI, GeS, HiC, Iud, MoI, MoS, Nor, PaS, Rae, Syr}"
[6] "{} {Aqu, Ara, Bae, Bel, Cil, Dal, Gal, Lug, Lus, Mes, PaI, Thr}"

$`4`
[1] "8"

$`1`
[1] "13"
```

Constructing the clustering information from formal concepts

```
# vector of the seven formal concepts with provinces
X <- vector("list", length = length(c(6:7,9:12))); j <- 1
for(k in c(6:7,9:12)) {
  X[[j]] <- REgt |> galois(labeling = "reduced") |>
    partial.order(type = "galois") |> dimnames() |>
    getElement(1) |> unlist() |> getElement(k)
j <- j+1L  }
X |> unlist() -> X
# extract provinces from these concepts
cls <- vector("list", length = length(X))
for(i in seq_len(length(X)))
        cls[[i]] <- strsplit(X[i],"\\} \\{")[[1]][2]
cls <- lapply(cls, function(z) { gsub("\\}","",z) }) |> multiplex::dhc()
# construct clustering info from FCA
cluFCA <- vector(length = nrow(REgt))
for(i in seq_len(length(X)))
        cluFCA[which(rownames(REgt)%in%cls[[i]])] <- i
```

```
cluFCA
```

```
[1] 4 1 3 2 2 6 6 3 6 6 5 5 5 6 4 4 4 4 5 6 4 6 5 5 5 4 5 6 6 4 2 4 2 6 5 5 4 5 6 5 5 4 4 5 6
```

Role structure FCA: Word tables & Equations

```
# or load FCA clustering & use string labels in reduction
load(file = "./data/cluFCA.rda")
REtn |> reduc(clu = cluFCA, slbs = c("t","m")) |> strings(equat = TRUE)
```

```
$wt
, , t                                , , tt

     [,1] [,2] [,3] [,4] [,5] [,6]        [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    0    0    1    0    1   [1,]    1    1    1    1    1    1
[2,]    0    1    1    1    1    0   [2,]    1    1    1    1    1    1
[3,]    0    1    1    1    1    1   [3,]    1    1    1    1    1    1
[4,]    1    1    1    1    1    1   [4,]    1    1    1    1    1    1
[5,]    0    1    1    1    1    1   [5,]    1    1    1    1    1    1
[6,]    1    0    1    1    1    1   [6,]    1    1    1    1    1    1

, , m                                , , mm

     [,1] [,2] [,3] [,4] [,5] [,6]        [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    0    0    1    1    0   [1,]    1    0    1    1    1    1
[2,]    0    1    1    0    0    1   [2,]    0    1    1    1    1    1
[3,]    0    1    1    1    1    0   [3,]    1    1    1    1    1    1
[4,]    1    0    1    1    1    1   [4,]    1    1    1    1    1    1
[5,]    1    0    1    1    1    1   [5,]    1    1    1    1    1    1
[6,]    0    1    0    1    1    1   [6,]    1    1    1    1    1    1

...
$equat
$equat$tt
[1] "tt" "tm" "mt"
```

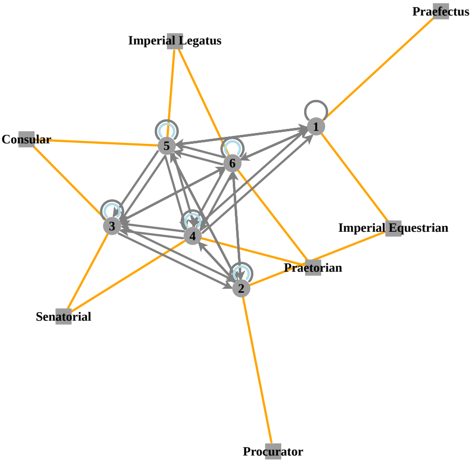
Role structure and Green's relations

Positional analysis with Formal Concept Analysis

```r
# semigroup role structure with principal ideals
REtn |> reduc(clu = cluFCA, slbs = c("t","m")) |>
  semigroup(type = "symbolic") |> green.rel()
```

```
$S
    t  m tt mm
t  tt tt tt tt
m  tt mm tt tt
tt tt tt tt tt
mm tt tt tt tt
...
$R
[1] t  tt mm |  m

$L
[1] t  tt mm |  m

$D
   t  tt mm   m

t  tt tt tt | tt
tt tt tt tt | tt
mm tt tt tt | tt
   -  -  -  - -
m  tt tt tt | mm
```

Multilevel role structure from Formal Concept Analysis




```
# recycle multilevel scope for graph with a random seed
REtn |> reduc(clu = cluFCA) |>
  mlvl(y = reduc(REgt,clu = cluFCA,row = TRUE), type = "bpn") |>
  mlgraph(layout = "force", scope = scpML)
```

4. Signed networks

Example 5: Incubator network

Structural Balance

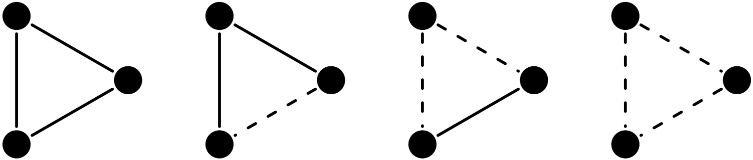
- Simmel (1950) studied "conflict as a mechanism for integration" in triadic relations
- Heider (1958) developed the *Structural Balance* theory as a special cases of transitivity
- Structural Balance theory applies to networks to see whether the system has an inherent equilibrium or not

  "*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 multiplex 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	$\rightarrow$	**clusterable**	$\rightarrow$	'*weak*' clusterable
(Cartwright & Harary, 1956)		(Davis, 1967)		

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

Classical

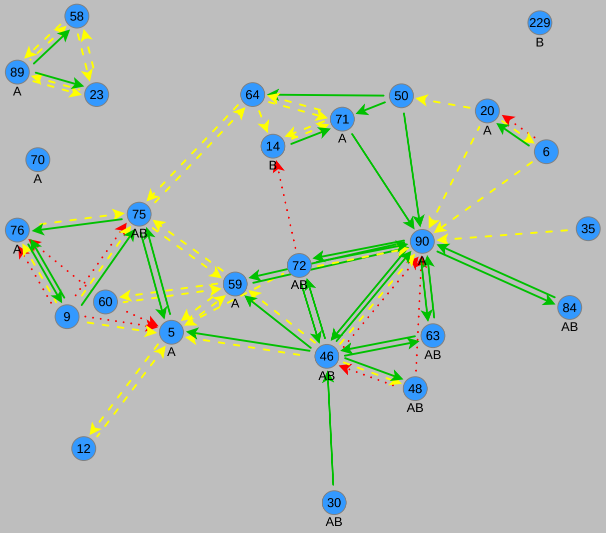
$\circ$	p	n	a
p	p	n	a
n	n	a	a
a	a	a	a

Extended

p $\rightarrow$ positive n $\rightarrow$ negative a $\rightarrow$ ambivalent

Incubator network "A"

Collaboration (green), Friendship (yellow), Competition (red)



Incubator network A

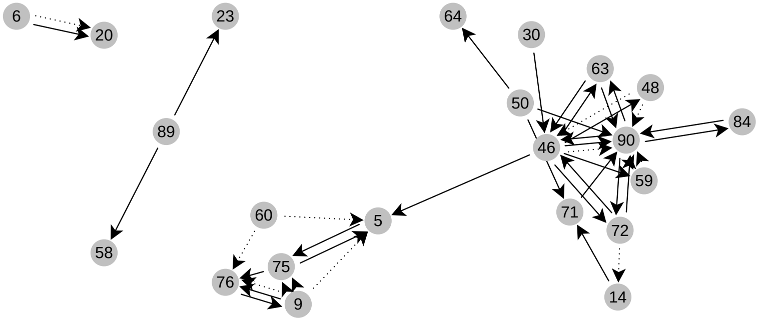
```
# incubator network A dataset and structure
utils::data("incA")
utils::str(incA)
```

```
List of 5
 $ 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" ...
 $ atnet:List of 1
  ..$ : num [1:5] 0 0 0 1 1
 $ IM   : num [1:4, 1:4, 1:7] 1 1 1 0 0 1 0 0 1 0 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ : NULL
  .. ..$ : NULL
  .. ..$ : chr [1:7] "C" "F" "K" "D" ...
 $ atIM : num [1:7] 0 0 0 0 0 0 1
 $ ...
```

```
# cooperation and competition ties in 'incA' without isolated actors
netA <- multiplex::rm.isol(incA$net[,,c(1,3)])
```

Signed structure in Incubator network A

```
# plot signed multigraph with scope
scpA <- list(ecol = 1, vcol = "#C0C0C0", cex = 3, fsize = 8, pos = 0, bwd = .5)
netA |>
  multigraph(scope = scpA, signed = TRUE, layout = "force", seed = 9)
```



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

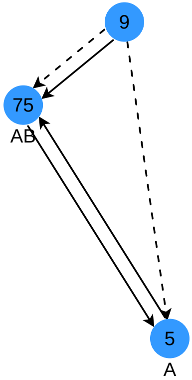
$\cdot$	o	n	p	a	q
o	o	o	o	o	o
n	o	q	n	n	q
p	o	n	p	a	q
a	o	n	a	a	q
q	o	q	q	q	q

$+$	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

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, $k > 2$

Signed Network *C and K in Incubator A*

```r
# create a "Signed" class object from matrices in netA
netAsg <- multiplex::signed(netA)
```

```
$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 function

```r
# arguments in function semiring()
formals(multiplex::semiring)
```

```
$x


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

$symclos
[1] TRUE

$transclos
[1] TRUE

$k
[1] 2

$lbs

```

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 equilibrium in Balance semiring

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

```
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 equilibrium in Cluster semiring

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

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

```
[1] FALSE
```

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

```
[1] TRUE
```

Weak balance structure with semipaths

```r
# length four and permutation with clustering
netAsg |> semiring(type = "balance", k = 4) |>
  perm(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 with paths

```r
# length four and permutation with clustering
netAsg |> semiring(type = "balance", symclos = FALSE, k = 4) |>
  perm(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
```

Main component of Incubator A

weak balance structure

```
# find components and isolates
multiplex::comps(netA)
```

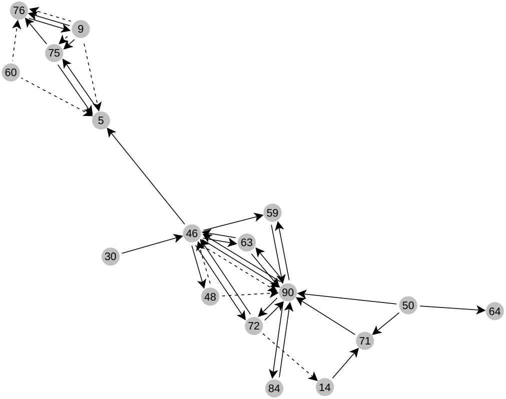
```
$com
$com[[1]]
 [1] "5"  "50" "59" "60" "63" "64" "71" "72" "75" "76" "84" "90" "9"  "14" "30" "46" "48"

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

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

```
# extract ties from component one in 'netA' with types of tie 1 and 3
# plus actor attributes into 'nsA'
com1 <- comps(netA)$com[[1]]
nsA <- incA$net[,,c(1,3,4:5)] |>
  rel.sys(type = "toarray", sel = com1)
```

Weak balance structure

Network relations 'C' and 'K' in main component of Incubator A

```r
nsA |>
  multigraph(layout = "force", seed = 123, scope = scpA)
```



Handling “scope” lists

Paths main component of Incubator A

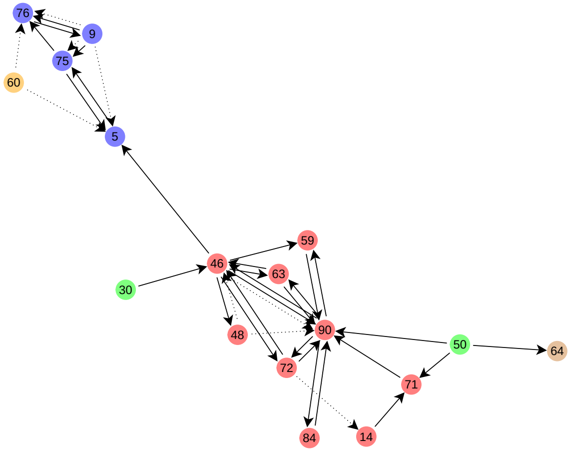
```
# scope with factions clustering
scpAc <- list(lty = c(1,3), clu = c(1,1,2,3,2,2,3,2,4,2,5,2,2,1,1,2,2),
+               vcol = c("blue","red","green","orange","peru"), alpha = .5)
# concatenating list scopes
c(scpAc, scpA)
```

```
$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"
...
```

Paths main component of Incubator A

Factions with weak balance structure

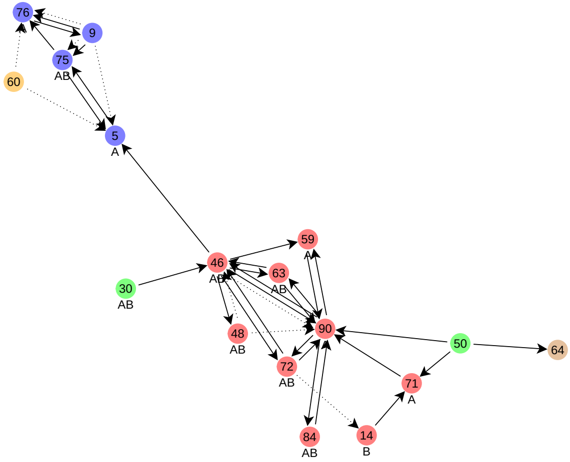
```
nsA |>
  multigraph(layout = "force", seed = 123, scope = c(scpA, scpAc))
```



Weak balance structure with paths

Social influence through comparison

```
nsA |> multigraph(att = nsA[,,3:4], layout = "force", seed = 123,
+                  scope = c(scpA, scpAc))
```



$5.$ Valued, many-valued & multilevel

Example 6: Group of Twenty trade network

Multilevel Structure of G20 Trade Network

Algebraic analyses and visualization of complex networks

Sunbelt XLIII Conference – Workshop

Antonio Rivero Ostoic

CESU, University of San Simón

multiplex@post.com

23 June 2023

Link

Relational structure in valued multiplex networks

- Assignments to labeled valued paths with the $\max - \min$ composition
  - ➟ minimum value of sending/receiving scores in nodes
  - ➟ then the maximum of these values

- For the G20 Trade network of milk (M) & honey (H)

$$\mathsf{M} \circ \mathsf{H} \;=\; \max_{k}\{\min\big(w_{\mathsf{M}}(i,k),\; w_{\mathsf{H}}(k,j)\big)\} \;=\; \mathsf{MH}$$

☛ `multiplex` supports relational structures of valued networks

Relational structure in valued multiplex networks

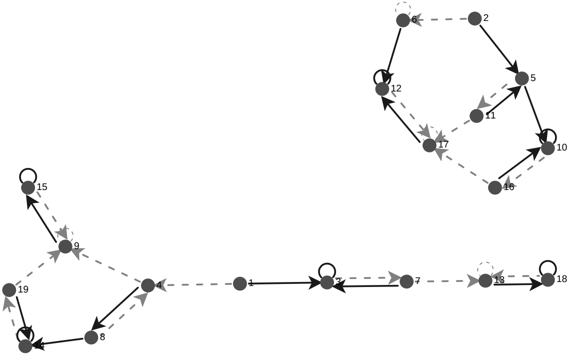
```r
load(file = "./data/vnet.rda")
```

```
vnet

, , M

      G7 BRICS MITKA
G7    19    23     4
BRICS  0     0     0
MITKA  1     8     0

, , H

      G7 BRICS MITKA
G7     3     0     0
BRICS 30     1     1
MITKA 13     1     0
```

Relational structure in valued multiplex networks

semigroup with $\max - \min$ *product*

```
# valued network's Cayley graph of semigroup
vnet |> semigroup(valued = TRUE) |> ccgraph(rot = 90, cex = 2, lwd = 2)
```



Many-valued contexts

(Wille, 1982)

- A *many-valued context*, $\mathbb{K}^V$ is defined as

$$\mathbb{K}^V = (\, G,\ M,\ W,\ I \,)$$

➟ G is the object set

➟ M is the many-valued attributes

➟ W are "weights" or attribute values

➟ I is the incidence relation between G and M

☛ the context is said to be an k-*valued context* when W has k elements

Many-valued context

Economic and socio-demographic indicators of G20 countries

```r
# four-valued context
load(file = "./data/G20mv.rda")
```

```
    T  N  G  H  P  A
ARG vl vl l  l  vl l
AUS vl vl vh vh vl h
BRA vl l  l  l  l  h
CAN l  l  vh vh vl h
CHN vh vh vl vl vh h
DEU h  h  vh vh l  vl
FRA l  l  h  h  vl vl
GBR l  l  h  vh vl vl
IDN vl vl vl vl h  l
IND l  l  vl vl vh l
ITA l  l  h  h  vl vl
JPN h  h  h  h  l  vl
KOR l  vl h  h  vl vl
MEX l  vl vl l  l  l
RUS l  vl l  l  l  vh
SAU vl vl l  h  vl l
TUR vl vl vl l  vl vl
USA vh vh vh vh h  h
ZAF vl vl vl vl vl vl

T=trade, N=nom_GDP, G=GDP_PC; H=HDI, P=population, A=area
```

Conceptual scaling

many-valued context of G20 network

	very-high	*high*	*low*	*very-low*
very-high	1	0	0	0
high	0	1	0	0
low	0	0	1	0
very-low	0	0	0	1

Nominal $\mathbb{N}_n = \langle n, n, = \rangle$

$\leq$*very-high*	$\leq$*high*	$\geq$*low*	$\geq$*very-low*
1	1	0	0
0	1	0	0
0	0	1	0
0	0	1	1

Inter-ordinal $\mathbb{I}_n = \langle n, n, \leq \mid \ n, n, \geq \rangle$

Scaling matrix in `scl` to dichotomize valued data frame in `G20mv`

```
# dot separation between extents and scale labels
G20mv |>
  multiplex::cscl(scl, sep = ".")
```

☛ plot Concept lattice, and apply order filters and order ideals

Pathfinder semiring and Triangle inequality

one-mode valued networks

For the analysis of adjacency matrices:

- *Pathfinder semiring* for symmetric valued relations

  ➟ matrix reflects the "proximity" between pairs of network members

- *Triangle inequality* for asymmetric valued ties

  ➟ then *salient* structure of valued network

Use functions `pfvn()` and `ti()` from `multiplex`

References

White, HC *An Anatomy of Kinship: Mathematical models for structures of cumulated roles*. Prentice-Hall. 1963

Pattison, PE *Algebraic Models for Social Networks*. Cambridge University Press. 1993

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, JAR *Algebraic Analysis of Social Networks*. Wiley. 2021

Ostoic, JAR 'Relational systems of transport network and provinces in ancient Rome,' in *Mathematics for social sciences and arts – algebraic modeling*. Springer Nature. 2023

R Development Core Team, ***R**: A language and environment for statistical computing*, v4.2.2

Ostoic, J.A.R. `multigraph`*: Plot and Manipulate Multigraphs*. **R**package version 0.98

Ostoic, J.A.R. `multiplex`*: Algebraic Tools for the Analysis of Multiple Social Networks*. **R**package version 3.0

Thanks!

Sunbelt 2023 Planning Committee

Texas A&M University

github.com/mplex

multiplex@post.com