PEC4 Operaciones y Logística: Casos de Negocio

Soluciones a los ejercicios propuestos

Febrero de 2019

Pregunta 1

Enunciado

Identifica un problema de decisión multicriterio en vuestra organización. En el caso de no encontrar ninguno, identificar uno genérico.

Expresar el problema definido en la pregunta anterior como un modelo para ejecutar con R. Carga del modelo. Analizar y visualizar los resultados. Comentar resultado final y tomar una decisión. La recomendación es utilizar un fichero AHP de los utilizados en el ejemplo o bien uno de los ejemplos que se pueden obtener ejecutando ahp::RunGUI(). Debéis de justificar las decisiones y la solución dada.

Respuesta

Para este ejemplo se ha escogido como problema la elección de un sistema ERP para la empresa. Después de hacer un estudio de la oferta se han seleccionado cuatro alternativas:

- Microsoft Dynamics
- SAP Business One
- Sage X3
- Unit4 Ekon

Tras varias reuniones discutiendo los criterios, se ha determinado que los más importantes para enfocar la decisión son estos:

- *Coste*: El coste de la solución (que se divide a su vez en el coste de implantación y el coste de las licencias)
- *Funcionalidad*: Las funcionalidades que incorpora de serie la solución
- *Personalización*: Las capacidades que tiene la solución para ser personalizada y ampliada
- *Facilidad de uso*: Facilidad para ser utilizado y puesto en marcha
- *Fortaleza*: De la solución y del fabricante (nivel de implantanción en el mercado, experiencia, partners, soporte,…)

A continuación vamos a cargar el modelo y a visulizar el resultado

```
#knitr::opts_chunk$set(echo = TRUE)


.packages = c("data.tree","ahp","influenceR","rgexf","visNetwork","shinythemes","shinyAce","shinyjs"
,"igraph")
.inst <- .packages %in% installed.packages()
if(length(.packages[!.inst]) > 0)
 suppressPackageStartupMessages(install.packages(.packages[!.inst]))

suppressPackageStartupMessages(library(ahp))
```

```
## Warning: package 'ahp' was built under R version 3.5.2
```

```
ahpFile <- system.file("extdata", "erp.ahp", package="ahp")

cat(readChar(ahpFile, file.info(ahpFile)$size))
```

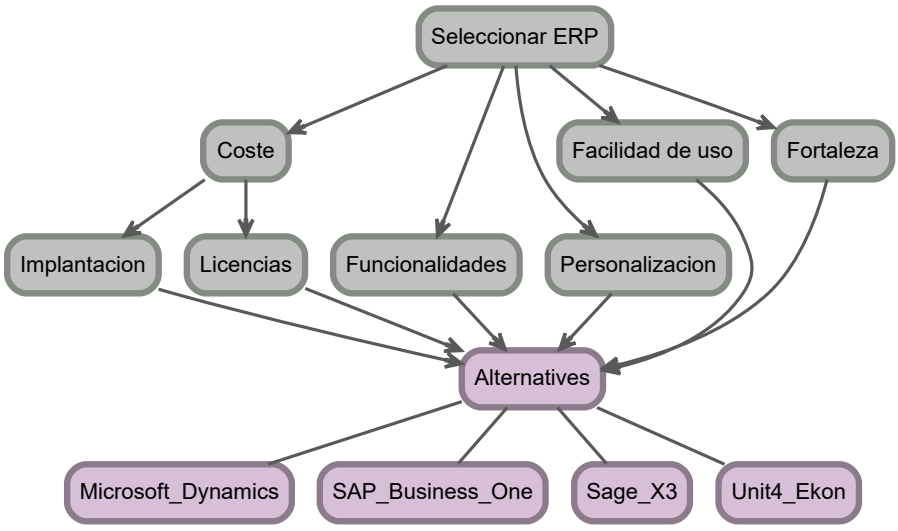
```
## Version: 2.0
##
## ########################
## # Alternatives Section
## #
##
## Alternatives: &alternatives
## # Here, we list all the alternatives, together with their attributes.
## # We can use these attributes later in the file when defining
## # preferenceFunctions. The attributes can be quantitative or
## # qualitative.
##   Microsoft_Dynamics:
##     coste: 145000
##     porcentaje_licencias: 0.10
##     funcionalidad: 8
##     personalizacion: 6
##     partners: 825
##     cloud: Sí
##
##   SAP_Business_One:
##     coste: 163000
##     porcentaje_licencias: 0.09
##     funcionalidad: 9
##     personalizacion: 4
##     partners: 708
##     cloud: Sí
##
##   Sage_X3:
##     coste: 115000
##     porcentaje_licencias: 0.11
##     funcionalidad: 7
##     personalizacion: 6
##     partners: 25
##     cloud: No
##
##   Unit4_Ekon:
##     coste: 130000
##     porcentaje_licencias: 0.15
##     funcionalidad: 6
##     personalizacion: 8
##     partners: 43
##     cloud: No
##
## #
## # End of Alternatives Section
## ####################################
##
## ####################################
## # Goal Section
## #
##
##
## Goal:
## # The goal spans a tree of criteria and the alternatives
##   name: Seleccionar ERP
##   description: >
##     Modelo del problema de selección de un ERP
##   author: Marcos Pereiro
##   preferences:
##     # preferences are typically defined pairwise
##     # 1 means: A is equal to B
##     # 9 means: A is highly preferrable to B
##     # 1/9 means: B is highly preferrable to A
##     pairwise:
##       - [Coste, Funcionalidades, 3]
##       - [Coste, Personalizacion, 5]
##       - [Coste, Facilidad de uso, 7]
##       - [Coste, Fortaleza, 8]
##       - [Funcionalidades, Personalizacion, 4]
##       - [Funcionalidades, Facilidad de uso, 7]
##       - [Funcionalidades, Fortaleza, 8]
##       - [Personalización, Facilidad de uso, 4]
##       - [Personalización, Fortaleza, 5]
##       - [Facilidad de uso, Fortaleza, 2]
##
##   children:
##     Coste:
##       preferences:
##         pairwise:
##           - [Implantacion, Licencias, 3]
##
##       children:
##         Implantacion:
##           preferences:
##             pairwise:
##               - [Microsoft_Dynamics, SAP_Business_One, 1/3]
```

```
##               - [Microsoft_Dynamics, Sage_X3, 5]
##               - [Microsoft_Dynamics, Unit4_Ekon, 7]
##               - [SAP_Business_One, Sage_X3, 7]
##               - [SAP_Business_One, Unit4_Ekon,9 ]
##               - [Sage_X3, Unit4_Ekon, 1]
##
##           children: *alternatives
##
##         Licencias:
##
##           preferences:
##             pairwiseFunction:
##               function(a1, a2) min(9, max(1/9, (a2$coste*a2$porcentaje_licencias)/(a1$coste*a1$porcentaje_licencias)))
##
##           children: *alternatives
##
##     Funcionalidades:
##       preferences:
##         scoreFunction:
##             function(a) a$funcionalidad
##       children: *alternatives
##
##     Personalizacion:
##       preferences:
##         pairwise:
##             - [Microsoft_Dynamics, SAP_Business_One, 5]
##             - [Microsoft_Dynamics, Sage_X3, 3]
##             - [Microsoft_Dynamics, Unit4_Ekon, 1/3]
##             - [SAP_Business_One, Sage_X3, 1/3]
##             - [SAP_Business_One, Unit4_Ekon, 1/5]
##             - [Sage_X3, Unit4_Ekon, 1/3]
##       children: *alternatives
##
##     Facilidad de uso:
##       preferences:
##         pairwise:
##             - [Microsoft_Dynamics, SAP_Business_One, 3]
##             - [Microsoft_Dynamics, Sage_X3, 5]
##             - [Microsoft_Dynamics, Unit4_Ekon, 7]
##             - [SAP_Business_One, Sage_X3, 3]
##             - [SAP_Business_One, Unit4_Ekon, 5]
##             - [Sage_X3, Unit4_Ekon, 3]
##       children: *alternatives
##
##     Fortaleza:
##       preferences:
##         pairwise:
##             - [Microsoft_Dynamics, SAP_Business_One, 1]
##             - [Microsoft_Dynamics, Sage_X3, 7]
##             - [Microsoft_Dynamics, Unit4_Ekon, 9]
##             - [SAP_Business_One, Sage_X3, 7]
##             - [SAP_Business_One, Unit4_Ekon, 9]
##             - [Sage_X3, Unit4_Ekon, 3]
##       children: *alternatives
##
## #
## # End of Goal Section
## ####################################
```

```
Ahp <- Load(ahpFile)
Ahp
```

```
##                        levelName
## 1  Seleccionar ERP
## 2   ¦--Coste
## 3   ¦   ¦--Implantacion
## 4   ¦   ¦   ¦--Microsoft_Dynamics
## 5   ¦   ¦   ¦--SAP_Business_One
## 6   ¦   ¦   ¦--Sage_X3
## 7   ¦   ¦   °--Unit4_Ekon
## 8   ¦   °--Licencias
## 9   ¦       ¦--Microsoft_Dynamics
## 10  ¦       ¦--SAP_Business_One
## 11  ¦       ¦--Sage_X3
## 12  ¦       °--Unit4_Ekon
## 13  ¦--Funcionalidades
## 14  ¦   ¦--Microsoft_Dynamics
## 15  ¦   ¦--SAP_Business_One
## 16  ¦   ¦--Sage_X3
## 17  ¦   °--Unit4_Ekon
## 18  ¦--Personalizacion
## 19  ¦   ¦--Microsoft_Dynamics
## 20  ¦   ¦--SAP_Business_One
## 21  ¦   ¦--Sage_X3
## 22  ¦   °--Unit4_Ekon
## 23  ¦--Facilidad de uso
## 24  ¦   ¦--Microsoft_Dynamics
## 25  ¦   ¦--SAP_Business_One
## 26  ¦   ¦--Sage_X3
## 27  ¦   °--Unit4_Ekon
## 28  °--Fortaleza
## 29      ¦--Microsoft_Dynamics
## 30      ¦--SAP_Business_One
## 31      ¦--Sage_X3
## 32      °--Unit4_Ekon
```

```
Calculate(Ahp)
Visualize(Ahp, criteriaNodesStyle = list(style = "filled,rounded", shape =
 "box", color = "honeydew4", fillcolor = "grey", penwidth = 4, fontname =
 "helvetica"), alternativeNodesStyle = list(style = "filled,rounded", shape =
 "box", color = "thistle4", fillcolor = "thistle", penwidth = 4, fontname =
 "helvetica"), criteriaEdgesStyle = list(arrowhead = "vee", color = "grey35",
 penwidth = 2), alternativeEdgesStyle = list(dir = "none", color = "grey35",
 penwidth = 2))
```




```
AnalyzeTable(Ahp)
```

	Weight	SAP_Business_One	Microsoft_Dynamics	Unit4_Ekon	Sage_X3	Inconsistency
Seleccionar ERP	79.6%	30.0%	24.4%	12.8%	12.4%	8.0%
Coste	37.4%	18.9%	10.6%	3.3%	4.6%	0.0%
Implantacion	28.0%	16.5%	8.2%	1.5%	1.8%	3.5%
Licencias	9.3%	2.4%	2.4%	1.8%	2.8%	0.0%
Funcionalidades	24.7%	7.4%	6.6%	4.9%	5.8%	NA
Personalizacion	7.9%	0.5%	2.3%	4.0%	1.1%	7.5%

	Weight	SAP_Business_One	Microsoft_Dynamics	Unit4_Ekon	Sage_X3	Inconsistency
Facilidad de uso	5.5%	1.4%	3.1%	0.3%	0.6%	4.4%
Fortaleza	4.1%	1.8%	1.8%	0.2%	0.3%	3.4%

A la vista de los resultados y dando por hecho que los criterios se han definido medianamente bien y han sido consensuados, podemos deducir que nuestra elección debería ser “SAP Business One”, ya que esta solución es la que presenta un mejor compromiso entre todos los criterios, destacando en el precio y en las funcionalidades y con puntuaciones altas en el resto de apartados.

Si llegados a este punto todavía surgieran importantes reticencias quizás significa que los criterios deben ser revisados. El análisis presente en este tabla puede ayudar a valorar si ha habido alguno que ha sido mal ponderado o si sobran o faltan otros criterios.

Pregunta 2

Enunciado

La compañía de retail (que hemos estado analizando durante este curso) está planificando abrir una nueva tienda y está sondeando el mercado. Ha realizado un estudio y ha obtenido algunos datos que desea utilizar como tabla de decisión para tener éxito en su elección.

Establece los criterios de decisión y toma AHP para la toma de la decisión final. Genera la el fichero ahp Visualiza el modelo resultante Analiza las opciones del resultado

Respuesta

Tras el estudio realizado se han escogido las cuatro variables remitidas en el enunciado, potencial, cuota, competideres y renta, y los criterios que se van a valorar para tomar las decisiones.

Así, establecemos como criterios de decisión en el siguiente orden para determinar la ubicación de la tienda:

En este caso los criterios de decisión son:

- Ganacia potencial
- Cuota de mercado
- Tiendas competidoras
- Renta per càpita

Se crea el fichero ahp que se muestra a continuación en el recuadro, y se carga mediante una función Load

```
#Cargamos el fichero "retail.aph"
```

```
Version: 2.0
########################
# Alternatives Section
#
Alternatives: &alternatives
# 1= not well; 10 = best possible
  Salamanca:
    ganancia: 390000
    cuota: 30.3
    competidoras: 3
    renta: 10
  Burgos:
    ganancia: 250000
    cuota: 15
    competidoras: 2
    renta: 6
  Leon:
    ganancia: 410000
    cuota: 28
    competidoras: 3
    renta: 6
  Granada:
    ganancia: 600000
    cuota: 40
    competidoras: 5
    renta: 10
  Badajoz:
    ganancia: 300000
    cuota: 20
    competidoras: 2
    renta: 6
  Malaga:
    ganancia: 540000
    cuota: 35
    competidoras: 3
    renta: 6
#
# End of Alternatives Section
####################################
# Goal Section
#
Goal:
# A Goal HAS preferences (within-Level comparison) and HAS Children (items in level)
  name: Nueva tienda a abrir
  preferences:
    pairwise:
      # preferences are defined pairwise
      # 1 means: A is equal to B
      # 9 means: A is highly preferrable to B
      # 1/9 means: B is highly preferrable to A
      - [Cuota, Competidoras, 4]
      - [Cuota, Renta, 7]
      - [Cuota, Ganancia, 1/3]
      - [Competidoras, Renta, 3]
      - [Competidoras, Ganancia, 1/3]
      - [Ganancia, Renta, 5]
  children:
    Cuota:
      preferences:
        pairwiseFunction: >
          CuotaPreference <- function(a1, a2) {
            if (a1$cuota < a2$cuota) return (1/CuotaPreference(a2, a1))
            ratio <- a1$cuota / a2$cuota
            if (ratio < 1.05) return (1)
            if (ratio < 1.2) return (2)
            if (ratio < 1.5) return (3)
            if (ratio < 1.8) return (4)
            if (ratio < 2.1) return (5)
            return (6)
          }
      children: *alternatives
    Competidoras:
      preferences:
        scoreFunction: function(a) 15-3*a$competidoras
      children: *alternatives
    Renta:
      preferences:
        pairwiseFunction: >
          RentaPreference <- function(a1, a2) {
            if (a1$renta < a2$renta) return (1/RentaPreference(a2, a1))
            ratio <- a1$renta / a2$renta
            if (ratio < 1) return (1)
            if (ratio < 1.05) return (2)
            if (ratio < 1.1) return (3)
            if (ratio < 1.2) return (4)
            if (ratio < 1.3) return (5)
```

```
            return (6)
          }
      children: *alternatives
    Ganancia:
      preferences:
        pairwiseFunction: >
          GananciaPreference <- function(a1, a2) {
            if (a1$ganancia < a2$ganancia) return (1/GananciaPreference(a2, a1))
            ratio <- a1$ganancia / a2$ganancia
            if (ratio < 1.05) return (1)
            if (ratio < 1.2) return (2)
            if (ratio < 1.5) return (3)
            if (ratio < 1.8) return (4)
            if (ratio < 2.1) return (5)
            return (6)
          }
      children: *alternatives
#
# End of Goal Section
###################################  - [Granada, Malaga, 4]
          - [Badajoz, Malaga, 4]
      children: *alternatives
#
# End of Goal Section
################################### ion
################################### lternatives
#
# End of Goal Section
###################################
```

```
#Revisamos su definición
retailAph <- Load("retail.ahp")
retailAph
```

```
##               levelName
## 1  Nueva tienda a abrir
## 2    ¦--Cuota
## 3    ¦   ¦--Salamanca
## 4    ¦   ¦--Burgos
## 5    ¦   ¦--Leon
## 6    ¦   ¦--Granada
## 7    ¦   ¦--Badajoz
## 8    ¦   °--Malaga
## 9    ¦--Competidoras
## 10   ¦   ¦--Salamanca
## 11   ¦   ¦--Burgos
## 12   ¦   ¦--Leon
## 13   ¦   ¦--Granada
## 14   ¦   ¦--Badajoz
## 15   ¦   °--Malaga
## 16   ¦--Renta
## 17   ¦   ¦--Salamanca
## 18   ¦   ¦--Burgos
## 19   ¦   ¦--Leon
## 20   ¦   ¦--Granada
## 21   ¦   ¦--Badajoz
## 22   ¦   °--Malaga
## 23   °--Ganancia
## 24       ¦--Salamanca
## 25       ¦--Burgos
## 26       ¦--Leon
## 27       ¦--Granada
## 28       ¦--Badajoz
## 29       °--Malaga
```

Puedo ejecutar el modelo:

```
Calculate(retailAph)
```

Y visualizar el modelo resultante:

```
Visualize(retailAph,criteriaNodesStyle = list(style = "filled,rounded", shape =
  "box", color = "honeydew4", fillcolor = "grey", penwidth = 4, fontname =
  "helvetica"), alternativeNodesStyle = list(style = "filled,rounded", shape =
  "box", color = "thistle4", fillcolor = "thistle", penwidth = 4, fontname =
  "helvetica"), criteriaEdgesStyle = list(arrowhead = "vee", color = "grey35",
  penwidth = 2), alternativeEdgesStyle = list(dir = "none", color = "grey35",
  penwidth = 2))
```

El análisis nos proporciona:

```
Analyze(retailAph)
```

```
##                       Weight Granada Malaga Salamanca  Leon Badajoz
## 1 Nueva tienda a abrir 100.0%   31.7%  24.3%     15.5% 13.6%    8.4%
## 2  ¦--Ganancia          49.4%   18.3%  13.8%      5.5%  7.1%    3.0%
## 3  ¦--Cuota             32.3%   11.6%   8.2%      5.5%  3.9%    2.0%
## 4  ¦--Competidoras      12.6%    0.0%   2.1%      2.1%  2.1%    3.1%
## 5  °--Renta              5.7%    1.9%   0.2%      2.4%  0.4%    0.3%
##   Burgos Inconsistency
## 1   6.5%         11.3%
## 2   1.8%          4.7%
## 3   1.1%          4.5%
## 4   3.1%            NA
## 5   0.5%          4.3%
```

Y se puede mejorar el formato mediante AnalyzeTable:

```
AnalyzeTable(retailAph)
```

	Weight	Granada	Malaga	Salamanca	Leon	Badajoz	Burgos	Inconsistency
Nueva tienda a abrir	100.0%	31.7%	24.3%	15.5%	13.6%	8.4%	6.5%	❗ 11.3%
Ganancia	49.4%	18.3%	13.8%	5.5%	7.1%	3.0%	1.8%	4.7%
Cuota	32.3%	11.6%	8.2%	5.5%	3.9%	2.0%	1.1%	4.5%
Competidoras	12.6%	0.0%	2.1%	2.1%	2.1%	3.1%	3.1%	NA
Renta	5.7%	1.9%	0.2%	2.4%	0.4%	0.3%	0.5%	4.3%

```
AnalyzeTable(retailAph,
             variable = "priority",
             sort = "orig",
             pruneFun = function(node, decisionMaker) PruneByCutoff(node, decisionMaker, 0.005),
             weightColor = "skyblue",
             consistencyColor = "red",
             alternativeColor = "green")
```

	Priority	Salamanca	Burgos	Leon	Granada	Badajoz	Malaga	Inconsistency
Nueva tienda a abrir	100.0%							❗ 11.3%
Cuota	32.3%	17.1%	3.4%	12.2%	35.9%	6.1%	25.3%	4.5%
Competidoras	12.6%	16.7%	25.0%	16.7%	0.0%	25.0%	16.7%	NA
Renta	5.7%	41.5%	8.7%	6.9%	33.0%	5.5%	4.4%	4.3%
Ganancia	49.4%	11.1%	3.6%	14.5%	36.9%	6.1%	27.8%	4.7%

El resultado nos muestra la ciudad de Granada como la que obtiene mejor nota en todos los criterios. A continuación quedan Málaga y Salamanca.

Pregunta 3

Enunciado

Una compañía de reparto se ha creado este año con un crecimiento estimado de 1 nueva ciudad al mes en su línea de reparto. Se proporciona el fichero Cities. La ruta siempre se inicia de la primera ciudad (con indicador 0 y que corresponde a los almacenes principales). Identificar la ruta óptima para los siguientes tres años.

Respuesta

Según se indica en el enunciado se nos solicita encontrar la ruta óptima, siempre considerando el inicio en la ciudad 0. Dado el número de ciudades que hay, usaremos la libreria TSP

```
# Librerías
.packages = c("TSP")

# Validación
.inst <- .packages %in% installed.packages()
if(length(.packages[!.inst]) > 0)
  suppressPackageStartupMessages(install.packages(.packages[!.inst]))

# Carga
suppressPackageStartupMessages(library("TSP"))
```

Procedemos a cargar los datos.

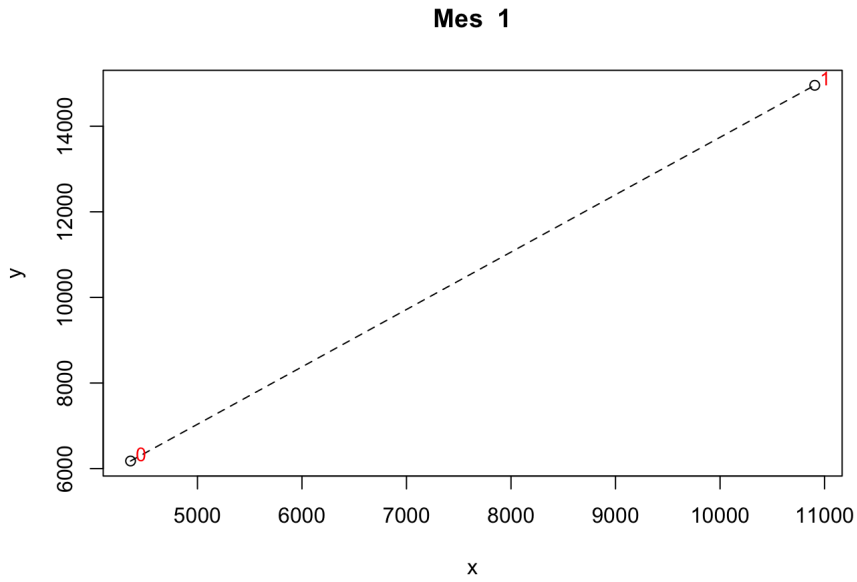
```
ruta <- read.csv("cities.csv")
head(ruta)
```

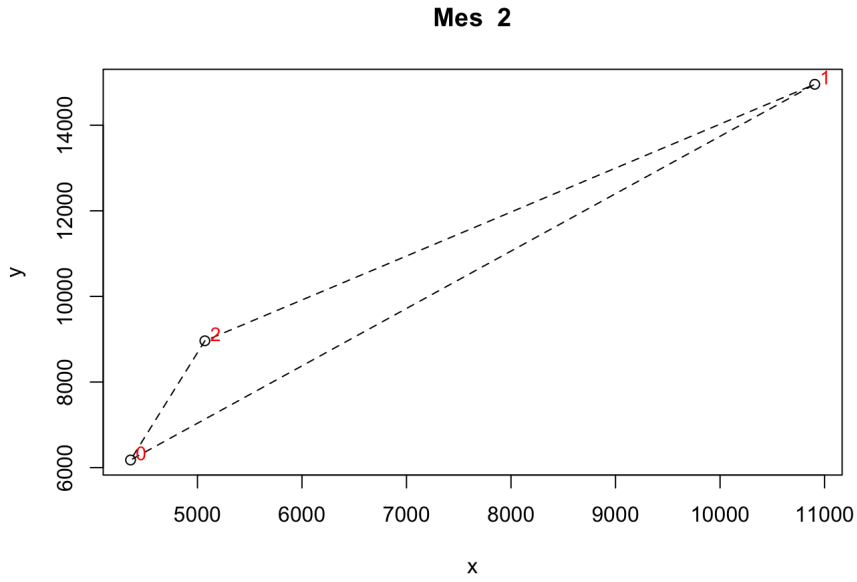
Se observa que tenemos un identificador de ciudad y luego las coordenadas x, y.

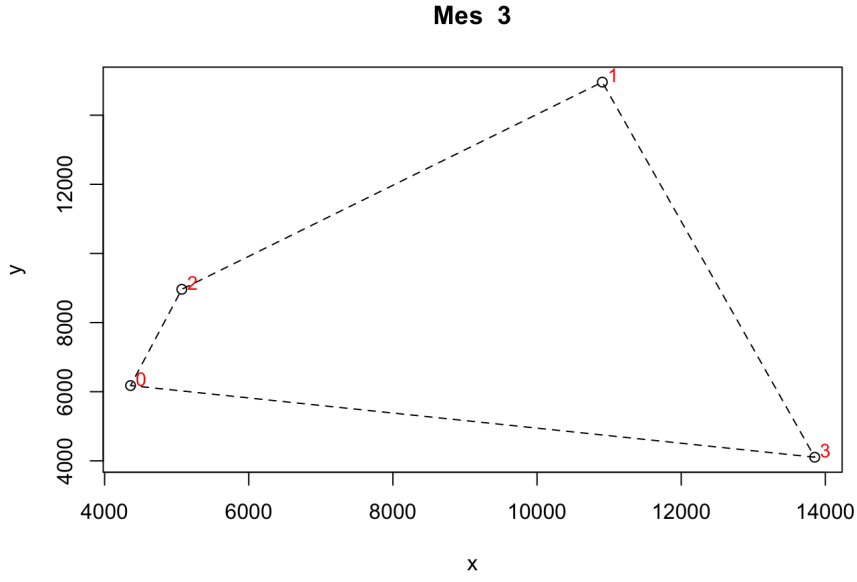
A continuación, vamos a calcular la ruta óptima según las condiciones del enunciado: - suponemos una nueva ciudad al mes. - se quieres calcular para los siguientes 3 años.

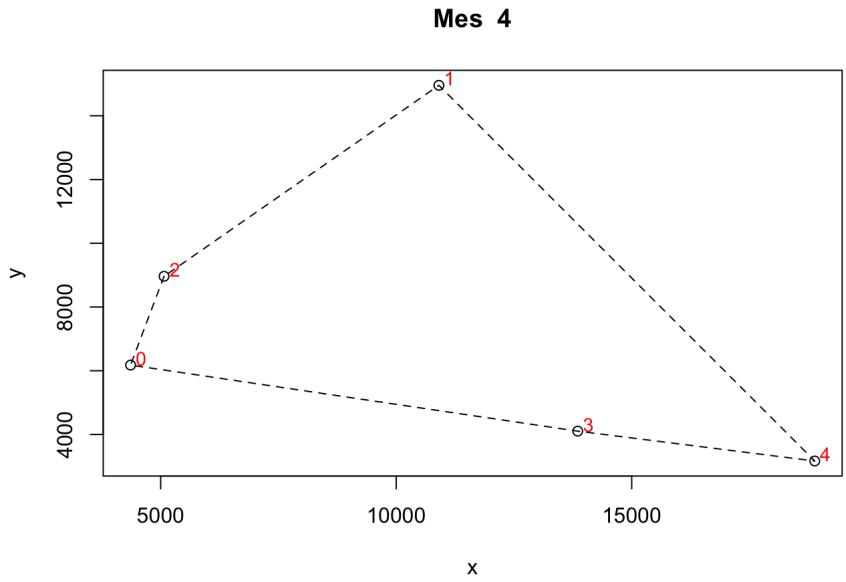
En este sentido, vamos a calcular las rutas mes a mes, añadiendo cada mes una nueva ciudad.

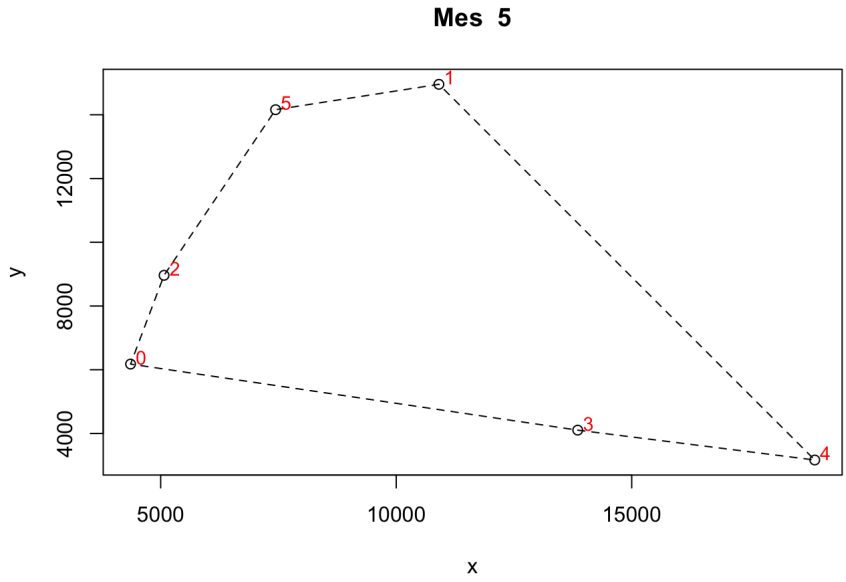
```
mes <- 36
final<-FALSE
j<-2
while(!final){
  ruta2<-ruta[1:(j),]
  ruta3<-ruta2[c(2,3)]
  etsp <- ETSP(ruta3)
  tour <- solve_TSP(etsp)
  tour
  tour_length<-tour_length(tour)
  plot(etsp, tour,main=paste("Mes ",j-1))
  text(ruta2[,2],ruta2[,3], labels=as.character(paste("",ruta2[,1])), pch=9, cex=0.9,col="red",adj = c(0,0))
  if(j>mes){
     final<-TRUE
  }
  if(tour_length>150000){
     final<-TRUE
  }
  j<-j+1
}
```

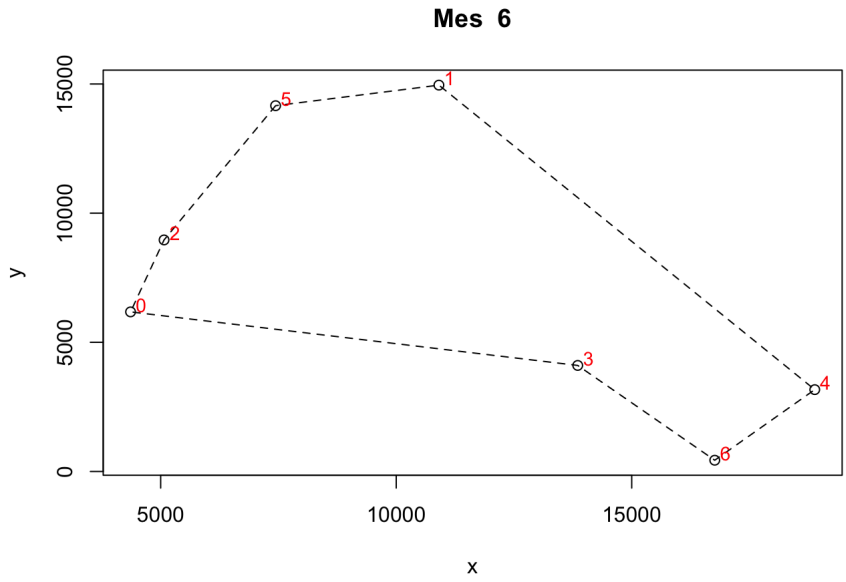

Mes 1
y
x
6000
8000
10000
12000
14000
5000
6000
7000
8000
9000
10000
11000
0
1

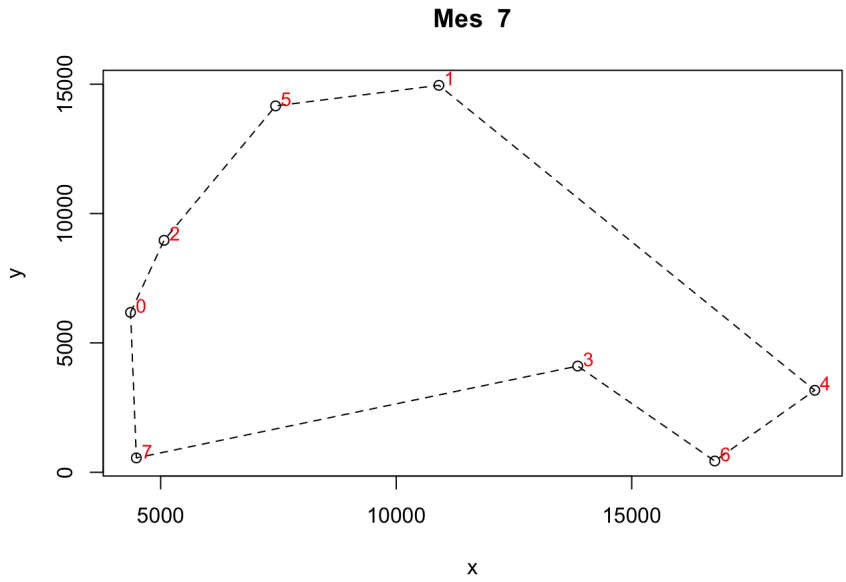

Mes 2
y
x
6000
8000
10000
12000
14000
5000
6000
7000
8000
9000
10000
11000
0
1
2

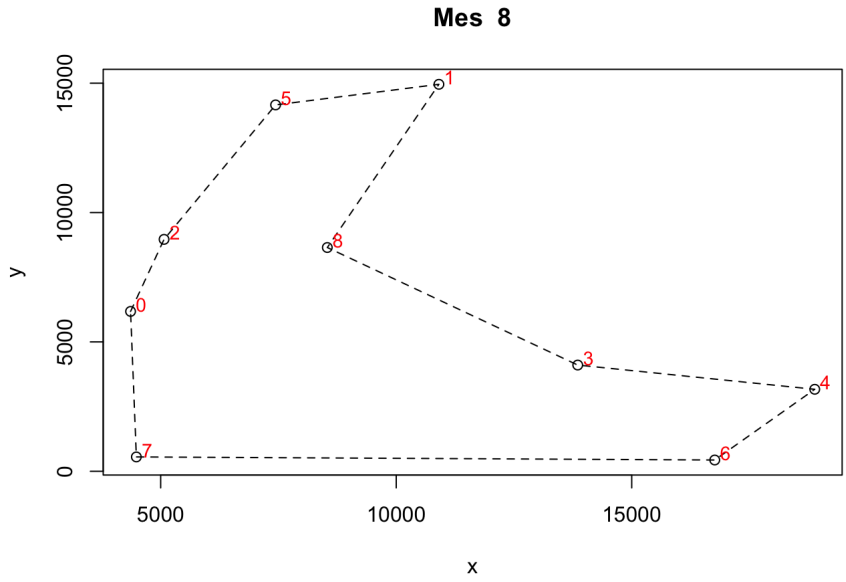

Mes 3
y
x
4000
6000
8000
12000
4000
6000
8000
10000
12000
14000
0
1
2
3

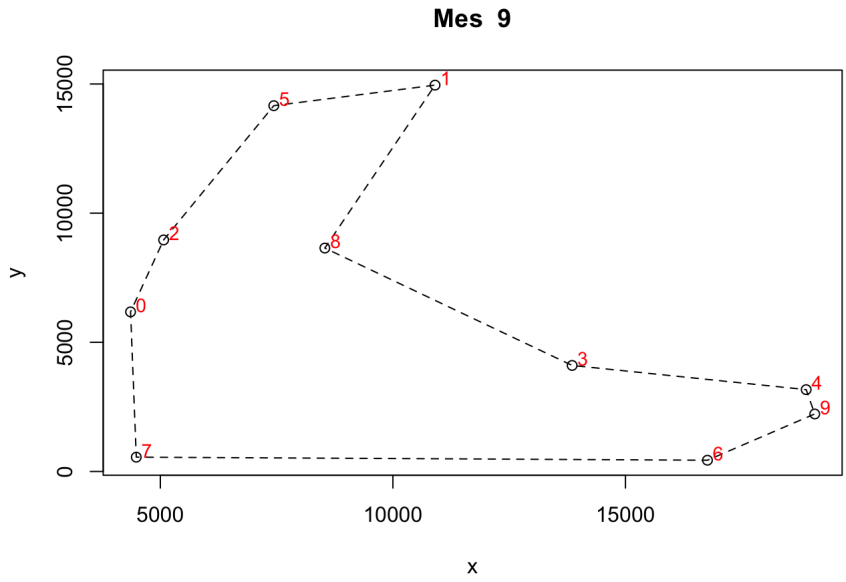

Mes 4
y
x
4000
8000
12000
5000
10000
15000
0
1
2
3
4

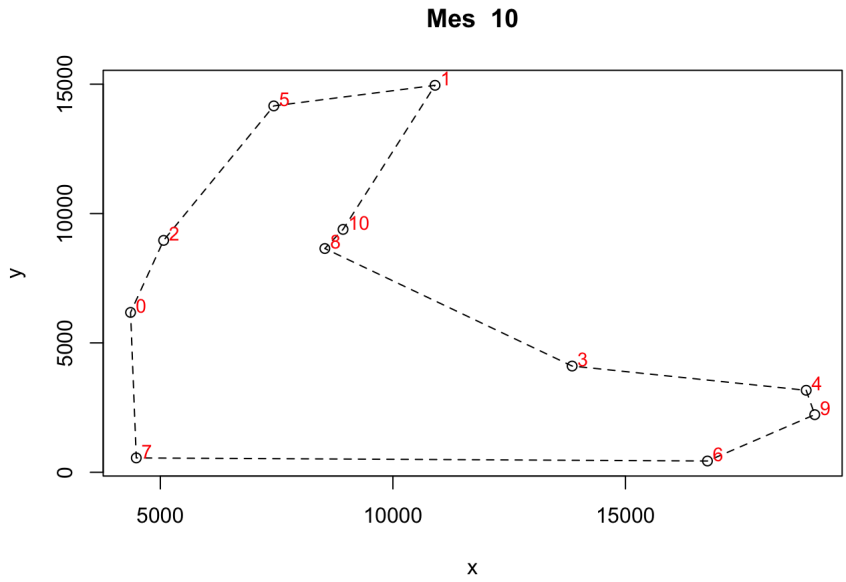

Mes 5
y
x
4000
8000
12000
5000
10000
15000
0
1
2
3
4
5

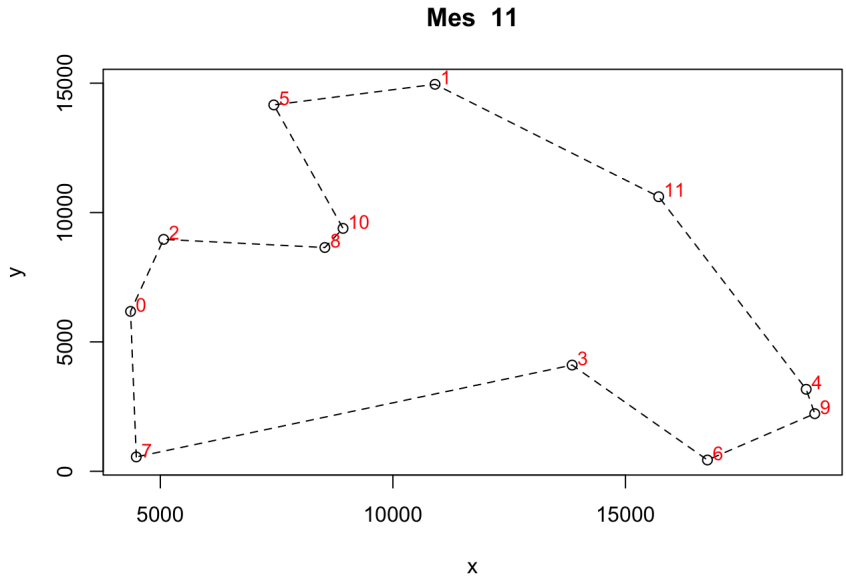

Mes 6
y
x
0
5000
10000
15000
5000
10000
15000
0
1
2
3
4
5
6

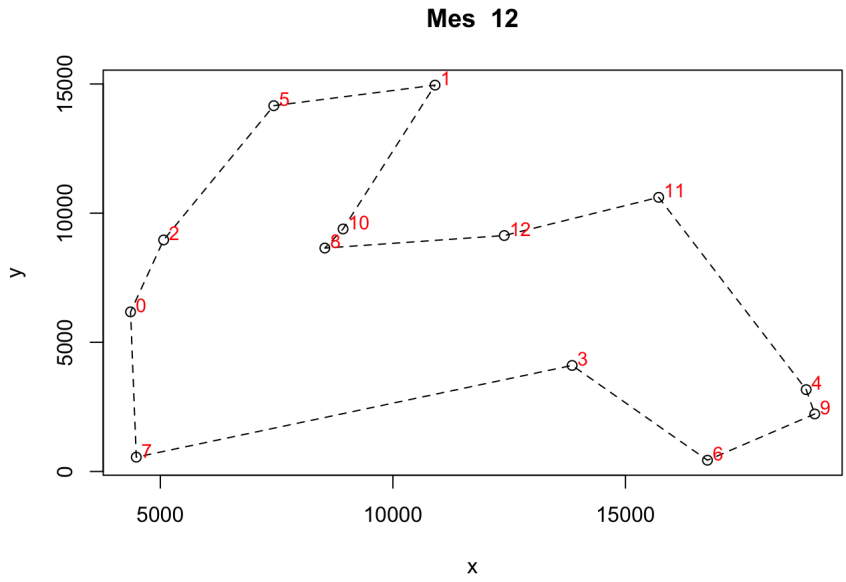

Mes 7
y
x
0
5000
10000
15000
0
1
2
3
4
5
6
7

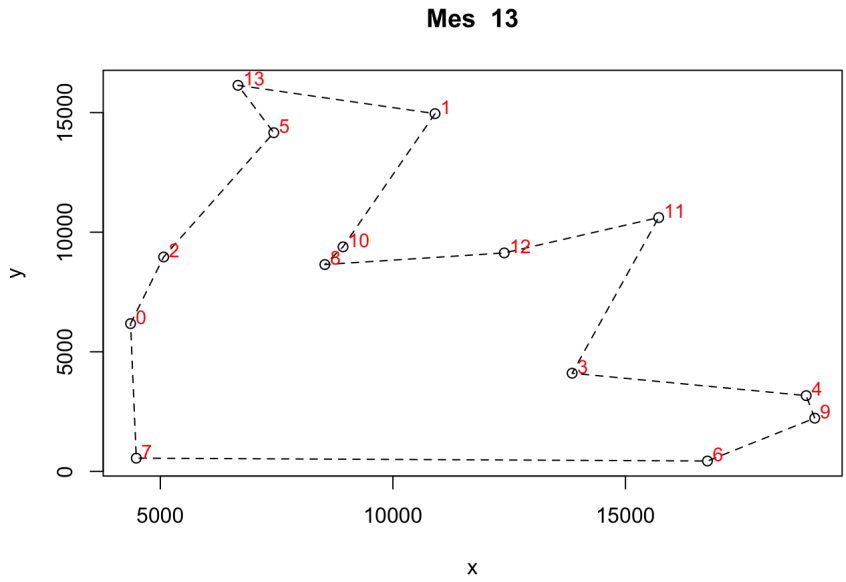

Mes 8
y
x
0
5000
10000
15000
0
1
2
3
4
5
6
7
8

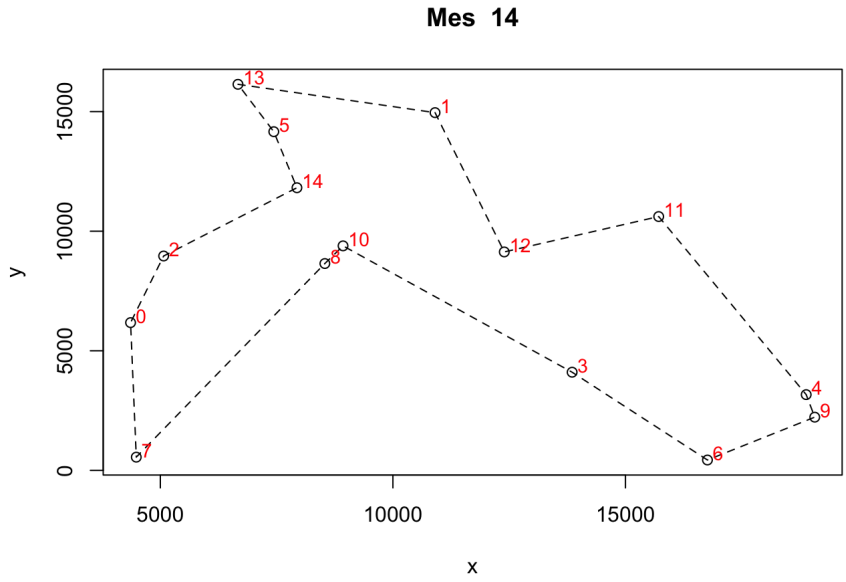

Mes 9
y
x
0
5000
10000
15000
0
1
2
3
4
5
6
7
8
9

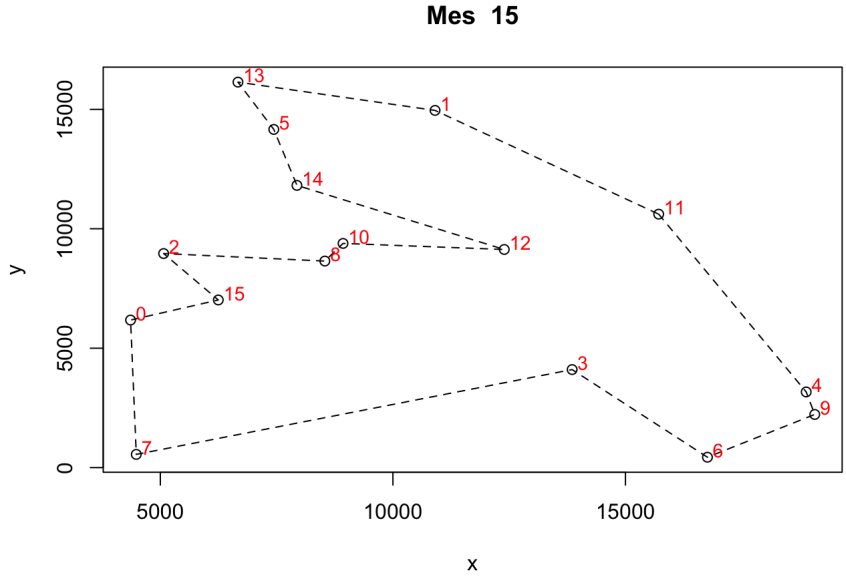

Mes 10
y
x
0
5000
10000
15000
0
1
2
3
4
5
6
7
8
9
10

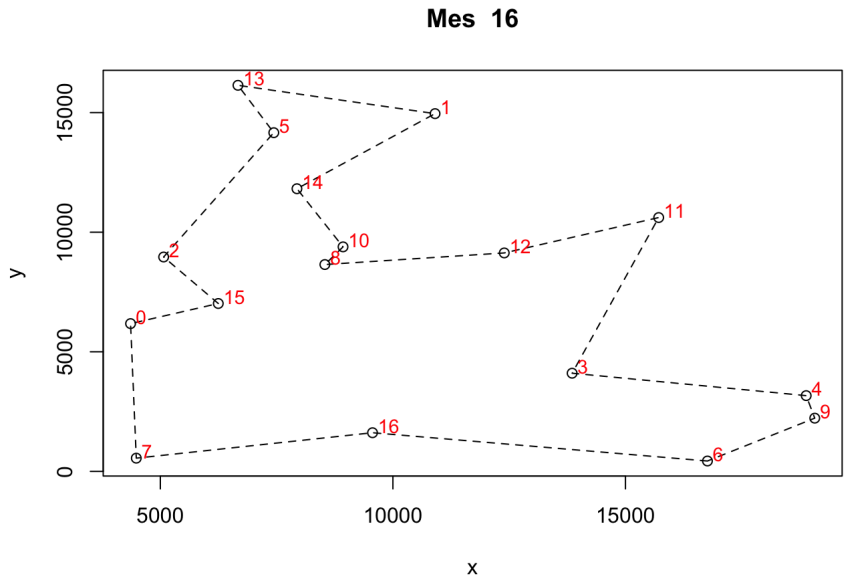

Mes 11
y
x
0
5000
10000
15000
0
1
2
3
4
5
6
7
8
9
10
11

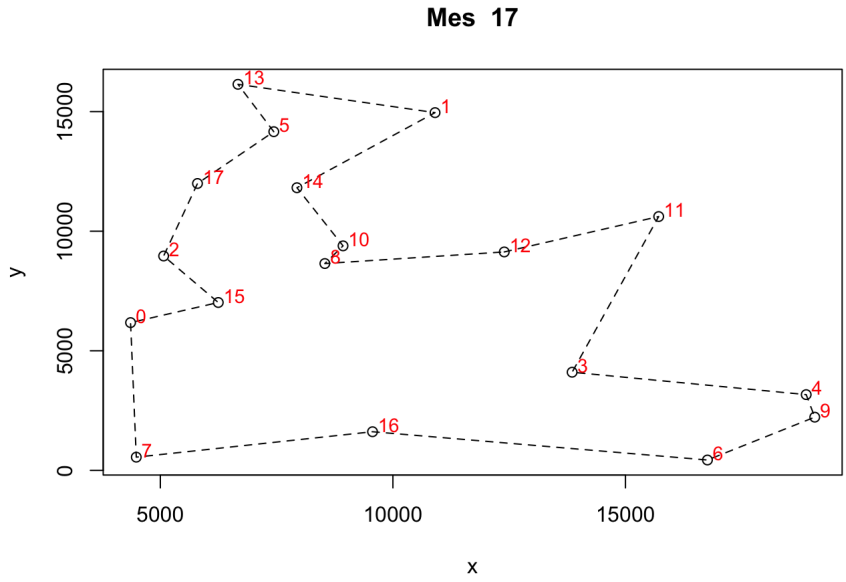

Mes 12
y
x
0
5000
10000
15000
0
1
2
3
4
5
6
7
8
9
10
11
12

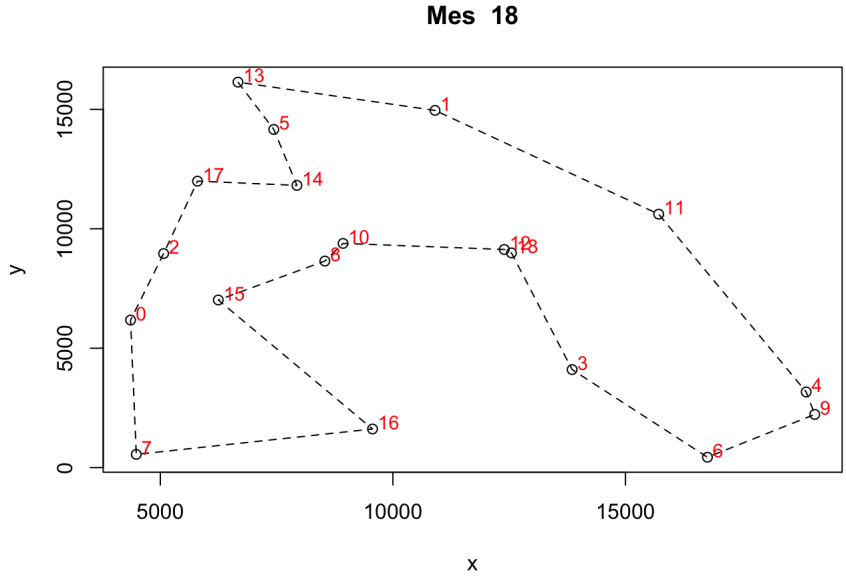

Mes 13
13
1
5
11
10
8
12
2
0
3
4
9
7
6
y
x
0
5000
10000
15000

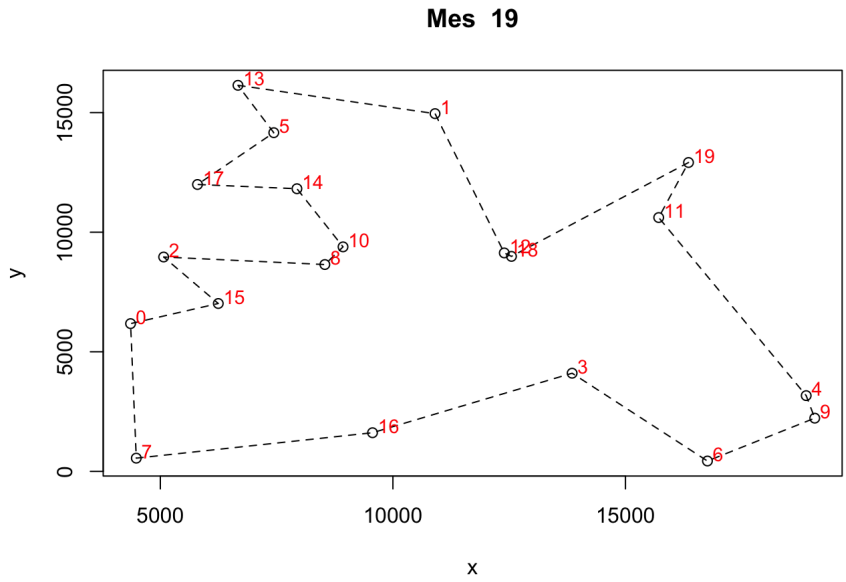

Mes 14
13
1
5
14
11
12
2
10
8
0
3
4
9
7
6
y
x
0
5000
10000
15000

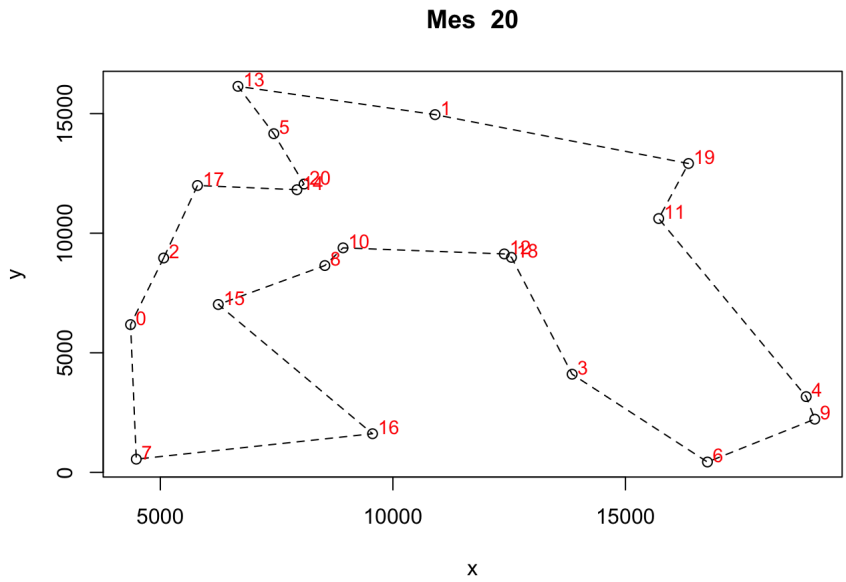

Mes 15
13
1
5
14
11
2
10
8
12
15
0
3
4
9
7
6
y
x
0
5000
10000
15000

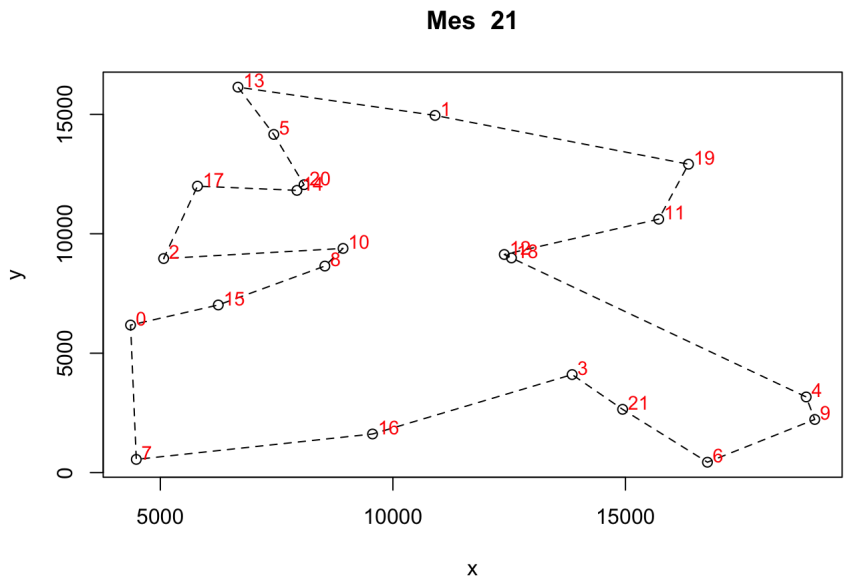

Mes 16
y
x
0
5000
10000
15000
13
1
5
14
11
10
8
12
2
15
0
3
4
9
16
7
6

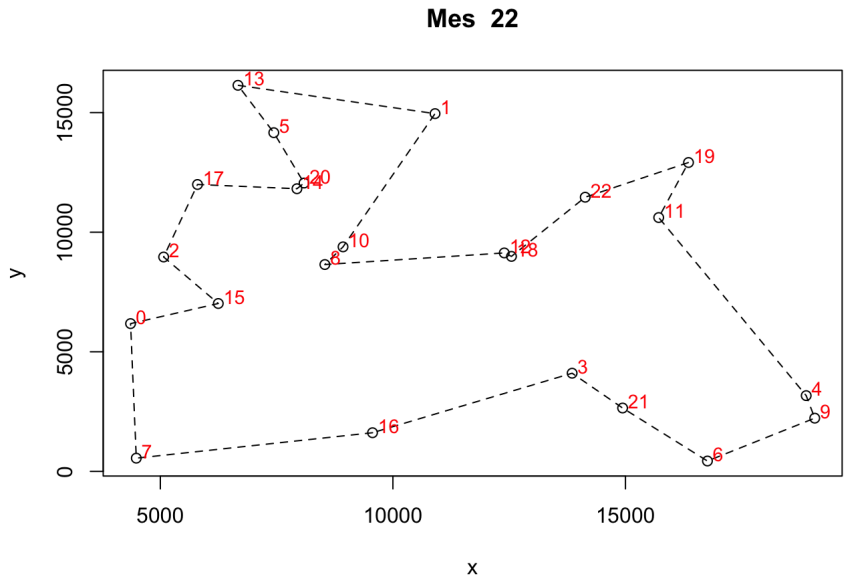

Mes 17
y
x
0
5000
10000
15000
13
1
5
17
14
11
10
8
12
2
15
0
3
4
9
16
7
6

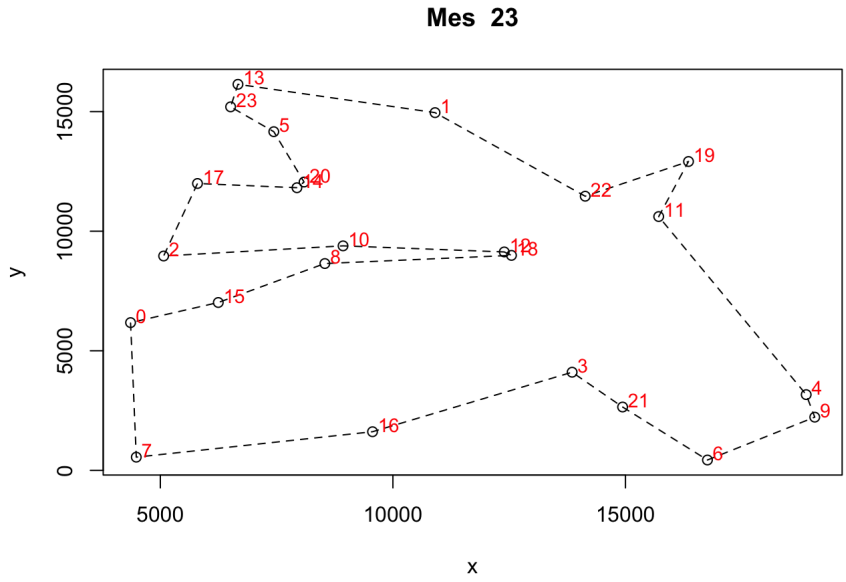

Mes 18
y
x
0
5000
10000
15000
13
1
5
17
14
11
10
8
18
2
15
0
3
4
9
16
7
6

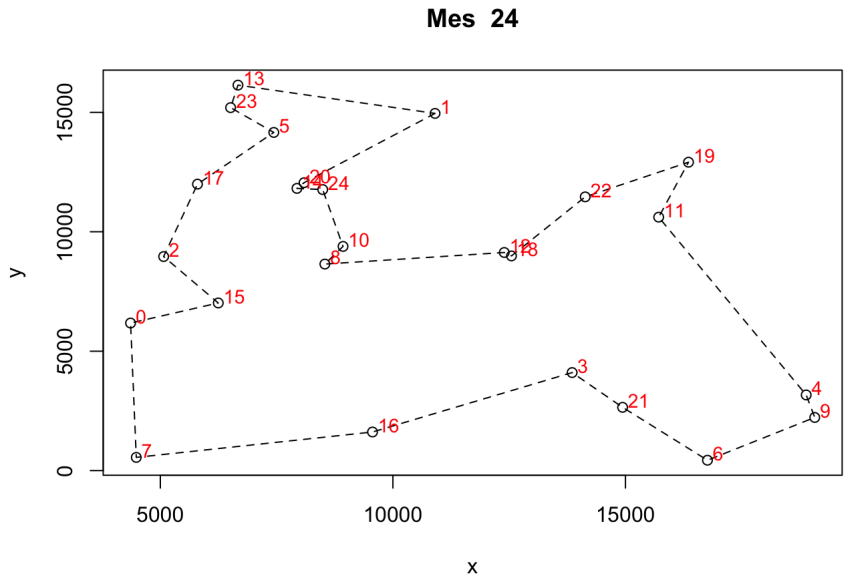

Mes 19
y
x
0
5000
10000
15000
5000
10000
15000
13
1
5
19
17
14
11
10
2
8
15
0
3
4
9
16
7
6

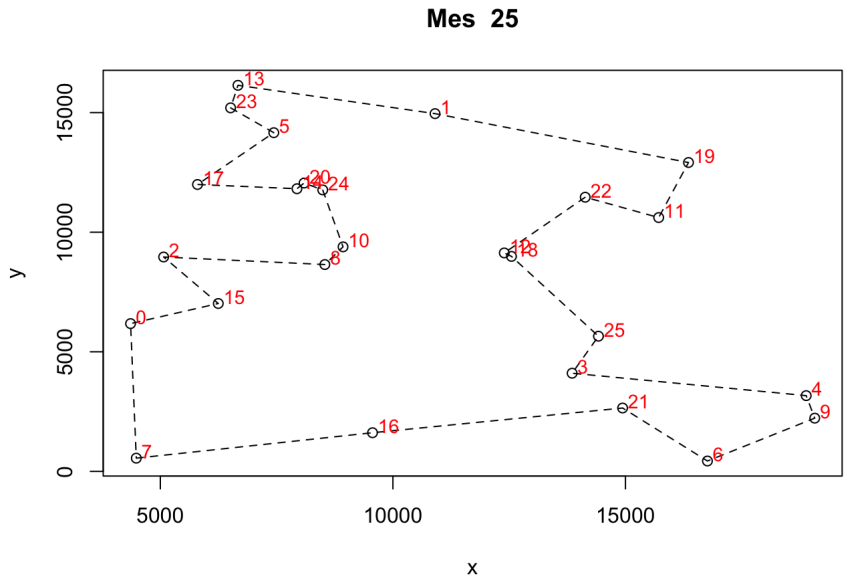

Mes 20
y
x
0
5000
10000
15000
5000
10000
15000
13
1
5
19
17
20
11
10
2
8
15
0
3
4
9
16
7
6

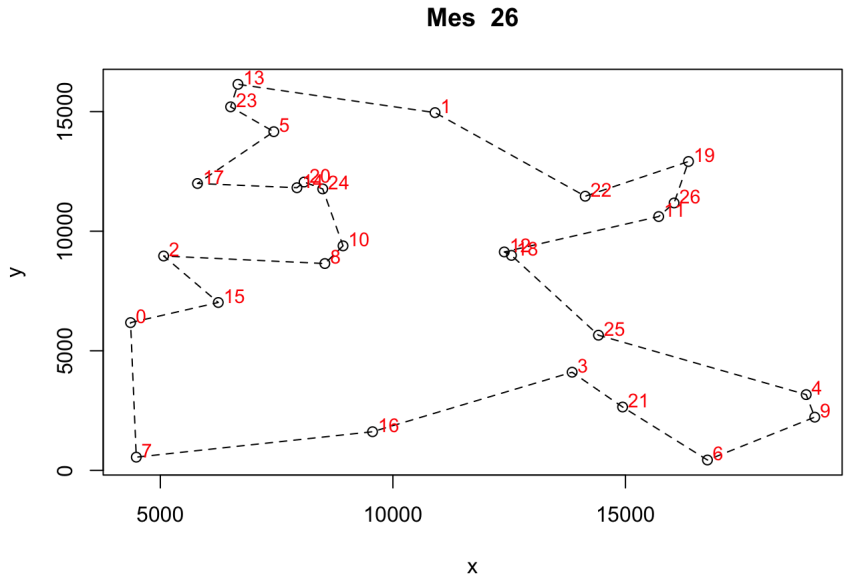

Mes 21
y
x
0
5000
10000
15000
5000
10000
15000
13
1
5
19
17
20
11
10
2
8
15
0
3
21
4
9
16
7
6

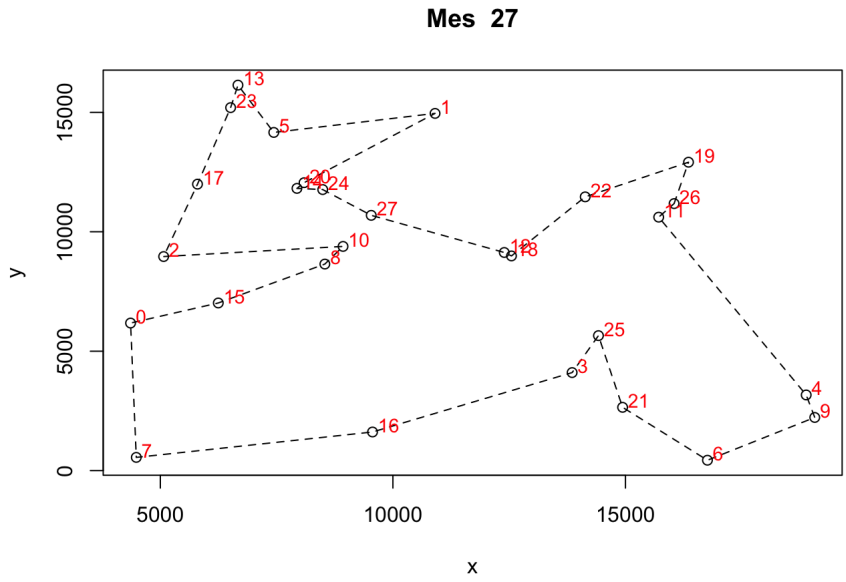

Mes 22
y
x
0
5000
10000
15000
5000
10000
15000
13
1
5
17
20
19
22
11
2
10
15
0
3
21
4
9
16
7
6


Mes 23
y
x
0
5000
10000
15000
5000
10000
15000
13
23
1
5
17
19
22
11
2
10
15
0
3
21
4
9
16
7
6

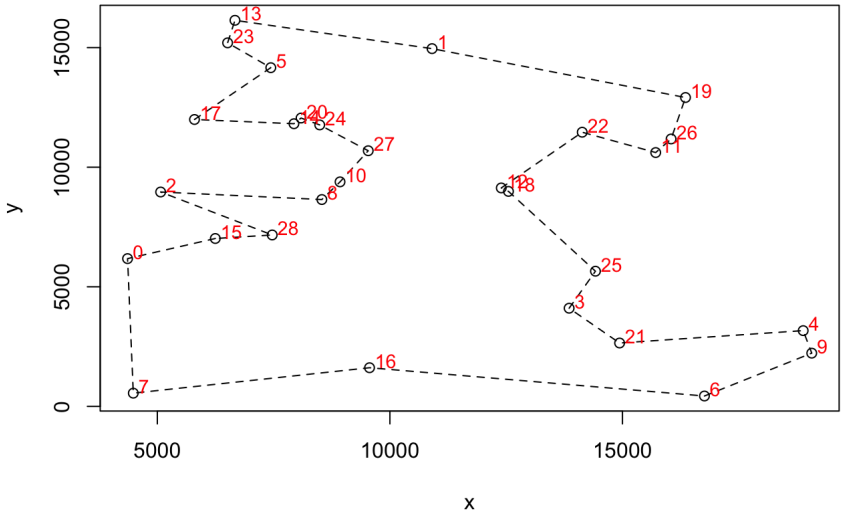

Mes 24
y
x
0
5000
10000
15000
5000
10000
15000
13
23
1
5
17
19
22
11
2
10
15
0
3
21
4
9
16
7
6


Mes 25
y
x
0
5000
10000
15000
5000
10000
15000

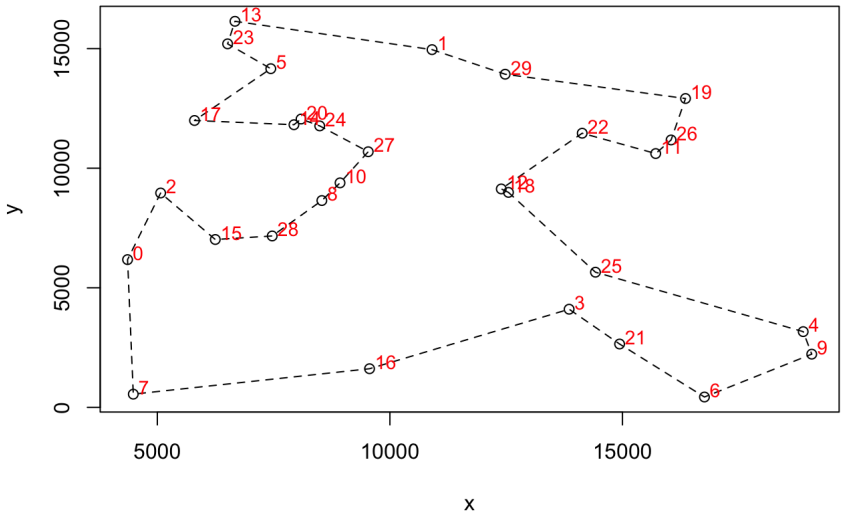

Mes 26
y
x
0
5000
10000
15000
5000
10000
15000


Mes 27
y
x
0
5000
10000
15000
5000
10000
15000

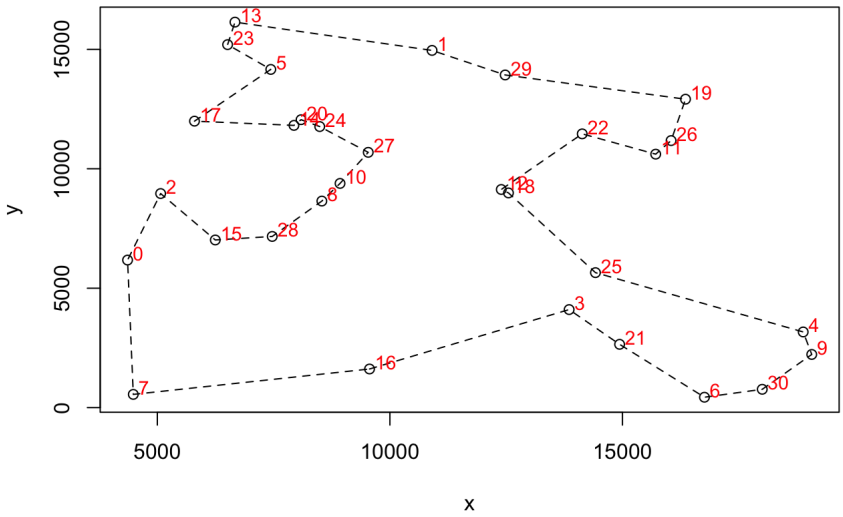
Mes 28

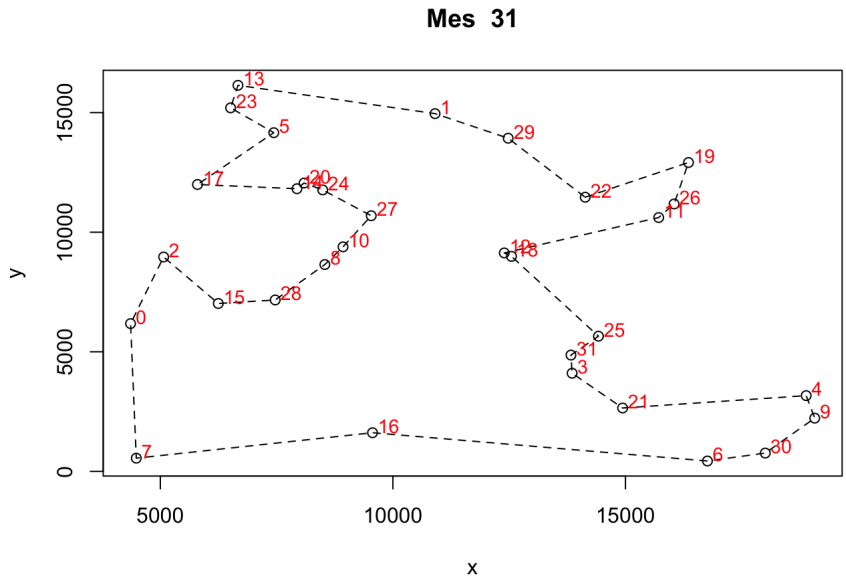


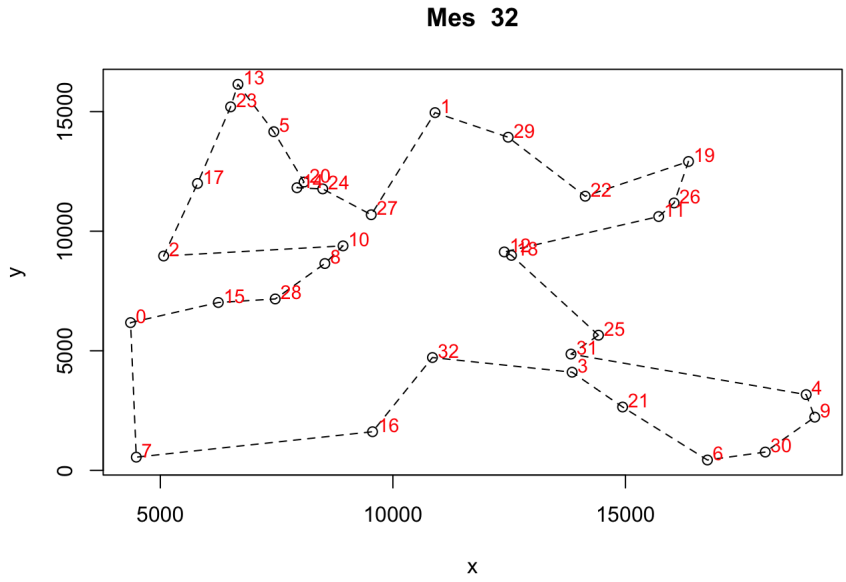
Mes 29

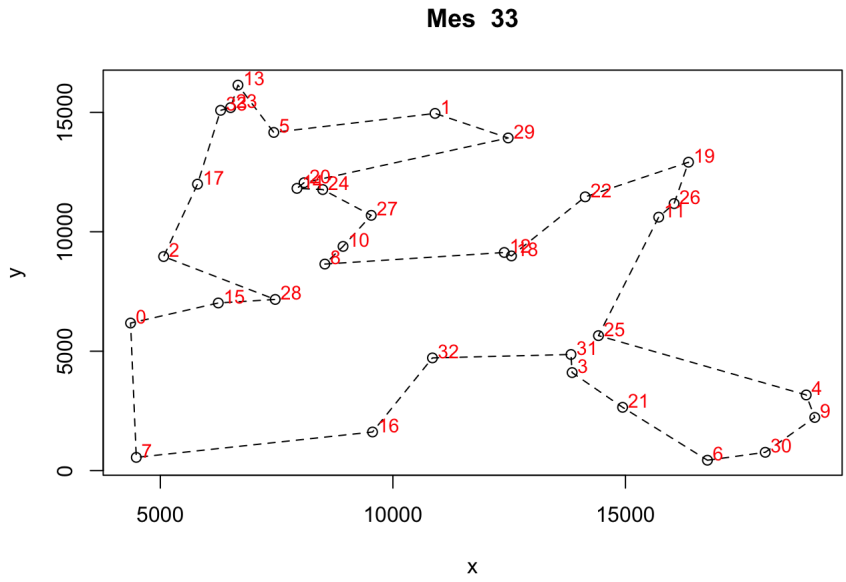


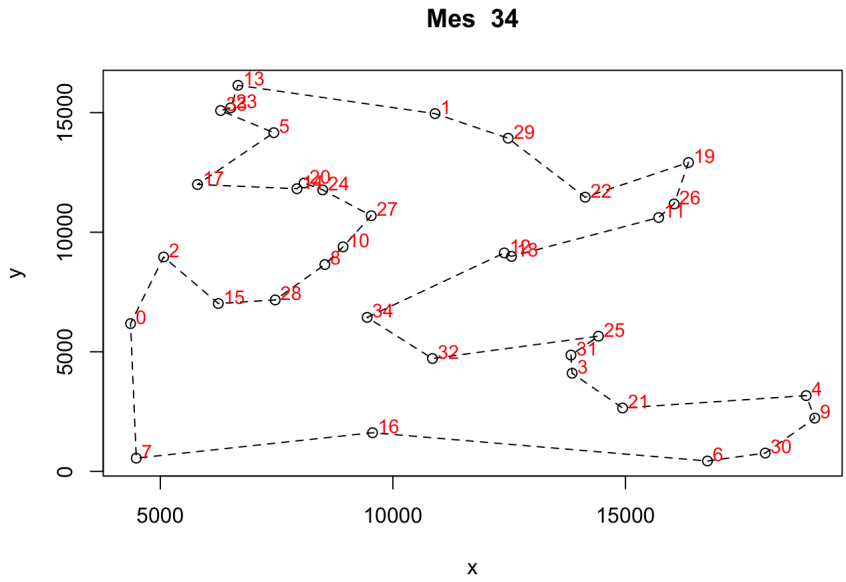
Mes 30

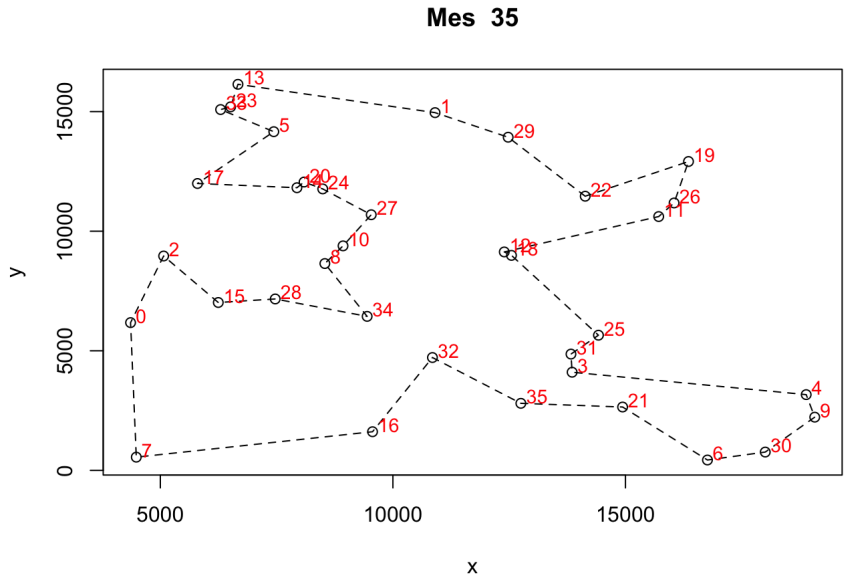


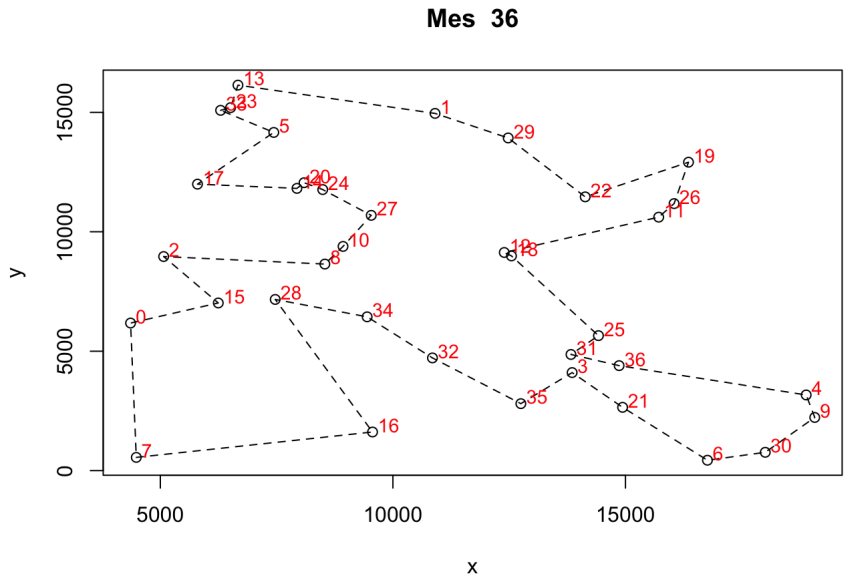

Mes 31
y
x
0
5000
10000
15000
5000
10000
15000


Mes 32
y
x
0
5000
10000
15000
5000
10000
15000


Mes 33
y
x
0
5000
10000
15000
5000
10000
15000


Mes 34
y
x
0
5000
10000
15000


Mes 35
y
x
0
5000
10000
15000


Mes 36
y
x
0
5000
10000
15000

Pregunta 4

Enunciado

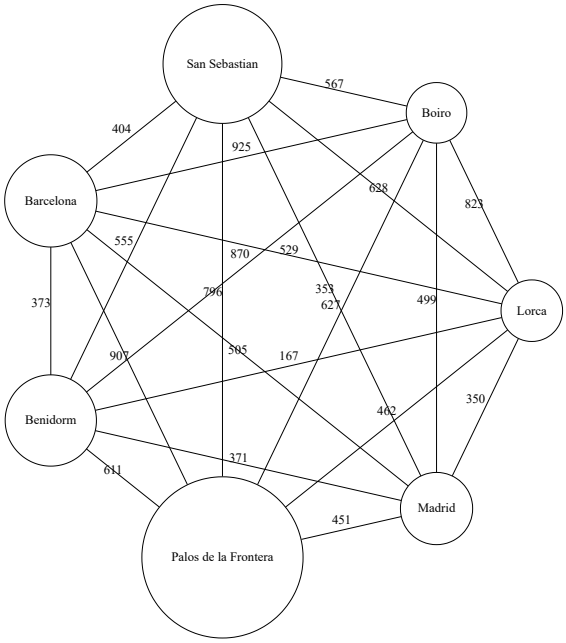
La compañía de retail está planificando las rutas de reparto de los refrescos que va a empezar a fabricar según hemos visto en los anteriores ejercicios. Teniendo en cuenta que la fábrica de refrescos se encuentra en Madrid:

Ayuda al equipo de planificación para buscar la ruta óptima, usando los algoritmos heurísticos NNA, RNNA y CLA (con script R). Las ciudades de suministro son San Sebastian, Palos de la Frontera, Madrid, Lorca, Boiro, Benidorm, Barcelona. Define los algoritmos heurísticos NNA, RNNA Y CLA.

Respuesta

Tal y como indica el enunciado debemos buscar la ruta óptima a partir de la ciudad de Madrid usando los algoritmos heurísticos NNA, RNNA y CLA. Lo primero se va a realizar una composición de los datos obtenidos y ubicaciones de las ciudades del problema con librerias de R explicadas en el código a continuación:

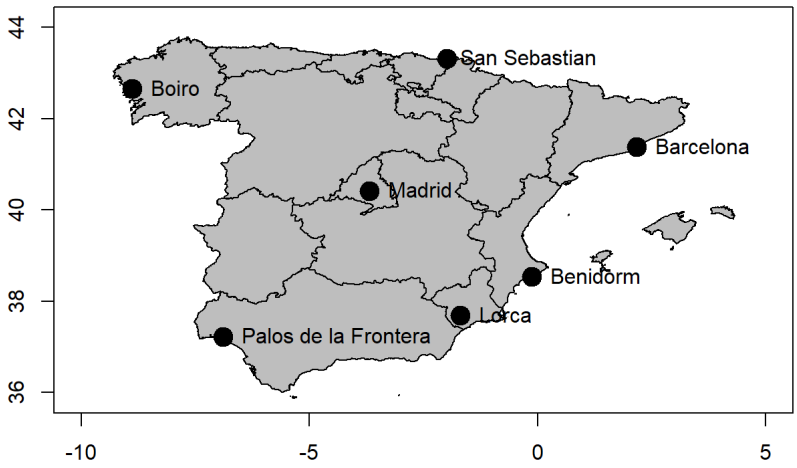
```
# Librería
.packages = c("DiagrammeR")

# Validación
.inst <- .packages %in% installed.packages()
if(length(.packages[!.inst]) > 0)
  suppressPackageStartupMessages(install.packages(.packages[!.inst]))

# Carga
suppressPackageStartupMessages(library(DiagrammeR))

# Creación diagrama
grViz("
  digraph {
    overlap = true
    graph [layout = circo]
    node [shape = circle]
    'Madrid'->'Lorca' [arrowhead = none, label = '350']
    'Madrid'->'Benidorm' [arrowhead = none, label = '371']
    'Lorca'->'Benidorm' [arrowhead = none, label = '167']
    'Madrid'->'Barcelona' [arrowhead = none, label = '505']
    'Lorca'->'Barcelona' [arrowhead = none, label = '529']
    'Benidorm'->'Barcelona' [arrowhead = none, label = '373']
    'Madrid'->'San Sebastian' [arrowhead = none, label = '353']
    'Lorca'->'San Sebastian' [arrowhead = none, label = '628']
    'Benidorm'->'San Sebastian' [arrowhead = none, label = '555']
    'Barcelona'->'San Sebastian' [arrowhead = none, label = '404']
    'Madrid'->'Boiro' [arrowhead = none, label = '499']
    'Lorca'->'Boiro' [arrowhead = none, label = '823']
    'Benidorm'->'Boiro' [arrowhead = none, label = '870']
    'Barcelona'->'Boiro' [arrowhead = none, label = '925']
    'San Sebastian'->'Boiro' [arrowhead = none, label = '567']
    'Madrid'->'Palos de la Frontera' [arrowhead = none, label = '451']
    'Lorca'->'Palos de la Frontera' [arrowhead = none, label = '462']
    'Benidorm'->'Palos de la Frontera' [arrowhead = none, label = '611']
    'Barcelona'->'Palos de la Frontera' [arrowhead = none, label = '907']
    'San Sebastian'->'Palos de la Frontera' [arrowhead = none, label = '796']
    'Boiro'->'Palos de la Frontera' [arrowhead = none, label = '627']
  }")
```



De esta forma se ha obtenido el grafo de distancias entre las ciudades. A continuación se representan las ciudades en el mapa

```
# Librerías
.packages = c("TSP","maps","sp","maptools","geosphere")
library(geosphere)
library(maptools)
library(rgdal)
library(raster)
library(maps)
library(mapdata)
library(ggmap)
library(marmap)
library(lattice)

# Validación
.inst <- .packages %in% installed.packages()
if(length(.packages[!.inst]) > 0)
  suppressPackageStartupMessages(install.packages(.packages[!.inst]))

# Carga
suppressPackageStartupMessages(library("TSP"))
suppressPackageStartupMessages(library("sp"))
suppressPackageStartupMessages(library("maps"))
suppressPackageStartupMessages(library("maptools"))
suppressPackageStartupMessages(library("geosphere"))

reparto_refrescos <- read.csv("Ejercicio4.csv",sep=";")
reparto_refrescos
```

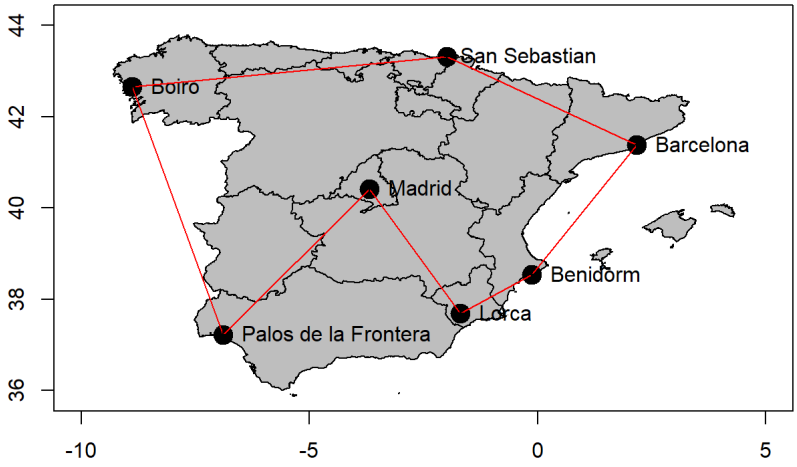
```
#Cargamos las coordenadas de las ciudades en coords
coords = cbind(reparto_refrescos$long, reparto_refrescos$lat)
#cargamos en spdf un dataframe con los puntos espaciales de las coordenadas
spdf = SpatialPointsDataFrame(coords, reparto_refrescos)


# calcular distancia real entre los puntos utilizando el método distHaversine que mide la distancia más corta entre dos punt
os. Este método asume una tierra esférica, ignorando los efectos elipsoidales.
dist_real<-distm(coords,fun = distHaversine)

#Necesitamos crear una matriz de distancias de tipo TSP para que funcione solve_TSP
tsp<-TSP(dist_real)

#Descargamos el mapa de España, con level=1 lo tendremos por comunidad autónoma
Spain_basemap <- getData('GADM', country="SPAIN", level=1)

# mapa con nombres de ciudaddes
#Creamos un marco para incluir el mapa con un tamaño un poco más grande para que quepa correctamente el mapa de España, por
 defecto lo corta.
plot(as(spdf, "Spatial"), axes=TRUE,xlim=c(-10,5),ylim=c(36,44))
#Incluimos el mapa y lo pintamos en gris
plot(Spain_basemap, add=TRUE, col = "gray")

# se puede reducir el tamaño de los puntos reduciendo el valor de cex
points(spdf, pch=20, cex=3, col="black")
# añadir etiquetas de nombre de ciudad, modificar pos para poner texto encima, debajo, derecha o izquierad del punto
text(spdf$long,spdf$lat,labels=spdf$name,pos=4)
```



Nearest-Neighbour Algorithm (NNA)

El primer algorirmo que revisamos es el vecino más cercano, en inglés Nearest-Neighbour Algorithm (NNA). Consiste en el siguiente proceso: - Se comienza por una ciudad de origen, en nuestro caso A. - Se revisan todos los arcos que salen hacia nodos (ciudades) que no han sido visitados y se elige el nodo más cercano. - Se repite esta operación hasta que se han visitado todas las ciudades y se regresa a la ciudad de origen.

```
#Calculamos el tour con el método por defecto y decimos que empiece por Madrid (1)
tour1 <- solve_TSP(tsp,method = "nn", start=1)

tour_line1 <- SpatialLines(list(Lines(list(Line(coords[c(tour1, tour1[1]),])), ID="1")))

# mapa con nombres de ciudaddes
#Creamos un marco para incluir el mapa con un tamaño un poco más grande para que quepa correctamente el mapa de España, por
 defecto lo corta.
plot(as(spdf, "Spatial"), axes=TRUE,xlim=c(-10,5),ylim=c(36,44))
#Incluimos el mapa y lo pintamos en gris
plot(Spain_basemap, add=TRUE, col = "gray")

# se puede reducir el tamaño de los puntos reduciendo el valor de cex
points(spdf, pch=20, cex=3, col="black")
# añadir etiquetas de nombre de ciudad, modificar pos para poner texto encima, debajo, derecha o izquierad del punto
text(spdf$long,spdf$lat,labels=spdf$name,pos=4)

plot(tour_line1, add=TRUE, col = "red")
points(coords, pch=3, cex=0.4, col="black")
```




```
tail(tour1)
```

```
## 2 3 4 5 6 7
## 2 3 4 5 6 7
```

```
tour1
```

```
## object of class 'TOUR'
## result of method 'nn' for 7 cities
## tour length: 2938859
```

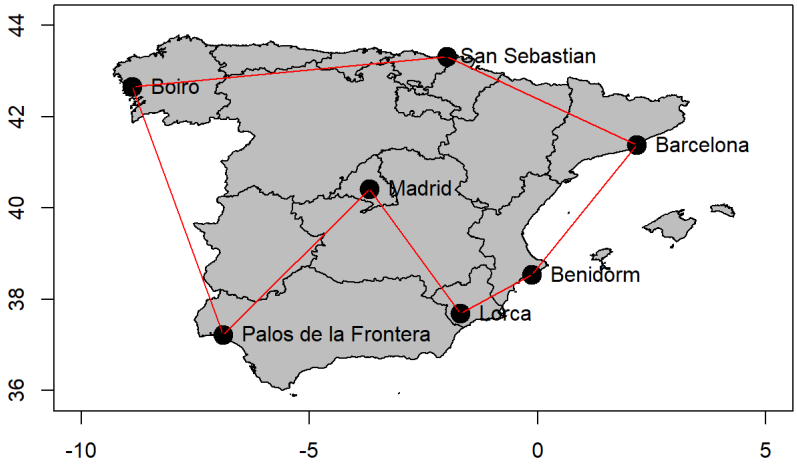
En nuestro caso, empezamos por $Madrid$ y necesariamente seguimos por $Lorca$. Después $Benidorm$, $Barcelona$, $SanSebastian$, $Boiro$, $PalosdelaFrontera$ y regresamos a $Madrid$.En total son 2939 km.

$$Madrid \Longrightarrow^{350} Lorca \Longrightarrow^{167} Benidorm \Longrightarrow^{373} Barcelona \Longrightarrow^{404} SanSebastian \Longrightarrow^{567} Boiro \Longrightarrow^{627} PalosdelaFrontera \Longrightarrow^{451} Madri$$

Recursive Nearest-Neighbour Algorithm (RNNA)

Este es una variante del algoritmo heurístico anterior. Consisten en aplicar NNA a cada uno de los nodos y posteriormente escoger el circuito que minimiza la métrica usada.

```
tour2 <- solve_TSP(tsp,method = "repetitive_nn", start=1)

tour_line2 <- SpatialLines(list(Lines(list(Line(coords[c(tour2, tour2[1]),])), ID="1")))

# mapa con nombres de ciudaddes
#Creamos un marco para incluir el mapa con un tamaño un poco más grande para que quepa correctamente el mapa de España, por
 defecto lo corta.
plot(as(spdf, "Spatial"), axes=TRUE,xlim=c(-10,5),ylim=c(36,44))
#Incluimos el mapa y lo pintamos en gris
plot(Spain_basemap, add=TRUE, col = "gray")

# se puede reducir el tamaño de los puntos reduciendo el valor de cex
points(spdf, pch=20, cex=3, col="black")
# añadir etiquetas de nombre de ciudad, modificar pos para poner texto encima, debajo, derecha o izquierad del punto
text(spdf$long,spdf$lat,labels=spdf$name,pos=4)

plot(tour_line2, add=TRUE, col = "red")
points(coords, pch=3, cex=0.4, col="black")
```




```
tail(tour2)
```

```
## 2 3 4 5 6 7
## 2 3 4 5 6 7
```

```
tour2
```

```
## object of class 'TOUR'
## result of method 'repetitive_nn' for 7 cities
## tour length: 2938859
```

En nuestro ejemplo: * Desde Madrid, la distancia es 2939 km usando NNA:

$$Madrid \Longrightarrow^{350} Lorca \Longrightarrow^{167} Benidorm \Longrightarrow^{373} Barcelona \Longrightarrow^{404} SanSebastian \Longrightarrow^{567} Boiro \Longrightarrow^{627} PalosdelaFrontera \Longrightarrow^{451} Madri$$

- Desde Lorca, la distancia es 2939 km usando NNA:

$$Lorca \Longrightarrow^{167} Benidorm \Longrightarrow^{373} Barcelona \Longrightarrow^{404} SanSebastian \Longrightarrow^{567} Boiro \Longrightarrow^{627} PalosdelaFrontera \Longrightarrow^{451} Madrid \Longrightarrow^{350} Lorca$$

- Desde Benidorm, la distancia es 2939 km usando NNA:

$$Benidorm \Longrightarrow^{373} Barcelona \Longrightarrow^{404} SanSebastian \Longrightarrow^{567} Boiro \Longrightarrow^{627} PalosdelaFrontera \Longrightarrow^{451} Madrid \Longrightarrow^{350} Lorca \Longrightarrow^{167} Benido$$

- Desde Barcelona, la distancia es 2939 km usando NNA:

$$Barcelona \Longrightarrow^{404} SanSebastian \Longrightarrow^{567} Boiro \Longrightarrow^{627} PalosdelaFrontera \Longrightarrow^{451} Madrid \Longrightarrow^{350} Lorca \Longrightarrow^{167} Benidorm \Longrightarrow^{373} Barcelo$$

- Desde San Sebastian, la distancia es 2939 km usando NNA:

$$SanSebastian \Longrightarrow^{567} Boiro \Longrightarrow^{627} PalosdelaFrontera \Longrightarrow^{451} Madrid \Longrightarrow^{350} Lorca \Longrightarrow^{167} Benidorm \Longrightarrow^{373} Barcelona \Longrightarrow^{404} SanSeb$$

- Desde Boiro, la distancia es usando 2939 km NNA:

$$Boiro \Longrightarrow^{627} PalosdelaFrontera \Longrightarrow^{451} Madrid \Longrightarrow^{350} Lorca \Longrightarrow^{167} Benidorm \Longrightarrow^{373} Barcelona \Longrightarrow^{404} SanSebastian \Longrightarrow^{567} Boiro$$

- Desde Palos de la Frontera, la distancia es 2939 km usando NNA:

$$PalosdelaFrontera \Longrightarrow^{451} Madrid \Longrightarrow^{350} Lorca \Longrightarrow^{167} Benidorm \Longrightarrow^{373} Barcelona \Longrightarrow^{404} SanSebastian \Longrightarrow^{567} Boiro \Longrightarrow^{627} Palosd$$

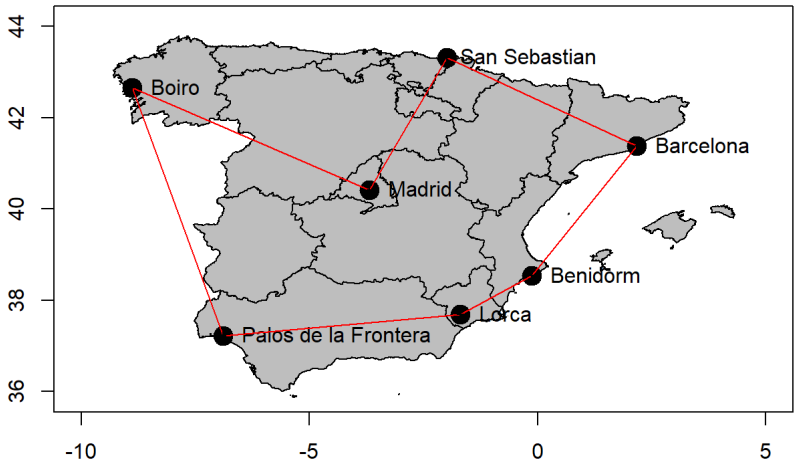
Como es posible detectar en este caso, los resultados de NNA y RNNA coincide en diversos escenarios, pero no tiene por qué ser así. En este caso, la solución de NNA sigue siendo la que seleccionamos.

Cheapest-Link Algorithm (CLA)

El algoritmo “el enlace más barato”, en inglés Cheapest-Link Algorithm (CLA). Consiste en el siguiente proceso:

- Se selecciona el arco con menos peso y se pinta.
- Se repite el paso anterior a menos que:
  - Se cierra la ruta sin cubrir todos los nodos.
  - De un mismo nodo (o vértice) tiene más de dos arcos seleccionados mediante el criterio.
  - Se haya completado la ruta.

```
tour3 <- solve_TSP(tsp,method = "cheapest_insertion", start=1)
tour_line3 <- SpatialLines(list(Lines(list(Line(coords[c(tour3, tour3[1]),])), ID="1")))

# mapa con nombres de ciudaddes
#Creamos un marco para incluir el mapa con un tamaño un poco más grande para que quepa correctamente el mapa de España, por
 defecto lo corta.
plot(as(spdf, "Spatial"), axes=TRUE,xlim=c(-10,5),ylim=c(36,44))
#Incluimos el mapa y lo pintamos en gris
plot(Spain_basemap, add=TRUE, col = "gray")

# se puede reducir el tamaño de los puntos reduciendo el valor de cex
points(spdf, pch=20, cex=3, col="black")
# añadir etiquetas de nombre de ciudad, modificar pos para poner texto encima, debajo, derecha o izquierad del punto
text(spdf$long,spdf$lat,labels=spdf$name,pos=4)


plot(tour_line3, add=TRUE, col = "red")
points(coords, pch=3, cex=0.4, col="black")
```




```
tail(tour3)
```

```
## 5 4 3 2 7 6
## 5 4 3 2 7 6
```

```
tour3
```

```
## object of class 'TOUR'
## result of method 'cheapest_insertion' for 7 cities
## tour length: 2884407
```

- Con este método, la distancia recorrida se reduce a 2885 km y la secuencia sería:

$$Madrid \overset{353}{\Longrightarrow} SanSebastian \overset{404}{\Longrightarrow} Barcelona \overset{373}{\Longrightarrow} Benidorm \overset{167}{\Longrightarrow} Lorca \overset{462}{\Longrightarrow} PalosdelaFrontera \overset{627}{\Longrightarrow} Boiro \overset{499}{\Longrightarrow} Madri$$