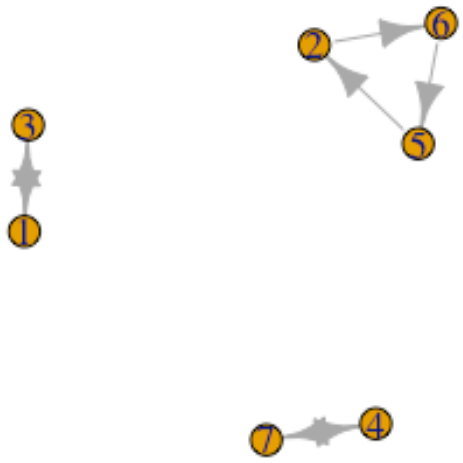
A1_Problem4

```
library("gurobi")

## Loading required package: slam

library("Matrix")
library("igraph")

##
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union

n <- 7
C.ij = matrix(c(10000, 82, 34, 64, 141, 201, 62, 82, 10000, 94, 124,
79, 142, 123, 34, 94, 10000, 57, 154, 214, 52, 64, 124, 57, 10000, 184,
244, 22, 141, 79, 154, 184, 10000, 81, 179, 201, 142, 214, 244, 81,
10000, 239, 62, 123, 52, 22, 179, 239, 10000), nrow = n, ncol=n, byrow
= TRUE)
cvec = as.vector(t(C.ij))

bvec = c(rep(1, n), rep(1, n))
dir = c(rep("=", n), rep("=", n))



Amat = Matrix(0, nrow = (n + n), ncol = (n * n))
for (j in 1:n) {
  Amat[j, seq(j, by = n, length.out = n)] = 1
  Amat[j, ((j - 1) * n + j)] = 0
}
for (i in 1:n) {
  Amat[n + i, ((i - 1) * n + 1):(i * n)] = 1
  Amat[n + i, ((i - 1) * n + i)] = 0
}

myLP = list()
myLP$obj = cvec
myLP$A = Amat
myLP$sense = dir
myLP$rhs = bvec
myLP$vtypes = "B"
```

```r
# myLP$vtypes = "C"
# myLP$ub = 1

check = F


mysol = gurobi(myLP)
```

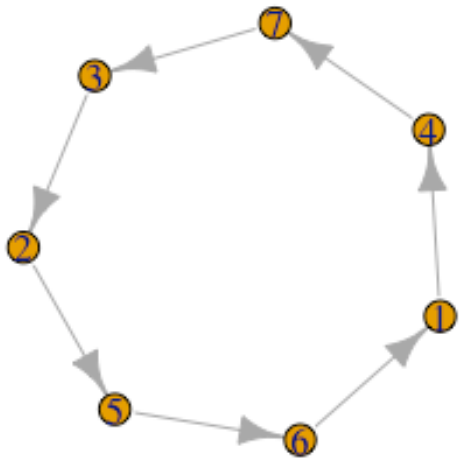
```
## Optimize a model with 14 rows, 49 columns and 84 nonzeros
## Variable types: 0 continuous, 49 integer (49 binary)
## Coefficient statistics:
##   Matrix range     [1e+00, 1e+00]
##   Objective range  [2e+01, 1e+04]
##   Bounds range     [1e+00, 1e+00]
##   RHS range        [1e+00, 1e+00]
## Found heuristic solution: objective 936.0000000
## Presolve removed 0 rows and 7 columns
## Presolve time: 0.00s
## Presolved: 14 rows, 42 columns, 84 nonzeros
## Variable types: 0 continuous, 42 integer (42 binary)
##
## Root relaxation: objective 4.140000e+02, 11 iterations, 0.00 seconds
##
##     Nodes    |    Current Node    |     Objective Bounds      |
Work
##  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap |
It/Node Time
##
## *    0     0               0     414.0000000  414.00000  0.00%     -
0s
##
## Explored 0 nodes (11 simplex iterations) in 0.01 seconds
## Thread count was 4 (of 4 available processors)
##
## Solution count 2: 414 936
##
## Optimal solution found (tolerance 1.00e-04)
## Best objective 4.140000000000e+02, best bound 4.140000000000e+02,
gap 0.0000%
```

```r
x.ij = matrix(mysol$x, nrow = n, ncol = n, byrow = T)

myG = graph_from_adjacency_matrix(x.ij, weighted = T)
plot(myG)
```




```
decomposed.graph = clusters(myG)
if (decomposed.graph$no > 1) {
  for (i in 1:decomposed.graph$no) {
    cities = which(decomposed.graph$membership == i)
    links = t(combn(cities, 2))
    d.ij = matrix(0, n, n)
    for (m in 1:nrow(links)) {
      d.ij[links[m, 1], links[m, 2]] = 1
    }
    d.ij = d.ij + t(d.ij)
    myLP$A = rBind(myLP$A, as.vector(d.ij))
    myLP$rhs = c(myLP$rhs, (length(cities) - 1))
    myLP$sense = c(myLP$sense, "<=")
  }
}
if (decomposed.graph$no == 1) {
  check = T
}
#



while (!check) {
```

```
  params = list(OutputFlag = 0)
  mysol = gurobi(myLP, params)
  x.ij = matrix(mysol$x, nrow = n, ncol = n, byrow = T)

  myG = graph_from_adjacency_matrix(x.ij, weighted = T)

  decomposed.graph = clusters(myG)
  if (decomposed.graph$no > 1) {
    for (i in 1:decomposed.graph$no) {
      cities = which(decomposed.graph$membership == i)
      links = t(combn(cities, 2))
      d.ij = matrix(0, n, n)
      for (m in 1:nrow(links)) {
        d.ij[links[m, 1], links[m, 2]] = 1
      }
      d.ij = d.ij + t(d.ij)
      myLP$A = rBind(myLP$A, as.vector(d.ij))
      myLP$rhs = c(myLP$rhs, (length(cities) - 1))
      myLP$sense = c(myLP$sense, "<=")
    }
  }
  if (decomposed.graph$no == 1) {
    check = T
  }
}
plot(myG)
```


7
3
4
2
1
5
6