Anexo

June 26, 2022

1 Código y datos del modelo SPLP en IBM ILOG CPLEX Optimization Studio

[ ]:
```
/*********************************************
 * OPL 22.1.0.0 Model
 * Author: Usuario
 * Creation Date: 7 jun. 2022 at 12:29:14
 *********************************************/

  // MODELO SPLP


  // Defino las variables fijas
 int n = ...;//número de cliente
 int m = ...;//número de ubicaciones


 range cliente=1..n;
 range ubicacion=1..m;

 float u[cliente]=...;

 float f[ubicacion]=...;

 float distancia[cliente][ubicacion] = ... ;

  // Defino variables de decision
 dvar boolean x[cliente][ubicacion];
 dvar boolean y[ubicacion];

 minimize sum (i in cliente,j in ubicacion) (distancia[i][j]*u[i]*x[i][j]) +␣
 ↪sum(j in ubicacion) f[j]*y[j];
 subject to {
//cada cliente es atendido por una y solo una instalación
forall (i in cliente) sum(j in ubicacion)
        x[i][j] == 1;
//un cliente es atendido por una instalacion que esta abierta
```

```
forall (i in cliente, j in ubicacion)
          x[i][j] <= y[j];



}
main {
       var f = new IloOplOutputFile("solucionSPLP1.txt");
       thisOplModel.generate();
          var time0 = new Date();
       var OK = cplex.solve();
       var time1 = new Date();

       var tTotal = (time1.getTime()-time0.getTime())/1000;

       if (OK) {
               f.writeln("Instalaciones abiertas");
               for(var j in thisOplModel.ubicacion){
                                       if(thisOplModel.y[j] != 0){
                                       f.writeln("Abro instalacion ",j);
                               }
               }
               f.writeln("Asignacion de clientes");
               for(var j in thisOplModel.ubicacion){
                       for(var i in thisOplModel.cliente){
                                       if(thisOplModel.x[i][j] != 0){
                                       f.writeln("a la instalacion ",j," van␣
↪los clientes ",i);
                                       }
                       }
               }
               f.writeln("Tiempo: ",tTotal,"segundos. Objetivo: "+cplex.
↪getObjValue( ));
               }
       f.close();
}
```

[ ]:
```
/*********************************************
 * OPL 22.1.0.0 Data
 * Author: Usuario
 * Creation Date: 7 jun. 2022 at 12:29:14
 *********************************************/

 //Numero de nodos
 n=20;
 m=20;
```

```
 //DATOS SPLP


//Demanda en los puntos
u=[9,28,77,59,86,6,52,78,80,10,16,9,33,48,25,66,21,34,48,24];

// Costes fijos

f=[100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100,␣
 ↪100, 100, 100, 100, 100];

//SE CAMBIA f CON LOS DATOS DE CADA ESCENARIO


//Distancia entre los nodos
distancia=[[0, 2.6675082 , 1.38751577, 4.86004115, 5.41826541,
        6.38451251, 4.68772866, 7.3801084 , 7.04420329, 5.33816448,
        8.22982381, 5.71153219, 4.6462458 , 4.00284899, 4.22497337,
        4.43312125, 2.80927037, 5.24453277, 3.68893535, 1.42702628],
       [2.6675082 , 0.        , 1.43387587, 2.20735135, 3.0016662 ,
        3.91959182, 2.61457454, 5.664274  , 5.62697077, 4.57930126,
        9.04707688, 7.08025423, 6.36452669, 5.96422669, 5.65971731,
        5.97333441, 3.57843541, 5.75086811, 2.15916743, 3.85608143],
       [1.38751577, 1.43387587, 0.        , 3.54045195, 4.03112887,
        4.99807963, 3.31843337, 6.11731967, 5.86668561, 4.36206373,
        8.10106166, 5.89185879, 5.05489861, 4.58911756, 4.42235232,
        4.70943776, 2.48652368, 4.8845229 , 2.38564121, 2.42937111],
       [4.86004115, 2.20735135, 3.54045195, 0.        , 1.37927517,
        1.99529446, 1.91760267, 4.59464906, 4.90493629, 4.72787479,
        9.92380975, 8.45165073, 7.99031914, 7.74578595, 7.1797493 ,
        7.52602976, 5.05916989, 6.68611128, 2.56312388, 5.949602  ],
       [5.41826541, 3.0016662 , 4.03112887, 1.37927517, 0.        ,
        0.97015463, 1.07424392, 3.22024844, 3.58452228, 3.75712656,
        9.11765321, 7.99249648, 7.74010336, 7.64002618, 6.87592903,
        7.23355542, 4.85831246, 6.01399069, 2.18210999, 6.26426085],
       [6.38451251, 3.91959182, 4.99807963, 1.99529446, 0.97015463,
        0.        , 1.93762742, 3.03907881, 3.62397572, 4.28788992,
        9.67388236, 8.75792213, 8.58920252, 8.53290103, 7.71318352,
        8.07183275, 5.74891294, 6.69380639, 3.09240424, 7.23020913],
       [4.68772866, 2.61457454, 3.31843337, 1.91760267, 1.07424392,
        1.93762742, 0.        , 3.06646376, 3.15296686, 2.82651729,
        8.11817714, 6.91968207, 6.67418909, 6.596969  , 5.80637581,
        6.16438837, 3.81266311, 4.96582762, 1.1607601 , 5.37783637],
       [7.3801084 , 5.664274  , 6.11731967, 4.59464906, 3.22024844,
        3.03907881, 3.06646376, 0.        , 0.90354856, 2.80178515,
        7.56861942, 7.49252961, 7.78310992, 8.02765221, 6.90938492,
        7.24197791, 5.53407626, 5.25040989, 3.73378146, 7.69028504],
```

```
       [7.04420329, 5.62697077, 5.86668561, 4.90493629, 3.58452228,
        3.62397572, 3.15296686, 0.90354856, 0.        , 2.06649462,
        6.66564325, 6.64896985, 7.00927956, 7.30733878, 6.14966666,
        6.47174196, 4.94683737, 4.40962629, 3.54388544, 7.20011694],
       [5.33816448, 4.57930126, 4.36206373, 4.72787479, 3.75712656,
        4.28788992, 2.82651729, 2.80178515, 2.06649462, 0.        ,
        5.3883207 , 4.75857121, 4.98144557, 5.24492135, 4.10823563,
        4.44020765, 2.94550505, 2.54086678, 2.46136629, 5.27121276],
       [8.22982381, 9.04707688, 8.10106166, 9.92380975, 9.11765321,
        9.67388236, 8.11817714, 7.56861942, 6.66564325, 5.3883207 ,
        0.        , 2.8276492 , 4.13782552, 4.97521859, 4.11246884,
        4.03402578, 5.61729472, 3.30301741, 7.38952258, 7.18618146],
       [5.71153219, 7.08025423, 5.89185879, 8.45165073, 7.99249648,
        8.75792213, 6.91968207, 7.49252961, 6.64896985, 4.75857121,
        2.8276492 , 0.        , 1.31041978, 2.14895323, 1.49345238,
        1.29287432, 3.50473965, 2.24216057, 5.92189193, 4.51266706],
       [4.6462458 , 6.36452669, 5.05489861, 7.99031914, 7.74010336,
        8.58920252, 6.67418909, 7.78310992, 7.00927956, 4.98144557,
        4.13782552, 1.31041978, 0.        , 0.83952367, 0.89106678,
        0.54236888, 2.93264386, 2.75469127, 5.57668755, 3.34395036],
       [4.00284899, 5.96422669, 4.58911756, 7.74578595, 7.64002618,
        8.53290103, 6.596969  , 8.02765221, 7.30733878, 5.24492135,
        4.97521859, 2.14895323, 0.83952367, 0.        , 1.25936492,
        1.07762888, 2.7843132 , 3.28588557, 5.45792671, 2.63262683],
       [4.22497337, 5.65971731, 4.42235232, 7.1797493 , 6.87592903,
        7.71318352, 5.80637581, 6.90938492, 6.14966666, 4.10823563,
        4.11246884, 1.49345238, 0.89106678, 1.25936492, 0.        ,
        0.35866976, 2.13053984, 2.0265251 , 4.72537025, 3.08007857],
       [4.43312125, 5.97333441, 4.70943776, 7.52602976, 7.23355542,
        8.07183275, 6.16438837, 7.24197791, 6.47174196, 4.44020765,
        4.03402578, 1.29287432, 0.54236888, 1.07762888, 0.35866976,
        0.        , 2.47040159, 2.2631836 , 5.08062988, 3.22018633],
       [2.80927037, 3.57843541, 2.48652368, 5.05916989, 4.85831246,
        5.74891294, 3.81266311, 5.53407626, 4.94683737, 2.94550505,
        5.61729472, 3.50473965, 2.93264386, 2.7843132 , 2.13053984,
        2.47040159, 0.        , 2.46078118, 2.67719331, 2.36142415],
       [5.24453277, 5.75086811, 4.8845229 , 6.68611128, 6.01399069,
        6.69380639, 4.96582762, 5.25040989, 4.40962629, 2.54086678,
        3.30301741, 2.24216057, 2.75469127, 3.28588557, 2.0265251 ,
        2.2631836 , 2.46078118, 0.        , 4.12800194, 4.50497503],
       [3.68893535, 2.15916743, 2.38564121, 2.56312388, 2.18210999,
        3.09240424, 1.1607601 , 3.73378146, 3.54388544, 2.46136629,
        7.38952258, 5.92189193, 5.57668755, 5.45792671, 4.72537025,
        5.08062988, 2.67719331, 4.12800194, 0.        , 4.25205833],
       [1.42702628, 3.85608143, 2.42937111, 5.949602  , 6.26426085,
        7.23020913, 5.37783637, 7.69028504, 7.20011694, 5.27121276,
        7.18618146, 4.51266706, 3.34395036, 2.63262683, 3.08007857,
```

```
       3.22018633, 2.36142415, 4.50497503, 4.25205833, 0.        ]];
```

2 Código y datos del modelo de la p-mediana en IBM ILOG CPLEX Optimization Studio

[ ]:
```
/*********************************************
 * OPL 22.1.0.0 Model
 * Author: Usuario
 * Creation Date: 10 jun. 2022 at 8:33:00
 *********************************************/


   // MODELO P-MEDIANA CON DISTANCIA EUCLIDEA


  // Defino las variables fijas
 int n = ...;//número de cliente
 int m = ...;//número de ubicaciones
 int p=...;//núméo de instalaciones a ubicar

 range cliente=1..n;
 range ubicacion=1..m;

 float u[cliente]=...;

 float distancia[cliente][ubicacion] = ... ;

  // Defino variables de decision
 dvar boolean x[cliente][ubicacion];
 dvar boolean y[ubicacion];

 minimize sum (i in cliente,j in ubicacion) distancia[i][j]*u[i]*x[i][j];
 subject to {
//cada cliente es atendido por una y solo una instalación
forall (i in cliente) sum(j in ubicacion)
        x[i][j] == 1;
//un cliente es atendido por una instalacion que esta abierta
 forall (i in cliente, j in ubicacion)
           x[i][j] <= y[j];


//se abren justamente p instalaciones
sum(j in ubicacion) y[j] == p ;
 }
```

```
main {
       var f = new IloOplOutputFile("solucionpmed.txt");
       thisOplModel.generate();
          var time0 = new Date();
       var OK = cplex.solve();
       var time1 = new Date();

       var tTotal = (time1.getTime()-time0.getTime())/1000;

       if (OK) {
               f.writeln("Instalaciones abiertas");
               for(var j in thisOplModel.ubicacion){
                                       if(thisOplModel.y[j] != 0){
                                       f.writeln("Abro instalacion ",j);
                                }
               }
               f.writeln("Asignacion de clientes");
               for(var j in thisOplModel.ubicacion){
                       for(var i in thisOplModel.cliente){
                                       if(thisOplModel.x[i][j] != 0){
                                       f.writeln("a la instalacion ",j," van␣
↪los clientes ",i);
                                }
                       }
               }
               f.writeln("Tiempo: ",tTotal,"segundos. Objetivo: "+cplex.
↪getObjValue( ));
               }
       f.close();
}
```

[ ]:
```
/*********************************************
 * OPL 22.1.0.0 Data
 * Author: pardo
 * Creation Date: 10 jun. 2022 at 8:33:00
 *********************************************/


//DATOS DE LOS MODELOS P-MEDIANA CON DISTANCIA EUCLIDEA

 //Numero de nodos
 n=20;
 m=20;
 p=2;
//Se cambia p para cada escenario
```

```
//Demanda en los puntos
u=[9,28,77,59,86,6,52,78,80,10,16,9,33,48,25,66,21,34,48,24];


//Distancia entre los nodos
distancia=[[0, 2.6675082 , 1.38751577, 4.86004115, 5.41826541,
        6.38451251, 4.68772866, 7.3801084 , 7.04420329, 5.33816448,
        8.22982381, 5.71153219, 4.6462458 , 4.00284899, 4.22497337,
        4.43312125, 2.80927037, 5.24453277, 3.68893535, 1.42702628],
       [2.6675082 , 0.        , 1.43387587, 2.20735135, 3.0016662 ,
        3.91959182, 2.61457454, 5.664274  , 5.62697077, 4.57930126,
        9.04707688, 7.08025423, 6.36452669, 5.96422669, 5.65971731,
        5.97333441, 3.57843541, 5.75086811, 2.15916743, 3.85608143],
       [1.38751577, 1.43387587, 0.        , 3.54045195, 4.03112887,
        4.99807963, 3.31843337, 6.11731967, 5.86668561, 4.36206373,
        8.10106166, 5.89185879, 5.05489861, 4.58911756, 4.42235232,
        4.70943776, 2.48652368, 4.8845229 , 2.38564121, 2.42937111],
       [4.86004115, 2.20735135, 3.54045195, 0.        , 1.37927517,
        1.99529446, 1.91760267, 4.59464906, 4.90493629, 4.72787479,
        9.92380975, 8.45165073, 7.99031914, 7.74578595, 7.1797493 ,
        7.52602976, 5.05916989, 6.68611128, 2.56312388, 5.949602  ],
       [5.41826541, 3.0016662 , 4.03112887, 1.37927517, 0.        ,
        0.97015463, 1.07424392, 3.22024844, 3.58452228, 3.75712656,
        9.11765321, 7.99249648, 7.74010336, 7.64002618, 6.87592903,
        7.23355542, 4.85831246, 6.01399069, 2.18210999, 6.26426085],
       [6.38451251, 3.91959182, 4.99807963, 1.99529446, 0.97015463,
        0.        , 1.93762742, 3.03907881, 3.62397572, 4.28788992,
        9.67388236, 8.75792213, 8.58920252, 8.53290103, 7.71318352,
        8.07183275, 5.74891294, 6.69380639, 3.09240424, 7.23020913],
       [4.68772866, 2.61457454, 3.31843337, 1.91760267, 1.07424392,
        1.93762742, 0.        , 3.06646376, 3.15296686, 2.82651729,
        8.11817714, 6.91968207, 6.67418909, 6.596969  , 5.80637581,
        6.16438837, 3.81266311, 4.96582762, 1.1607601 , 5.37783637],
       [7.3801084 , 5.664274  , 6.11731967, 4.59464906, 3.22024844,
        3.03907881, 3.06646376, 0.        , 0.90354856, 2.80178515,
        7.56861942, 7.49252961, 7.78310992, 8.02765221, 6.90938492,
        7.24197791, 5.53407626, 5.25040989, 3.73378146, 7.69028504],
       [7.04420329, 5.62697077, 5.86668561, 4.90493629, 3.58452228,
        3.62397572, 3.15296686, 0.90354856, 0.        , 2.06649462,
        6.66564325, 6.64896985, 7.00927956, 7.30733878, 6.14966666,
        6.47174196, 4.94683737, 4.40962629, 3.54388544, 7.20011694],
       [5.33816448, 4.57930126, 4.36206373, 4.72787479, 3.75712656,
        4.28788992, 2.82651729, 2.80178515, 2.06649462, 0.        ,
        5.3883207 , 4.75857121, 4.98144557, 5.24492135, 4.10823563,
        4.44020765, 2.94550505, 2.54086678, 2.46136629, 5.27121276],
       [8.22982381, 9.04707688, 8.10106166, 9.92380975, 9.11765321,
        9.67388236, 8.11817714, 7.56861942, 6.66564325, 5.3883207 ,
```

```
        0.        , 2.8276492 , 4.13782552, 4.97521859, 4.11246884,
        4.03402578, 5.61729472, 3.30301741, 7.38952258, 7.18618146],
       [5.71153219, 7.08025423, 5.89185879, 8.45165073, 7.99249648,
        8.75792213, 6.91968207, 7.49252961, 6.64896985, 4.75857121,
        2.8276492 , 0.        , 1.31041978, 2.14895323, 1.49345238,
        1.29287432, 3.50473965, 2.24216057, 5.92189193, 4.51266706],
       [4.6462458 , 6.36452669, 5.05489861, 7.99031914, 7.74010336,
        8.58920252, 6.67418909, 7.78310992, 7.00927956, 4.98144557,
        4.13782552, 1.31041978, 0.        , 0.83952367, 0.89106678,
        0.54236888, 2.93264386, 2.75469127, 5.57668755, 3.34395036],
       [4.00284899, 5.96422669, 4.58911756, 7.74578595, 7.64002618,
        8.53290103, 6.596969  , 8.02765221, 7.30733878, 5.24492135,
        4.97521859, 2.14895323, 0.83952367, 0.        , 1.25936492,
        1.07762888, 2.7843132 , 3.28588557, 5.45792671, 2.63262683],
       [4.22497337, 5.65971731, 4.42235232, 7.1797493 , 6.87592903,
        7.71318352, 5.80637581, 6.90938492, 6.14966666, 4.10823563,
        4.11246884, 1.49345238, 0.89106678, 1.25936492, 0.        ,
        0.35866976, 2.13053984, 2.0265251 , 4.72537025, 3.08007857],
       [4.43312125, 5.97333441, 4.70943776, 7.52602976, 7.23355542,
        8.07183275, 6.16438837, 7.24197791, 6.47174196, 4.44020765,
        4.03402578, 1.29287432, 0.54236888, 1.07762888, 0.35866976,
        0.        , 2.47040159, 2.2631836 , 5.08062988, 3.22018633],
       [2.80927037, 3.57843541, 2.48652368, 5.05916989, 4.85831246,
        5.74891294, 3.81266311, 5.53407626, 4.94683737, 2.94550505,
        5.61729472, 3.50473965, 2.93264386, 2.7843132 , 2.13053984,
        2.47040159, 0.        , 2.46078118, 2.67719331, 2.36142415],
       [5.24453277, 5.75086811, 4.8845229 , 6.68611128, 6.01399069,
        6.69380639, 4.96582762, 5.25040989, 4.40962629, 2.54086678,
        3.30301741, 2.24216057, 2.75469127, 3.28588557, 2.0265251 ,
        2.2631836 , 2.46078118, 0.        , 4.12800194, 4.50497503],
       [3.68893535, 2.15916743, 2.38564121, 2.56312388, 2.18210999,
        3.09240424, 1.1607601 , 3.73378146, 3.54388544, 2.46136629,
        7.38952258, 5.92189193, 5.57668755, 5.45792671, 4.72537025,
        5.08062988, 2.67719331, 4.12800194, 0.        , 4.25205833],
       [1.42702628, 3.85608143, 2.42937111, 5.949602  , 6.26426085,
        7.23020913, 5.37783637, 7.69028504, 7.20011694, 5.27121276,
        7.18618146, 4.51266706, 3.34395036, 2.63262683, 3.08007857,
        3.22018633, 2.36142415, 4.50497503, 4.25205833, 0.        ]];
```

[ ]:
```
/*********************************************
 * OPL 22.1.0.0 Model
 * Author: pardo
```

```
 * Creation Date: 10 jun. 2022 at 8:35:21
 *********************************************/

//MODELO DE LA P-MEDIANA CON DISTANCIA MANHATTAN


  // Defino las variables fijas
 int n = ...;//número de cliente
 int m = ...;//número de ubicaciones
 int p=...;//númeo de instalaciones a ubicar

 range cliente=1..n;
 range ubicacion=1..m;
 range dimension=1..2;

 float u[cliente]=...;
 float coor[cliente][dimension]= ...;

 float distancia[cliente][ubicacion] ;

 execute{
        for (var i in cliente){
                for(var j in ubicacion){
                //defino la distancia Manhattan
                distancia[i][j]=Math.abs(coor[i][1]-coor[j][1])+Math.
 ↪abs(coor[i][2]-coor[j][2]);

                }
        }
}

  // Defino variables de decision
 dvar float+ x[cliente][ubicacion];
 dvar boolean y[ubicacion];

 minimize sum (i in cliente,j in ubicacion) distancia[i][j]*u[i]*x[i][j];
 subject to {
// se satisface la demanda de todos los clientes
forall (i in cliente) sum(j in ubicacion)
        x[i][j] == 1;
//un cliente es atendido por una instalacion que esta abierta
 forall (i in cliente, j in ubicacion)
           x[i][j] <= y[j];
//se abren justamente p instalaciones
sum(j in ubicacion) y[j] == p ;
 }
```

```
main {
       var f = new IloOplOutputFile("solucionpmedManhattan.txt");
       thisOplModel.generate();
          var time0 = new Date();
       var OK = cplex.solve();
       var time1 = new Date();

       var tTotal = (time1.getTime()-time0.getTime())/1000;

       if (OK) {
               f.writeln("Instalaciones abiertas");
               for(var j in thisOplModel.ubicacion){
                                       if(thisOplModel.y[j] != 0){
                                       f.writeln("Abro instalacion ",j);
                                }
               }
               f.writeln("Asignacion de clientes");
               for(var j in thisOplModel.ubicacion){
                       for(var i in thisOplModel.cliente){
                                       if(thisOplModel.x[i][j] != 0){
                                       f.writeln("a la instalacion ",j," van␣
 ↪los clientes ",i);
                                }
                       }
               }
               f.writeln("Tiempo: ",tTotal,"segundos. Objetivo: "+cplex.
 ↪getObjValue( ));
               }
       f.close();
}
```

[ ]:
```
/*********************************************
 * OPL 22.1.0.0 Data
 * Author: Usuario
 * Creation Date: 10 jun. 2022 at 8:35:21
 *********************************************/

//DATOS DE LOS MODELOS DE LA P-MEDIANA CON DISTANCIA MANHATTAN


 //Numero de nodos
 n=20;
 m=20;
 p=2;
// Se cambia p con el valor necesario en cada escenario
```

```
//Demanda en los puntos
u=[9,28,77,59,86,6,52,78,80,10,16,9,33,48,25,66,21,34,48,24];


//Coordenadas de los nodos
coor=[[0.8,4]
[3.46,4.2]
[2.14,3.64]
[5.66,4.02]
[6.06,2.7]
[7.02,2.56]
[5.12,2.18]
[6.68,-0.46]
[5.88,-0.88]
[3.88,-0.36]
[0.1,-4.2]
[-0.74,-1.5]
[-1.1,-0.24]
[-1.28,0.58]
[-0.22,-0.1]
[-0.558,-0.22]
[1.42,1.26]
[1.482,-1.2]
[3.962,2.1]
[-0.198,2.98]
];
```

3 Código y datos de los modelos de capacidad en IBM ILOG CPLEX Optimization Studio

[ ]:
```
/*********************************************
 * OPL 20.1.0.0 Model
 * Author: Usuario
 * Creation Date: 22 jun. 2022 at 18:06:25
 *********************************************/


 //MODELOS CAPACIDAD CAP-U,CAP-M,CAP-U-C,CAP-U-C<600,CAP-M-RE,CAP-U-RE
 //EN CADA ESCENARIO SE DESCOMENTA LA RESTRICCION NECESARIA Y SE EJECUTA


 // Defino las variables fijas
```

```
 int n = ...;//número de cliente

 int m = ...;//número de ubicaciones

 //int p=...; // número de instalaciones a ubicar

 float eps=...;



 range cliente=1..n;

 range ubicacion=1..m;

 range dimension=1..2;



 float B[cliente]= ...;
 float b[cliente]= ...;
 float u[cliente]= ...;  // numero de usuarios en cada centro de demanda
 float coor[cliente][dimension]= ...;
 float distancia[cliente][ubicacion]=...; // distancia entre los puntos
 float f[ubicacion]=...;
 float A[cliente][ubicacion];
 float BB[cliente][ubicacion];
 float C[cliente][ubicacion];

 float d1;
 float d2;

 //conjuntos Pij

 {int} P[i in cliente][j in ubicacion];

float cardinal[cliente][ubicacion];
 //float coordenada[cliente][2];

 //float demanda[cliente];

 //float capacidad[ubicacion];

 //execute initializeArray{

execute{
        var f = new IloOplOutputFile("datos.txt");
        var dr;
```

```
        var d1;
        var d2;
        var aux1;
        var aux2;
        //calculo primero las ecuaciones de las rectas
        for (var i in cliente){
                for(var j in ubicacion){
                        //calculo la recta Ax+By+C que pasa por (xi,yi) y␣
↪(xj,yj) si i es distinto de j
                        if(j != i){
                                A[i][j]=coor[j][2]-coor[i][2];//
                                BB[i][j]=coor[i][1]-coor[j][1];//
                                C[i][j]=(coor[j][1]-
                                coor[i][1])*coor[i][2]-(coor[j][2]-coor[i][2])*coor[i][1];
↪//
                        }
                        //recorro todos los clientes
                    for (var k in cliente){
                                dr=10;
                                d1=10;
                                d2=10;
                                aux1=0;
                                aux2=0
                                //calculo la distancia del punto (xk,yk) a la␣
↪recta
                                if (j != i) dr=Math.abs((A[i][j]*coor[k][1] +␣
↪BB[i][j]*coor[k][2] +
                            C[i][j])/Math.sqrt(A[i][j]^2+BB[i][j])^2);
↪
                            //calculo la distancia del punto (xk,yk) a (xi,yi)
                            aux1=(coor[i][1]-coor[k][1])*(coor[i][1]-coor[k][1]);
                            aux2=(coor[i][2]-coor[k][2])*(coor[i][2]-coor[k][2]);
                            d1=Math.sqrt(aux1+aux2);
                            //calculo la distancia del punto (xk,yk) a (xj,yj)
                            aux1=(coor[j][1]-coor[k][1])*(coor[j][1]-coor[k][1]);
                            aux2=(coor[j][2]-coor[k][2])*(coor[j][2]-coor[k][2]);
                            d2=Math.sqrt(aux1+aux2);
                            if(k==13 && i==2 && j== 2)
                            f.writeln("aux1 y aux2 ",aux1, " y ", aux2,"d1 y d2␣
↪",d1," ",d2);
                                  //Veo si está a menos de epsilon
                                  //if (Math.min(coor[i][1],coor[j][1])-eps <=␣
↪coor[k][1]
                                  //&& coor[k][1] <= Math.
↪max(coor[i][1],coor[j][1])+eps
```

```
                            //&& Math.abs((A[i][j]*coor[k][1] +␣
↪D[i][j]*coor[k][2] +
                            //C[i][j])/Math.sqrt((A[i][j])^2+(D[i][j])^2)) < eps)
                            if (dr<eps || d1<eps || d2 <eps)
                            {
                                        P[i][j].add(k);
                                        cardinal[i][j]=cardinal[i][j]+1; ␣
↪
                                    }
                        }
                }
        }
        //fclose();
}

// Defino variables de decision

dvar float+ x[cliente][ubicacion];//o dvar boolean x[cliente][ubicacion]; si la␣
↪asignación es única

dvar boolean y[ubicacion];//si la ubicacion esta abierta



minimize sum (i in cliente , j in ubicacion)  (distancia[i][j]*u[i]*x[i][j]) +␣
↪sum(j in ubicacion) f[j]*y[j];



subject to {

  //se atiende toda la demanda del cliente i

  forall (i in cliente)

    sum(j in ubicacion) x[i][j] == 1;

  //a un cliente solo se atiende desde ubicaciones abiertas

  forall (i in cliente , j in ubicacion)

    x[i][j] <= y[j];

  //cada ubicación atiende al menos la demanda mínima

  forall (j in ubicacion)
```

```
    sum(i in cliente) u[i]* x[i][j] >= b[j]*y[j];

  //cada ubicación atiende a lo mas la demanda máxima

  forall (j in ubicacion)

    sum(i in cliente) u[i]* x[i][j] <= B[j]*y[j]  ;


  // los clientes van a la instalacion abierta mas cercana
  // forall(i in cliente , j in ubicacion)
       // sum(k in ubicacion : distancia[i][k] <= distancia[i][j]) x[i][k] >=
↪y[j] ;


       //presupuesto de 600 para la apertura de instalaciones
   //sum(j in ubicacion) f[j]*y[j] <= 600 ;



  //todos los clientes cercanos a la ruta de i a j se asignen a la instalacion j

        //forall(i in cliente , j in ubicacion)

       //sum(k in P[i][j]) x[k][j] >= card(P[i][j]) * x[i][j] ;

}



main {

       var f = new IloOplOutputFile("solucionCAP.txt");

       thisOplModel.generate();

          var time0 = new Date();

       var OK = cplex.solve();

       var time1 = new Date();



       var tTotal = (time1.getTime()-time0.getTime())/1000;
```

```
        if (OK) {
                f.writeln("Instalaciones abiertas");
                for(var j in thisOplModel.ubicacion){
                                        if(thisOplModel.y[j] != 0){
                                        f.writeln("Abro instalacion ",j);
                                }
                }
                f.writeln("Asignacion de clientes");
                for(var j in thisOplModel.ubicacion){
                        for(var i in thisOplModel.cliente){
                                        if(thisOplModel.x[i][j] != 0){
                                        f.writeln("a la instalacion ",j," van␣
 ↪los clientes ",i);
                                }
                        }
                }
                f.writeln("Tiempo: ",tTotal,"segundos. Objetivo: "+cplex.
 ↪getObjValue( ));
                for(var i in thisOplModel.cliente){
                        for(var j in thisOplModel.ubicacion){
                        //f.writeln("punto ",i,"valor x: ",thisOplModel.
 ↪coor[i][1],"valor y: ",thisOplModel.coor[i][2]);
                        f.writeln("conjunto ",i,j,"cardinal :",thisOplModel.
 ↪cardinal[i][j]);
                        }
                }
        }
f.close();
}
```

[ ]:
```
/*********************************************
 * OPL 22.1.0.0 Data
 * Author: Usuario
 * Creation Date: 24 jun. 2022 at 14:55:37
 *********************************************/


//DATOS PARA LOS MODELOS CAP-U,CAP-M,CAP-U-C,CAP-U-C<600,CAP-M-RE,CAP-U-RE


 //Numero de nodos
 n=20;
 m=20;
 eps=1.5;
//p=4;
//Coordenadas de los nodos
coor=[[0.8 4]
[3.46 4.2]
[2.14 3.64]
[5.66 4.02]
[6.06 2.7]
[7.02 2.56]
[5.12 2.18]
[6.68 -0.46]
[5.88 -0.88]
[3.88 -0.36]
[0.1 -4.2]
[-0.74 -1.5]
[-1.1 -0.24]
[-1.28 0.58]
[-0.22 -0.1]
[-0.558 -0.22]
[1.42 1.26]
[1.482 -1.2]
[3.962 2.1]
[-0.198 2.98]
];

//Demanda en los puntos
u=[9,28,77,59,86,6,52,78,80,10,16,9,33,48,25,66,21,34,48,24];

//Capacidades mínimas y máximas

B=[100,100,100,100,450,100,450,100,450,100,100,450,450,450,450,450,100,100,100,100];
 ↪

b=[20,20,20,20,50,20,50,20,50,20,20,50,50,50,50,50,20,20,20,20];
```

```
//Costes fijos

f=[100,100,100,100,200,100,200,100,200,100,100,200,200,200,200,200,100,100,100,100];
 ↪

// distancia entre los puntos
distancia=[[0, 2.6675082 , 1.38751577, 4.86004115, 5.41826541,
        6.38451251, 4.68772866, 7.3801084 , 7.04420329, 5.33816448,
        8.22982381, 5.71153219, 4.6462458 , 4.00284899, 4.22497337,
        4.43312125, 2.80927037, 5.24453277, 3.68893535, 1.42702628],
       [2.6675082 , 0.        , 1.43387587, 2.20735135, 3.0016662 ,
        3.91959182, 2.61457454, 5.664274  , 5.62697077, 4.57930126,
        9.04707688, 7.08025423, 6.36452669, 5.96422669, 5.65971731,
        5.97333441, 3.57843541, 5.75086811, 2.15916743, 3.85608143],
       [1.38751577, 1.43387587, 0.        , 3.54045195, 4.03112887,
        4.99807963, 3.31843337, 6.11731967, 5.86668561, 4.36206373,
        8.10106166, 5.89185879, 5.05489861, 4.58911756, 4.42235232,
        4.70943776, 2.48652368, 4.8845229 , 2.38564121, 2.42937111],
       [4.86004115, 2.20735135, 3.54045195, 0.        , 1.37927517,
        1.99529446, 1.91760267, 4.59464906, 4.90493629, 4.72787479,
        9.92380975, 8.45165073, 7.99031914, 7.74578595, 7.1797493 ,
        7.52602976, 5.05916989, 6.68611128, 2.56312388, 5.949602  ],
       [5.41826541, 3.0016662 , 4.03112887, 1.37927517, 0.        ,
        0.97015463, 1.07424392, 3.22024844, 3.58452228, 3.75712656,
        9.11765321, 7.99249648, 7.74010336, 7.64002618, 6.87592903,
        7.23355542, 4.85831246, 6.01399069, 2.18210999, 6.26426085],
       [6.38451251, 3.91959182, 4.99807963, 1.99529446, 0.97015463,
        0.        , 1.93762742, 3.03907881, 3.62397572, 4.28788992,
        9.67388236, 8.75792213, 8.58920252, 8.53290103, 7.71318352,
        8.07183275, 5.74891294, 6.69380639, 3.09240424, 7.23020913],
       [4.68772866, 2.61457454, 3.31843337, 1.91760267, 1.07424392,
        1.93762742, 0.        , 3.06646376, 3.15296686, 2.82651729,
        8.11817714, 6.91968207, 6.67418909, 6.596969  , 5.80637581,
        6.16438837, 3.81266311, 4.96582762, 1.1607601 , 5.37783637],
       [7.3801084 , 5.664274  , 6.11731967, 4.59464906, 3.22024844,
        3.03907881, 3.06646376, 0.        , 0.90354856, 2.80178515,
        7.56861942, 7.49252961, 7.78310992, 8.02765221, 6.90938492,
        7.24197791, 5.53407626, 5.25040989, 3.73378146, 7.69028504],
       [7.04420329, 5.62697077, 5.86668561, 4.90493629, 3.58452228,
        3.62397572, 3.15296686, 0.90354856, 0.        , 2.06649462,
        6.66564325, 6.64896985, 7.00927956, 7.30733878, 6.14966666,
        6.47174196, 4.94683737, 4.40962629, 3.54388544, 7.20011694],
       [5.33816448, 4.57930126, 4.36206373, 4.72787479, 3.75712656,
        4.28788992, 2.82651729, 2.80178515, 2.06649462, 0.        ,
        5.3883207 , 4.75857121, 4.98144557, 5.24492135, 4.10823563,
        4.44020765, 2.94550505, 2.54086678, 2.46136629, 5.27121276],
```

```
      [8.22982381, 9.04707688, 8.10106166, 9.92380975, 9.11765321,
       9.67388236, 8.11817714, 7.56861942, 6.66564325, 5.3883207 ,
       0.        , 2.8276492 , 4.13782552, 4.97521859, 4.11246884,
       4.03402578, 5.61729472, 3.30301741, 7.38952258, 7.18618146],
      [5.71153219, 7.08025423, 5.89185879, 8.45165073, 7.99249648,
       8.75792213, 6.91968207, 7.49252961, 6.64896985, 4.75857121,
       2.8276492 , 0.        , 1.31041978, 2.14895323, 1.49345238,
       1.29287432, 3.50473965, 2.24216057, 5.92189193, 4.51266706],
      [4.6462458 , 6.36452669, 5.05489861, 7.99031914, 7.74010336,
       8.58920252, 6.67418909, 7.78310992, 7.00927956, 4.98144557,
       4.13782552, 1.31041978, 0.        , 0.83952367, 0.89106678,
       0.54236888, 2.93264386, 2.75469127, 5.57668755, 3.34395036],
      [4.00284899, 5.96422669, 4.58911756, 7.74578595, 7.64002618,
       8.53290103, 6.596969  , 8.02765221, 7.30733878, 5.24492135,
       4.97521859, 2.14895323, 0.83952367, 0.        , 1.25936492,
       1.07762888, 2.7843132 , 3.28588557, 5.45792671, 2.63262683],
      [4.22497337, 5.65971731, 4.42235232, 7.1797493 , 6.87592903,
       7.71318352, 5.80637581, 6.90938492, 6.14966666, 4.10823563,
       4.11246884, 1.49345238, 0.89106678, 1.25936492, 0.        ,
       0.35866976, 2.13053984, 2.0265251 , 4.72537025, 3.08007857],
      [4.43312125, 5.97333441, 4.70943776, 7.52602976, 7.23355542,
       8.07183275, 6.16438837, 7.24197791, 6.47174196, 4.44020765,
       4.03402578, 1.29287432, 0.54236888, 1.07762888, 0.35866976,
       0.        , 2.47040159, 2.2631836 , 5.08062988, 3.22018633],
      [2.80927037, 3.57843541, 2.48652368, 5.05916989, 4.85831246,
       5.74891294, 3.81266311, 5.53407626, 4.94683737, 2.94550505,
       5.61729472, 3.50473965, 2.93264386, 2.7843132 , 2.13053984,
       2.47040159, 0.        , 2.46078118, 2.67719331, 2.36142415],
      [5.24453277, 5.75086811, 4.8845229 , 6.68611128, 6.01399069,
       6.69380639, 4.96582762, 5.25040989, 4.40962629, 2.54086678,
       3.30301741, 2.24216057, 2.75469127, 3.28588557, 2.0265251 ,
       2.2631836 , 2.46078118, 0.        , 4.12800194, 4.50497503],
      [3.68893535, 2.15916743, 2.38564121, 2.56312388, 2.18210999,
       3.09240424, 1.1607601 , 3.73378146, 3.54388544, 2.46136629,
       7.38952258, 5.92189193, 5.57668755, 5.45792671, 4.72537025,
       5.08062988, 2.67719331, 4.12800194, 0.        , 4.25205833],
      [1.42702628, 3.85608143, 2.42937111, 5.949602  , 6.26426085,
       7.23020913, 5.37783637, 7.69028504, 7.20011694, 5.27121276,
       7.18618146, 4.51266706, 3.34395036, 2.63262683, 3.08007857,
       3.22018633, 2.36142415, 4.50497503, 4.25205833, 0.        ]];
```

[ ]:
```
/*********************************************
 * OPL 20.1.0.0 Model
 * Author: Usuario
 * Creation Date: 22 jun. 2022 at 18:17:02
 *********************************************/
```

```
//MODELOS CAP-U-RM Y CAP-M-RM


  // Defino las variables fijas

 int n = ...;//número de cliente
 int m = ...;//número de ubicaciones
 //int p=...; // número de instalaciones a ubicar
 float eps=...;
 range cliente=1..n;
 range ubicacion=1..m;
 range dimension=1..2;
 float B[cliente]= ...;
 float b[cliente]= ...;
 float u[cliente]= ...;  // numero de usuarios en cada centro de demanda
 float coor[cliente][dimension]= ...;
 float distancia[cliente][ubicacion]=...; // distancia entre los puntos
 float f[ubicacion]=...;
 float cardinal[cliente][ubicacion];

 //conjuntos Pij

 {int} P[i in cliente][j in ubicacion];



 //float coordenada[cliente][2];

 //float demanda[cliente];

 //float capacidad[ubicacion];

 //execute initializeArray{

 execute{

        for (var i in cliente){

                for(var j in ubicacion){

                //defino la distancia Manhattan

                distancia[i][j]=Math.abs(coor[i][1]-coor[j][1])+Math.
↪abs(coor[i][2]-coor[j][2]);
```

```
            //defino los conjuntos (4 esquinas primero, rectangulo del␣
↪medio, rectangulo de arriba, rectangulo de abajo,rectangulo izq,  rectangulo␣
↪derecho)

                    for(var k in cliente){

                                if (coor[k][1] <= Math.
↪min(coor[i][1],coor[j][1])

                                && coor[k][2] <= Math.
↪min(coor[i][2],coor[j][2])

                            && Math.abs(Math.
↪min(coor[i][1],coor[j][1])-coor[k][1])+Math.abs(Math.
↪min(coor[i][2],coor[j][2])-coor[k][2]) <= eps)
                                P[i][j].add(k);

                    }

                    for(var k in cliente){

                                if (coor[k][1] <= Math.
↪min(coor[i][1],coor[j][1])

                                && coor[k][2] >= Math.
↪max(coor[i][2],coor[j][2])

                            && Math.abs(Math.
↪min(coor[i][1],coor[j][1])-coor[k][1])+Math.abs(Math.
↪max(coor[i][2],coor[j][2])-coor[k][2]) <= eps )
                                P[i][j].add(k);
                    }

                    for(var k in cliente){

                                if (coor[k][1] >= Math.
↪max(coor[i][1],coor[j][1])

                                && coor[k][2] <= Math.
↪min(coor[i][2],coor[j][2])

                            && Math.abs(Math.
↪max(coor[i][1],coor[j][1])-coor[k][1])+Math.abs(Math.
↪min(coor[i][2],coor[j][2])-coor[k][2]) <=eps )

                                P[i][j].add(k);
```

```
                        }

                        for(var k in cliente){

                                if (coor[k][1] >= Math.
↪max(coor[i][1],coor[j][1])

                                && coor[k][2] >= Math.
↪max(coor[i][2],coor[j][2])

                            && Math.abs(Math.
↪max(coor[i][1],coor[j][1])-coor[k][1])+Math.abs(Math.
↪max(coor[i][2],coor[j][2])-coor[k][2]) <=eps )

                                P[i][j].add(k);

                        }

                        for(var k in cliente){

                                if (Math.min(coor[i][1],coor[j][1]) <=␣
↪coor[k][1]

                                && coor[k][1] <= Math.
↪max(coor[i][1],coor[j][1])

                            && Math.min(coor[i][2],coor[j][2]) <= coor[k][2]

                            && coor[k][2] <= Math.max(coor[i][2],coor[j][2]))

                                P[i][j].add(k);

                        }
                        for(var k in cliente){

                                if (Math.min(coor[i][1],coor[j][1]) <=␣
↪coor[k][1]

                                && coor[k][1] <= Math.
↪max(coor[i][1],coor[j][1])

                            && Math.max(coor[i][2],coor[j][2]) <= coor[k][2]

                            && coor[k][2] <= Math.max(coor[i][2],coor[j][2])+eps)
```

```
                                P[i][j].add(k);

                        }
                        for(var k in cliente){

                                if (Math.min(coor[i][1],coor[j][1]) <=␣
↪coor[k][1]

                                && coor[k][1] <= Math.
↪max(coor[i][1],coor[j][1])

                            && Math.min(coor[i][2],coor[j][2]) - eps <=␣
↪coor[k][2]

                            && coor[k][2] <= Math.min(coor[i][2],coor[j][2]))

                                P[i][j].add(k);

                        }
                        for(var k in cliente){

                                if (Math.min(coor[i][1],coor[j][1]) - eps <=␣
↪coor[k][1]

                                && coor[k][1] <= Math.
↪min(coor[i][1],coor[j][1])

                            && Math.min(coor[i][2],coor[j][2]) <= coor[k][2]

                            && coor[k][2] <= Math.max(coor[i][2],coor[j][2]))

                                P[i][j].add(k);

                        }
                        for(var k in cliente){

                                if (Math.max(coor[i][1],coor[j][1]) <=␣
↪coor[k][1]

                                && coor[k][1] <= Math.
↪max(coor[i][1],coor[j][1]) + eps

                            && Math.min(coor[i][2],coor[j][2]) <= coor[k][2]

                            && coor[k][2] <= Math.max(coor[i][2],coor[j][2]))
```

```
                                P[i][j].add(k);

                        }
                        cardinal[i][j]=Opl.card(P[i][j]);

                }

        }

}

 // Defino variables de decision

 dvar float+ x[cliente][ubicacion];//o dvar boolean x[cliente][ubicacion]; si la␣
 ↪asignación es única

 dvar boolean y[ubicacion];//si la ubicacion esta abierta



 minimize sum (i in cliente , j in ubicacion)  (distancia[i][j]*u[i]*x[i][j]) +␣
 ↪sum(j in ubicacion) f[j]*y[j];



 subject to {

   //se atiende toda la demanda del cliente i

   forall (i in cliente)

     sum(j in ubicacion) x[i][j] == 1;

   //a un cliente solo se atiende desde ubicaciones abiertas

   forall (i in cliente , j in ubicacion)

     x[i][j] <= y[j];

   //cada ubicación atiende al menos la demanda mínima

   forall (j in ubicacion)

     sum(i in cliente) u[i]* x[i][j] >= b[j]*y[j];

   //cada ubicación atiende a lo mas la demanda máxima
```

```
  forall (j in ubicacion)

    sum(i in cliente) u[i]* x[i][j] <= B[j]*y[j]  ;


  //todos los clientes cercanos a la ruta de i a j se asignen a la instalacion j

        forall(i in cliente , j in ubicacion)

       sum(k in P[i][j]) x[k][j] >= card(P[i][j]) * x[i][j] ;

}


  main {

       var f = new IloOplOutputFile("solucionCAP-M-R2.txt");

       thisOplModel.generate();

          var time0 = new Date();

       var OK = cplex.solve();

       var time1 = new Date();


       var tTotal = (time1.getTime()-time0.getTime())/1000;


       if (OK) {

               f.writeln("Instalaciones abiertas");

               for(var j in thisOplModel.ubicacion){

                                       if(thisOplModel.y[j] != 0){

                                       f.writeln("Abro instalacion ",j);

                                }

               }
```

```
                f.writeln("Asignacion de clientes");

                for(var j in thisOplModel.ubicacion){

                        for(var i in thisOplModel.cliente){

                                        if(thisOplModel.x[i][j] != 0){

                                        f.writeln("a la instalacion ",j," van␣
 ↪los clientes ",i);

                                }

                        }

                }

                f.writeln("Tiempo: ",tTotal,"segundos. Objetivo: "+cplex.
 ↪getObjValue( ));

                }
                for(var i in thisOplModel.cliente){
                        for(var j in thisOplModel.ubicacion){
                        //f.writeln("punto ",i,"valor x: ",thisOplModel.
 ↪coor[i][1],"valor y: ",thisOplModel.coor[i][2]);
                        f.writeln("conjunto ",i,j,"cardinal :",thisOplModel.
 ↪cardinal[i][j]);
                        }
                }
        f.close();

}
```

[ ]:
```
/*********************************************
 * OPL 20.1.0.0 Data
 * Author: Usuario
 * Creation Date: 22 jun. 2022 at 18:17:02
 *********************************************/


//DATOS DE LOS MODELOS CAP-U-RM Y CAP-M-RM


 //Numero de nodos
 n=20;
```

```
 m=20;
 eps=1.5;
//p=4;
//Coordenadas de los nodos
coor=[[0.8,4]
[3.46,4.2]
[2.14,3.64]
[5.66,4.02]
[6.06,2.7]
[7.02,2.56]
[5.12,2.18]
[6.68,-0.46]
[5.88,-0.88]
[3.88,-0.36]
[0.1,-4.2]
[-0.74,-1.5]
[-1.1,-0.24]
[-1.28,0.58]
[-0.22,-0.1]
[-0.558,-0.22]
[1.42,1.26]
[1.482,-1.2]
[3.962,2.1]
[-0.198,2.98]
];

//Demanda en los puntos
u=[9,28,77,59,86,6,52,78,80,10,16,9,33,48,25,66,21,34,48,24];

//Capacidades mínimas y máximas

B=[100,100,100,100,450,100,450,100,450,100,100,450,450,450,450,450,100,100,100,100];
 ↪

b=[20,20,20,20,50,20,50,20,50,20,20,50,50,50,50,50,20,20,20,20];

//Costes fijos

f=[100,100,100,100,200,100,200,100,200,100,100,200,200,200,200,200,100,100,100,100];
 ↪
```

4 Código y datos de los modelos jerarquizados en IBM ILOG CPLEX Optimization Studio

[ ]:
```
/*********************************************
 * OPL 22.1.0.0 Model
 * Author: Usuario
 * Creation Date: 24 jun. 2022 at 8:22:53
 *********************************************/

//ESCENARIOS DEL MODELO JERARQUIZADO



 // Defino las variables fijas
 int n = ...; // número de cliente
 int m = ...; // número de ubicaciones
 int s=...; // demanda

 range cliente=1..n;
 range ubicacion=1..m;
 range demanda=1..s;


 int B[cliente][demanda]= ...;  // capacidades máximas
 int b[cliente][demanda]= ...;  // capacidades mínimas
 int u[cliente][demanda]= ...;   //numero de usuarios en cada centro de demanda
 float distancia[cliente][ubicacion] = ... ; //distancia entre los puntos
 float D[demanda]=...;
 float f[ubicacion][demanda]=...;
 int p[demanda]=...;



  // Defino variables de decision
 dvar boolean x[cliente][ubicacion][demanda];
 dvar boolean y[ubicacion][demanda];
 dvar float+ z[ubicacion][demanda][demanda];

 minimize sum (i in cliente , j in ubicacion , s in demanda) ␣
 ↪(distancia[i][j]*u[i][s]*x[i][j][s]) + sum(j in ubicacion , s in demanda)␣
 ↪(f[j][s]*y[j][s]);

 subject to {

   forall (i in cliente , s in demanda)
     sum(j in ubicacion) x[i][j][s] == 1;
```

```
  forall (i in cliente , j in ubicacion , s in demanda)
    x[i][j][s] <=  sum(t in demanda : t >= s) y[j][t];

  forall (j in ubicacion , s in demanda)
    sum(t in demanda : t >= s) z[j][s][t] == sum(i in cliente)␣
↪u[i][s]*x[i][j][s];

  forall (j in ubicacion , t in demanda)
    sum(s in demanda : s <= t) z[j][s][t]  >= b[j][t]*y[j][t];

  forall (j in ubicacion , t in demanda)
    sum(s in demanda : s <= t) z[j][s][t]  <= B[j][t]*y[j][t];


  forall (i in cliente , j in ubicacion , s in demanda)
         forall(t in demanda : t >= s)
               sum(k in ubicacion : distancia[i][k] <= distancia[i][j])␣
↪x[i][k][s] >= y[j][t];


  forall(i in cliente, j in ubicacion, s in demanda : distancia[i][j] > D[s])
         x[i][j][s] == 0 ;


  forall(s in demanda)
    sum(j in ubicacion) y[j][s] <= p[s];


  }

  main {
       var f = new IloOplOutputFile("solucionjJER-0-(4,3).txt");
       thisOplModel.generate();
          var time0 = new Date();
       var OK = cplex.solve();
       var time1 = new Date();
       var tTotal = (time1.getTime()-time0.getTime())/1000;
       var instalacion1 = 0;
       var instalacion2 = 0;
       var instalacion3 = 0;
       var capacidad1 = 0;
       var capacidad2 = 0;
       var demanda1con2 = 0;

       if (OK) {
               f.writeln("Instalaciones abiertas");
               for(var j in thisOplModel.ubicacion){
```

```
                for(var s in thisOplModel.demanda){
                                        if(thisOplModel.y[j][s] != 0){
                                        f.writeln("Abro instalacion ", j , " de␣
↪tipo ", s);
                                }
                            }                   ␣
↪
                }
                f.writeln("Asignacion de clientes");
                for(var j in thisOplModel.ubicacion){
                        for(var i in thisOplModel.cliente){
                         for(var s in thisOplModel.demanda){
                                        if(thisOplModel.x[i][j][s] != 0){
                                        f.writeln("a la instalacion ", j ," van␣
↪los clientes ", i , " de nivel ", s);
                                }
                        }
        }
                }
                f.writeln("Tiempo: ",tTotal,"segundos. Objetivo: "+cplex.
↪getObjValue( ));
                for(var j in thisOplModel.ubicacion){
                                if(thisOplModel.y[j][1] != 0␣
↪&&        thisOplModel.y[j][2] != 0) instalacion3 = instalacion3+1;
                                else{
                                        if(thisOplModel.y[j][1] != 0)␣
↪instalacion1 = instalacion1+1;
                                        if(thisOplModel.y[j][2] != 0)␣
↪instalacion2 = instalacion2+1;
                                }
                        }
                f.writeln("abierto 1: ", instalacion1);
                f.writeln("abierto 2: ", instalacion2);
                f.writeln("abierto1+2: ", instalacion3);
                for(var j in thisOplModel.ubicacion){
                        if(thisOplModel.y[j][1] != 0) capacidad1 = capacidad1 +␣
↪thisOplModel.B[j][1];
                        if(thisOplModel.y[j][2] != 0) capacidad2 = capacidad2 +␣
↪thisOplModel.B[j][2];
                }
                f.writeln("capacidad máxima nivel 1: ", capacidad1);
                f.writeln("capacidad máxima nivel 2: ", capacidad2);
                for(var j in thisOplModel.ubicacion){
                                if(thisOplModel.z[j][1][2] != 0) demanda1con2 =␣
↪demanda1con2 + thisOplModel.z[j][1][2];
                }
```

```
                f.writeln("demanda de nivel 1 con nivel 2: ", demanda1con2);
        f.close();
}
}
```

[ ]:
```
/*********************************************
 * OPL 22.1.0.0 Data
 * Author: Usuario
 * Creation Date: 24 jun. 2022 at 8:22:53
 *********************************************/

//DATOS DE LOS MODELOS JERARQUIZADO
//Modificar p y D para resolver el escenario deseado



//Numero de nodos
 n=20;
 m=20;
 s=2;
 p=[4,3];

 //Tipos de instalacion (A,B,C o D):␣
 ↪tipos=[A,C,A,B,B,A,D,C,C,D,A,A,A,A,D,D,B,C,D,D]



 //Costes fijos costA=[100,500], costB=[250,1250], costC=[750,3750],␣
 ↪costD=[1000,5000]

 f=[[100,500],[750,3750],[100,500]
 ,[250,1250],[250,1250],[100,500]
 ,[1000,5000],[750,3750],[750,3750],[1000,5000]
 ,[100,500],[100,500]
␣
 ↪,[100,500],[100,500],[1000,5000],[1000,5000],[250,1250],[750,3750],[1000,5000],[1000,5000]];
 ↪



//Demanda en los puntos (aleatoria dentro de cada tipo de instalación que ha␣
 ↪tocado)
u=[[9, 1],[75, 30],[9, 1],[23, 8],[25, 7],[11, 1],[110, 45],[83, 33],[75,␣
 ↪27],[110, 41],[10, 1],[5, 9],[11, 1],[11, 1],[100, 45],[90, 45],[23, 8],[68,␣
 ↪30],[110, 41],[100, 50]];
```

```
//Capacidades mínimas y máximas

B=[[60,150],[450, 600],[60, 150],[150, 210],[150, 210],[60, 150],[600,␣
 ↪900],[450, 600],[450, 600],[600, 900],[60, 150],[60, 150],[60, 150],[60,␣
 ↪150],[600, 900],[600, 900],[150, 210],[450, 600],[600, 900],[600, 900]];

b=[[20, 50],[150, 200],[20, 50],[50, 70],[50, 70],[20, 50],[200, 300],[150,␣
 ↪200],[150, 200],[200, 300],[20, 50],[20, 50],[20, 50],[20, 50],[200,␣
 ↪300],[200, 300],[50, 70],[150, 200],[200, 300],[200, 300]];


//Distancia máxima a la instalación
D=[4,4];



//Distancia entre los nodos
distancia=[[0, 2.6675082 , 1.38751577, 4.86004115, 5.41826541,
        6.38451251, 4.68772866, 7.3801084 , 7.04420329, 5.33816448,
        8.22982381, 5.71153219, 4.6462458 , 4.00284899, 4.22497337,
        4.43312125, 2.80927037, 5.24453277, 3.68893535, 1.42702628],
       [2.6675082 , 0.        , 1.43387587, 2.20735135, 3.0016662 ,
        3.91959182, 2.61457454, 5.664274  , 5.62697077, 4.57930126,
        9.04707688, 7.08025423, 6.36452669, 5.96422669, 5.65971731,
        5.97333441, 3.57843541, 5.75086811, 2.15916743, 3.85608143],
       [1.38751577, 1.43387587, 0.        , 3.54045195, 4.03112887,
        4.99807963, 3.31843337, 6.11731967, 5.86668561, 4.36206373,
        8.10106166, 5.89185879, 5.05489861, 4.58911756, 4.42235232,
        4.70943776, 2.48652368, 4.8845229 , 2.38564121, 2.42937111],
       [4.86004115, 2.20735135, 3.54045195, 0.        , 1.37927517,
        1.99529446, 1.91760267, 4.59464906, 4.90493629, 4.72787479,
        9.92380975, 8.45165073, 7.99031914, 7.74578595, 7.1797493 ,
        7.52602976, 5.05916989, 6.68611128, 2.56312388, 5.949602  ],
       [5.41826541, 3.0016662 , 4.03112887, 1.37927517, 0.        ,
        0.97015463, 1.07424392, 3.22024844, 3.58452228, 3.75712656,
        9.11765321, 7.99249648, 7.74010336, 7.64002618, 6.87592903,
        7.23355542, 4.85831246, 6.01399069, 2.18210999, 6.26426085],
       [6.38451251, 3.91959182, 4.99807963, 1.99529446, 0.97015463,
        0.        , 1.93762742, 3.03907881, 3.62397572, 4.28788992,
        9.67388236, 8.75792213, 8.58920252, 8.53290103, 7.71318352,
        8.07183275, 5.74891294, 6.69380639, 3.09240424, 7.23020913],
       [4.68772866, 2.61457454, 3.31843337, 1.91760267, 1.07424392,
```

```
       1.93762742, 0.        , 3.06646376, 3.15296686, 2.82651729,
       8.11817714, 6.91968207, 6.67418909, 6.596969  , 5.80637581,
       6.16438837, 3.81266311, 4.96582762, 1.1607601 , 5.37783637],
      [7.3801084 , 5.664274  , 6.11731967, 4.59464906, 3.22024844,
       3.03907881, 3.06646376, 0.        , 0.90354856, 2.80178515,
       7.56861942, 7.49252961, 7.78310992, 8.02765221, 6.90938492,
       7.24197791, 5.53407626, 5.25040989, 3.73378146, 7.69028504],
      [7.04420329, 5.62697077, 5.86668561, 4.90493629, 3.58452228,
       3.62397572, 3.15296686, 0.90354856, 0.        , 2.06649462,
       6.66564325, 6.64896985, 7.00927956, 7.30733878, 6.14966666,
       6.47174196, 4.94683737, 4.40962629, 3.54388544, 7.20011694],
      [5.33816448, 4.57930126, 4.36206373, 4.72787479, 3.75712656,
       4.28788992, 2.82651729, 2.80178515, 2.06649462, 0.        ,
       5.3883207 , 4.75857121, 4.98144557, 5.24492135, 4.10823563,
       4.44020765, 2.94550505, 2.54086678, 2.46136629, 5.27121276],
      [8.22982381, 9.04707688, 8.10106166, 9.92380975, 9.11765321,
       9.67388236, 8.11817714, 7.56861942, 6.66564325, 5.3883207 ,
       0.        , 2.8276492 , 4.13782552, 4.97521859, 4.11246884,
       4.03402578, 5.61729472, 3.30301741, 7.38952258, 7.18618146],
      [5.71153219, 7.08025423, 5.89185879, 8.45165073, 7.99249648,
       8.75792213, 6.91968207, 7.49252961, 6.64896985, 4.75857121,
       2.8276492 , 0.        , 1.31041978, 2.14895323, 1.49345238,
       1.29287432, 3.50473965, 2.24216057, 5.92189193, 4.51266706],
      [4.6462458 , 6.36452669, 5.05489861, 7.99031914, 7.74010336,
       8.58920252, 6.67418909, 7.78310992, 7.00927956, 4.98144557,
       4.13782552, 1.31041978, 0.        , 0.83952367, 0.89106678,
       0.54236888, 2.93264386, 2.75469127, 5.57668755, 3.34395036],
      [4.00284899, 5.96422669, 4.58911756, 7.74578595, 7.64002618,
       8.53290103, 6.596969  , 8.02765221, 7.30733878, 5.24492135,
       4.97521859, 2.14895323, 0.83952367, 0.        , 1.25936492,
       1.07762888, 2.7843132 , 3.28588557, 5.45792671, 2.63262683],
      [4.22497337, 5.65971731, 4.42235232, 7.1797493 , 6.87592903,
       7.71318352, 5.80637581, 6.90938492, 6.14966666, 4.10823563,
       4.11246884, 1.49345238, 0.89106678, 1.25936492, 0.        ,
       0.35866976, 2.13053984, 2.0265251 , 4.72537025, 3.08007857],
      [4.43312125, 5.97333441, 4.70943776, 7.52602976, 7.23355542,
       8.07183275, 6.16438837, 7.24197791, 6.47174196, 4.44020765,
       4.03402578, 1.29287432, 0.54236888, 1.07762888, 0.35866976,
       0.        , 2.47040159, 2.2631836 , 5.08062988, 3.22018633],
      [2.80927037, 3.57843541, 2.48652368, 5.05916989, 4.85831246,
       5.74891294, 3.81266311, 5.53407626, 4.94683737, 2.94550505,
       5.61729472, 3.50473965, 2.93264386, 2.7843132 , 2.13053984,
       2.47040159, 0.        , 2.46078118, 2.67719331, 2.36142415],
      [5.24453277, 5.75086811, 4.8845229 , 6.68611128, 6.01399069,
       6.69380639, 4.96582762, 5.25040989, 4.40962629, 2.54086678,
       3.30301741, 2.24216057, 2.75469127, 3.28588557, 2.0265251 ,
       2.2631836 , 2.46078118, 0.        , 4.12800194, 4.50497503],
```

```
       [3.68893535, 2.15916743, 2.38564121, 2.56312388, 2.18210999,
        3.09240424, 1.1607601 , 3.73378146, 3.54388544, 2.46136629,
        7.38952258, 5.92189193, 5.57668755, 5.45792671, 4.72537025,
        5.08062988, 2.67719331, 4.12800194, 0.        , 4.25205833],
       [1.42702628, 3.85608143, 2.42937111, 5.949602  , 6.26426085,
        7.23020913, 5.37783637, 7.69028504, 7.20011694, 5.27121276,
        7.18618146, 4.51266706, 3.34395036, 2.63262683, 3.08007857,
        3.22018633, 2.36142415, 4.50497503, 4.25205833, 0.        ]];
```

5 Cálculo de la distancia Euclídea entre los puntos con Python

[2]:
```
import math
import numpy as np
```

[3]:
```
p1=(0.8,4)
p2= (3.46,4.2)
p3= (2.14,3.64)
p4= (5.66,4.02)
p5= (6.06,2.7)
p6= (7.02,2.56)
p7= (5.12,2.18)
p8= (6.68,-0.46)
p9= (5.88,-0.88)
p10= (3.88,-0.36)
p11= (0.1,-4.2)
p12= (-0.74,-1.5)
p13= (-1.1,-0.24)
p14= (-1.28,0.58)
p15= (-0.22,-0.1)
p16= (-0.558,-0.22)
p17= (1.42,1.26)
p18= (1.482,-1.2)
p19= (3.962,2.1)
p20= (-0.198,2.98)
```

[4]:
```
puntos=[p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,p13,p14,p15,p16,p17,p18,p19,p20]
```

[5]:
```
def distancia(punto1,punto2):

```

```
    distancia = math.sqrt((punto2[0]-punto1[0])**2+(punto2[1]-punto1[1])**2)
    return distancia
```

[6]:
```
L=[]
for i in range(0,20):
    for j in range(0,20):
        distancias=distancia(puntos[i],puntos[j])

        L.append(distancias)
```

[7]:
```
D=np.array(L).reshape(20,20)
D
```

[7]:
```
array([[0.        , 2.6675082 , 1.38751577, 4.86004115, 5.41826541,
        6.38451251, 4.68772866, 7.3801084 , 7.04420329, 5.33816448,
        8.22982381, 5.71153219, 4.6462458 , 4.00284899, 4.22497337,
        4.43312125, 2.80927037, 5.24453277, 3.68893535, 1.42702628],
       [2.6675082 , 0.        , 1.43387587, 2.20735135, 3.0016662 ,
        3.91959182, 2.61457454, 5.664274  , 5.62697077, 4.57930126,
        9.04707688, 7.08025423, 6.36452669, 5.96422669, 5.65971731,
        5.97333441, 3.57843541, 5.75086811, 2.15916743, 3.85608143],
       [1.38751577, 1.43387587, 0.        , 3.54045195, 4.03112887,
        4.99807963, 3.31843337, 6.11731967, 5.86668561, 4.36206373,
        8.10106166, 5.89185879, 5.05489861, 4.58911756, 4.42235232,
        4.70943776, 2.48652368, 4.8845229 , 2.38564121, 2.42937111],
       [4.86004115, 2.20735135, 3.54045195, 0.        , 1.37927517,
        1.99529446, 1.91760267, 4.59464906, 4.90493629, 4.72787479,
        9.92380975, 8.45165073, 7.99031914, 7.74578595, 7.1797493 ,
        7.52602976, 5.05916989, 6.68611128, 2.56312388, 5.949602  ],
       [5.41826541, 3.0016662 , 4.03112887, 1.37927517, 0.        ,
        0.97015463, 1.07424392, 3.22024844, 3.58452228, 3.75712656,
        9.11765321, 7.99249648, 7.74010336, 7.64002618, 6.87592903,
        7.23355542, 4.85831246, 6.01399069, 2.18210999, 6.26426085],
       [6.38451251, 3.91959182, 4.99807963, 1.99529446, 0.97015463,
        0.        , 1.93762742, 3.03907881, 3.62397572, 4.28788992,
        9.67388236, 8.75792213, 8.58920252, 8.53290103, 7.71318352,
        8.07183275, 5.74891294, 6.69380639, 3.09240424, 7.23020913],
       [4.68772866, 2.61457454, 3.31843337, 1.91760267, 1.07424392,
        1.93762742, 0.        , 3.06646376, 3.15296686, 2.82651729,
        8.11817714, 6.91968207, 6.67418909, 6.596969  , 5.80637581,
        6.16438837, 3.81266311, 4.96582762, 1.1607601 , 5.37783637],
       [7.3801084 , 5.664274  , 6.11731967, 4.59464906, 3.22024844,
        3.03907881, 3.06646376, 0.        , 0.90354856, 2.80178515,
        7.56861942, 7.49252961, 7.78310992, 8.02765221, 6.90938492,
        7.24197791, 5.53407626, 5.25040989, 3.73378146, 7.69028504],
       [7.04420329, 5.62697077, 5.86668561, 4.90493629, 3.58452228,
        3.62397572, 3.15296686, 0.90354856, 0.        , 2.06649462,
```

```
 6.66564325, 6.64896985, 7.00927956, 7.30733878, 6.14966666,
 6.47174196, 4.94683737, 4.40962629, 3.54388544, 7.20011694],
[5.33816448, 4.57930126, 4.36206373, 4.72787479, 3.75712656,
 4.28788992, 2.82651729, 2.80178515, 2.06649462, 0.        ,
 5.3883207 , 4.75857121, 4.98144557, 5.24492135, 4.10823563,
 4.44020765, 2.94550505, 2.54086678, 2.46136629, 5.27121276],
[8.22982381, 9.04707688, 8.10106166, 9.92380975, 9.11765321,
 9.67388236, 8.11817714, 7.56861942, 6.66564325, 5.3883207 ,
 0.        , 2.8276492 , 4.13782552, 4.97521859, 4.11246884,
 4.03402578, 5.61729472, 3.30301741, 7.38952258, 7.18618146],
[5.71153219, 7.08025423, 5.89185879, 8.45165073, 7.99249648,
 8.75792213, 6.91968207, 7.49252961, 6.64896985, 4.75857121,
 2.8276492 , 0.        , 1.31041978, 2.14895323, 1.49345238,
 1.29287432, 3.50473965, 2.24216057, 5.92189193, 4.51266706],
[4.6462458 , 6.36452669, 5.05489861, 7.99031914, 7.74010336,
 8.58920252, 6.67418909, 7.78310992, 7.00927956, 4.98144557,
 4.13782552, 1.31041978, 0.        , 0.83952367, 0.89106678,
 0.54236888, 2.93264386, 2.75469127, 5.57668755, 3.34395036],
[4.00284899, 5.96422669, 4.58911756, 7.74578595, 7.64002618,
 8.53290103, 6.596969  , 8.02765221, 7.30733878, 5.24492135,
 4.97521859, 2.14895323, 0.83952367, 0.        , 1.25936492,
 1.07762888, 2.7843132 , 3.28588557, 5.45792671, 2.63262683],
[4.22497337, 5.65971731, 4.42235232, 7.1797493 , 6.87592903,
 7.71318352, 5.80637581, 6.90938492, 6.14966666, 4.10823563,
 4.11246884, 1.49345238, 0.89106678, 1.25936492, 0.        ,
 0.35866976, 2.13053984, 2.0265251 , 4.72537025, 3.08007857],
[4.43312125, 5.97333441, 4.70943776, 7.52602976, 7.23355542,
 8.07183275, 6.16438837, 7.24197791, 6.47174196, 4.44020765,
 4.03402578, 1.29287432, 0.54236888, 1.07762888, 0.35866976,
 0.        , 2.47040159, 2.2631836 , 5.08062988, 3.22018633],
[2.80927037, 3.57843541, 2.48652368, 5.05916989, 4.85831246,
 5.74891294, 3.81266311, 5.53407626, 4.94683737, 2.94550505,
 5.61729472, 3.50473965, 2.93264386, 2.7843132 , 2.13053984,
 2.47040159, 0.        , 2.46078118, 2.67719331, 2.36142415],
[5.24453277, 5.75086811, 4.8845229 , 6.68611128, 6.01399069,
 6.69380639, 4.96582762, 5.25040989, 4.40962629, 2.54086678,
 3.30301741, 2.24216057, 2.75469127, 3.28588557, 2.0265251 ,
 2.2631836 , 2.46078118, 0.        , 4.12800194, 4.50497503],
[3.68893535, 2.15916743, 2.38564121, 2.56312388, 2.18210999,
 3.09240424, 1.1607601 , 3.73378146, 3.54388544, 2.46136629,
 7.38952258, 5.92189193, 5.57668755, 5.45792671, 4.72537025,
 5.08062988, 2.67719331, 4.12800194, 0.        , 4.25205833],
[1.42702628, 3.85608143, 2.42937111, 5.949602  , 6.26426085,
 7.23020913, 5.37783637, 7.69028504, 7.20011694, 5.27121276,
 7.18618146, 4.51266706, 3.34395036, 2.63262683, 3.08007857,
 3.22018633, 2.36142415, 4.50497503, 4.25205833, 0.        ]])
```

6 Datos aleatorios para el problema jerarquizado con Python

[12]:
```
import math
import numpy as np
import random
```

[20]:
```
A=[[10,9,11],[1,0,2]]
B=[[25,23,28],[8,7,9]]
C=[[75,68,83],[30,27,33]]
D=[[100,90,110],[45,41,50]]
```

[19]:
```
costA=[100,500]
costB=[250,1250]
costC=[750,3750]
costD=[1000,5000]
```

[14]:
```
mMA=[[50,200],[150,500]]
mMB=[[20,150],[60,300]]
mMC=[[10,60],[50,250]]
mMD=[[50,300],[200,600]]
```

[15]:
```
tipos=[]
```

[16]:
```
for i in range(0,20):
    tipos.append(chr(random.randint(ord('A'), ord('D'))))
```

[17]:
```
u=[[random.choice(A[0]),random.choice(A[1])],
   [random.choice(C[0]),random.choice(C[1])],
   [random.choice(A[0]),random.choice(A[1])],
   [random.choice(B[0]),random.choice(B[1])],
   [random.choice(B[0]),random.choice(B[1])],
   [random.choice(A[0]),random.choice(A[1])],
   [random.choice(D[0]),random.choice(D[1])],
   [random.choice(C[0]),random.choice(C[1])],
   [random.choice(C[0]),random.choice(C[1])],
   [random.choice(D[0]),random.choice(D[1])],
   [random.choice(A[0]),random.choice(A[1])],
   [random.choice(B[0]),random.choice(B[1])],
   [random.choice(A[0]),random.choice(A[1])],
   [random.choice(A[0]),random.choice(A[1])],
   [random.choice(D[0]),random.choice(D[1])],
   [random.choice(D[0]),random.choice(D[1])],
   [random.choice(B[0]),random.choice(B[1])],
   [random.choice(C[0]),random.choice(C[1])],
   [random.choice(D[0]),random.choice(D[1])],
   [random.choice(D[0]),random.choice(D[1])]]
```

[21]:
```
f=[costA,
 costC,
 costA,
 costB,
 costB,
 costA,
 costD,
 costC,
 costC,
 costD,
 costA,
 costA,
 costA,
 costA,
 costD,
 costD,
 costB,
 costC,
 costD,
 costD]
```

[22]:
```
b=[mMA[0],
 mMC[0],
 mMA[0],
 mMB[0],
 mMB[0],
 mMA[0],
 mMD[0],
 mMC[0],
 mMC[0],
 mMD[0],
 mMA[0],
 mMA[0],
 mMA[0],
 mMA[0],
 mMA[0],
 mMD[0],
 mMD[0],
 mMB[0],
 mMC[0],
 mMD[0],
 mMD[0]]
```

[23]:
```
B=[mMA[1],
 mMC[1],
 mMA[1],
 mMB[1],
 mMB[1],
```

```
mMA[1],
mMD[1],
mMC[1],
mMC[1],
mMD[1],
mMA[1],
mMA[1],
mMA[1],
mMA[1],
mMD[1],
mMD[1],
mMB[1],
mMC[1],
mMD[1],
mMD[1]]
```

[ ]: