Resumen

Este volumen incluye los anexos A, B y C. En ellos se encuentra información relacionada con el proyecto que se considera de interés para el lector, ya que completa lo ya expuesto.

El anexo A muestra los modelos matemáticos presentados en el proyecto en lenguaje OPL, profundizando así en la modelización en este lenguaje de programación requerido por el optimizador ILOG. En el anexo B se presenta el programa utilizado para la generación de los ejemplares utilizados en la experimentación, desarrollado mediante el lenguaje de programación Java. En el anexo C se muestran los tiempos necesarios para la resolución de los diferentes modelos para los 675 ejemplares generados.

Sumario

A Modelos de planificación y replanificación en lenguaje OPL

A continuación se muestran los modelos matemáticos presentados anteriormente traducidos al lenguaje OPL (Optimization Programming Language). Este lenguaje permite introducir modelos matemáticos en el programa optimizador ILOG, donde son resueltos.

A.1 Modelo de planificación M1

```
//Tipos para índices

tuple i_t {
  int i;
  int t;
};

tuple t_j_k {
  int t;
  int j;
  int k;
};

//Datos

int T= ...;
{int} Periodos = asSet (1..T);
{int} E= ...;
{int} J= ...;
{int} K= ...;
{int} Ej[J]= ...;
float rtk[Periodos][K]= ...;
{int} Ai[E]= ...;
{int} vac[i in E]= Periodos diff Ai[i];
float Hi[E]= ...;
float penalizacion[J][K]= ...;
float eficiencia[J][K]= ...;
{int} Ck[K]= ...;
{int} Fj[J]= ...;
float landa= ...;
float alfa= ...;
```

```
float betai[E]= ...;
float gammak[K]= ...;
float hm= ...;
float hM= ...;
int L= ...;
int hL= ...;
int S= ...;
int hS= ...;
int W= ...;
int hW= ...;

float sitdato[E][Periodos]= ...;
float witdato[E][Periodos]= ...;
float vdato[E]= ...;
float z1= ...;
float epsilon= ...;
float W1= ...;
float W2= ...;
float xmedia[E]= ...;
float dmedia[K]= ...;

float treplan= ...;
float Trigidez= ...;
float rtknueva [Periodos][K]= ...;
float epsilon_replan= ...;
float nsmax= ...;
float ncmax= ...;
float ntmax= ...;
float tsi= ...;
float D= ...;
float Nf= ...;
float gamma= ...;
float x1[E][Periodos]= ...;
float x2[E][Periodos]= ...;
float wit1[E][Periodos]= ...;
float wit2[E][Periodos]= ...;
float sit1[E][Periodos]= ...;
float sit2[E][Periodos]= ...;
float Zprima[K]= ...;


//Variables

{i_t} indices_a = {<i,t> | i in E, t in Ai[i]};
{t_j_k} indices_b = {<t,j,k> | t in Periodos, j in J, k in K:j in Ck[k]};

dvar float x[i in indices_a];
```

```
dvar float y[i in indices_b];
dvar float+ v[E];
dvar float+ d[Periodos][K];
dvar int sit[i in indices_a] in 0..1;
dvar int wit[i in indices_a] in 0..1;

//Función objetivo

minimize
   (sum (i in E) betai[i] * v[i])
   +
    (sum (k in K) gammak[k]* ( sum (t in Periodos) d[t][k]))
   +
       (landa * sum (t in Periodos) sum ( k in K) sum (j in Ck[k])
        penalizacion[j][k]* y[<t,j,k>] )
 ;

//Restricciones

subject to {

//Restricción 2: horas requeridas se satisfacen entre los empleados de la
empresa y los externos
      forall (t in Periodos)
       forall (k in K)
          sum(j in Ck[k]) (eficiencia[j][k]*y[<t,j,k>])
           + d[t][k] >= rtk[t][k];

//Restricción 3: cota inferior de horas semanales
  forall (i in E)
 forall (t in Ai[i])
  x[<i,t>] >= hm;

//Restricción 3bis: cota superior de horas semanales
 forall (i in E)
 forall (t in Ai[i])
   x[<i,t>] <= hM;

//Restricción 4: cota del número de horas extras
  forall (i in E)
  v[i] <= alfa * Hi[i] ;

//Restricción 5 (horas trabajadas=horas extra+horas estipuladas)
      forall (i in E)
         sum(t in Ai[i]) x[<i,t>] == Hi[i]+v[i];

//Restricción 6
```

```
        forall(t in Periodos, j in J)
            sum(k in Fj[j]) y[<t,j,k>] ==
            sum(i in Ej[j]:t in Ai[i])  x[<i,t>] ;

//Restricción 7

      forall(i in E, tau in L..item(vac[i],0)-1)
      sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;

      forall(i in E, tau in item(vac[i],1)+L..item(vac[i],2)-1)
      sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;

      forall(i in E, tau in item(vac[i],5)+L..T)
      sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;


//Restricción 8,9: semanas fuertes
      forall(<i,t> in indices_a)
      x[<i,t>]<=(hS+(hM-hS)*sit[<i,t>]);

       forall(i in E)
      sum(t in Ai[i])(sit[<i,t>])
       <=S;

//Restricción 10,11: semanas flojas
       forall(<i,t> in indices_a)
       x[<i,t>]<=(hM-(hM-hW)*wit[<i,t>]);

       forall(i in E)
       sum(t in Ai[i]) wit[<i,t>]
       >=W;

//Restricción 15 : y(t,j,k) siempre positiva
     forall(<t,j,k> in indices_b)
     y[<t,j,k>]>=0;

}
```

A.2 Modelo de planificación M2

```
//Tipos para índices

tuple i_t {
  int i;
  int t;
```

```
};

tuple t_j_k {
  int t;
  int j;
  int k;
};

//Datos


int T= ...;
{int} Periodos = asSet (1..T);
{int} E= ...;
{int} J= ...;
{int} K= ...;
{int} Ej[J]= ...;
float rtk[Periodos][K]= ...;
{int} Ai[E]= ...;
{int} vac[i in E]= Periodos diff Ai[i];
float Hi[E]= ...;
float penalizacion[J][K]= ...;
float eficiencia[J][K]= ...;
{int} Ck[K]= ...;
{int} Fj[J]= ...;
float landa= ...;
float alfa= ...;
float betai[E]= ...;
float gammak[K]= ...;
float hm= ...;
float hM= ...;
int L= ...;
int hL= ...;
int S= ...;
int hS= ...;
int W= ...;
int hW= ...;

float sitdato[E][Periodos]= ...;
float witdato[E][Periodos]= ...;
float vdato[E]= ...;
float z1= ...;
float epsilon= ...;
float W1= ...;
float W2= ...;
float xmedia[E]= ...;
float dmedia[K]= ...;
```

```
float treplan= ...;
float Trigidez= ...;
float rtknueva [Periodos][K]= ...;
float epsilon_replan= ...;
float nsmax= ...;
float ncmax= ...;
 float ntmax= ...;
float tsi= ...;
float D= ...;
float hD= ...;
float Nf= ...;
float Zprima[K]= ...;
float gamma= ...;
float x1[E][Periodos]= ...;
float x2[E][Periodos]= ...;
float wit1[E][Periodos]= ...;
float wit2[E][Periodos]= ...;
float sit1[E][Periodos]= ...;
float sit2[E][Periodos]= ...;


//Variables

{i_t} indices_a = {<i,t> | i in E, t in Ai[i]};
{t_j_k} indices_b = {<t,j,k> | t in Periodos, j in J, k in K:j in Ck[k]};

dvar float x[i in indices_a];
dvar float y[i in indices_b];
dvar float+ d[Periodos][K];

dvar float+ deltapos[i in indices_a];
dvar float deltaneg[i in indices_a];
dvar float+ gammapos[Periodos][K];
dvar float gammaneg[Periodos][K];

//Función objetivo

minimize

W1* (
  sum (<i,t> in indices_a)
  (deltapos[<i,t>] - deltaneg[<i,t>])
   )

+W2* (
  sum(t in Periodos)
```

```
   sum(k in K)
   (gammapos[t][k]-gammaneg[t][k])
)

;

//Restricciones

subject to {

//Restricción 2: horas requeridas se satisfacen entre los empleados de la
empresa y los externos
      forall (t in Periodos)
        forall (k in K)
          sum(j in Ck[k]) (eficiencia[j][k]*y[<t,j,k>])
           + d[t][k] >= rtk[t][k];

//Restricción 3: cota inferior de horas semanales
   forall (i in E)
  forall (t in Ai[i])
   x[<i,t>] >= hm;

//Restricción 3bis: cota superior de horas semanales
  forall (i in E)
  forall (t in Ai[i])
    x[<i,t>] <= hM;

//Restricción 4: cota del número de horas extras

//Restricción 5: horas trabajadas=horas extra+horas estipuladas
      forall (i in E)
         sum(t in Ai[i]) x[<i,t>] == Hi[i]+vdato[i];

//Restricción 6: relación entre x,y
        forall(t in Periodos, j in J)
            sum(k in Fj[j]) y[<t,j,k>] ==
            sum(i in Ej[j]:t in Ai[i])  x[<i,t>] ;

//Restricción 7

      forall(i in E, tau in L..item(vac[i],0)-1)
      sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;

      forall(i in E, tau in item(vac[i],1)+L..item(vac[i],2)-1)
      sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;

      forall(i in E, tau in item(vac[i],5)+L..T)
```

```
      sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;


//Restricción 8,9: semanas fuertes

//Restricción 10,11: semanas flojas

//Restricción 15 : y(t,j,k) siempre positiva
     forall(<t,j,k> in indices_b)
     y[<t,j,k>]>=0;


// Restricción 17: el coste no puede ser superior al del modelo M1

(sum (i in E) betai[i] * vdato[i])
   +
(sum (k in K) gammak[k]* ( sum (t in Periodos) d[t][k]))
   +
(landa * sum (t in Periodos) sum ( k in K) sum (j in Ck[k])
         penalizacion[j][k]* y[<t,j,k>] )
  <= z1 + epsilon ;


// Restricción 18,19: Definición de las variables xit
     forall (<i,t> in indices_a)
     x[<i,t>]== xmedia[i]+
     deltapos[<i,t>]+
     deltaneg[<i,t>];

//Restricción 20,21: definición de las variables dtk
     forall (t in Periodos, k in K)
     d[t][k]==dmedia[k]+gammapos[t][k]+gammaneg[t][k];

//Restricciones 24 y 27 : deltaneg[i,t] y gammaneg[Periodos][K] siempre
negativas
     forall(t in Periodos, k in K)
     gammaneg[t][k]<=0;

     forall(<i,t> in indices_a)
     deltaneg[<i,t>]<=0;
}
```

A.3 Modelos de replanificación

A.3.1 Modelo de la fase I (R1)

```
//Tipos para índices

tuple i_t {
  int i;
  int t;
};

tuple t_j_k {
  int t;
  int j;
  int k;
};

//Datos

int T= ...;
{int} Periodos = asSet (1..T);
{int} E= ...;
{int} J= ...;
{int} K= ...;
{int} Ej[J]= ...;
float rtk[Periodos][K]= ...;
{int} Ai[E]= ...;
{int} vac[i in E]= Periodos diff Ai[i];

float Hi[E]= ...;
float penalizacion[J][K]= ...;
float eficiencia[J][K]= ...;
{int} Ck[K]= ...;
{int} Fj[J]= ...;
float landa= ...;
float alfa= ...;
float betai[E]= ...;
float gammak[K]= ...;
float hm= ...;
float hM= ...;
int L= ...;
int hL= ...;
int S= ...;
int hS= ...;
```

```
int W= ...;
int hW= ...;

float sitdato[E][Periodos]= ...;
float witdato[E][Periodos]= ...;
float vdato[E]= ...;

float z1= ...;
float epsilon= ...;
float W1= ...;
float W2= ...;
float xmedia[E]= ...;
float dmedia[K]= ...;
int treplan= ...;
int Trigidez=...;
{int} Preplan = asSet (treplan+Trigidez..T);
{int} Pfijo = asSet (1..treplan+Trigidez-1);

float x1[E][Pfijo]= ...;
float x2[E][Preplan]= ...;
float sit1[E][Pfijo]= ...;
float sit2[E][Preplan]= ...;
float wit1[E][Pfijo]= ...;
float wit2[E][Preplan]= ...;
float HRi[i in E]= Hi[i] - (sum (t in Pfijo) x1[i][t]);
float rtknueva[Preplan][K]= ...;
float epsilon_replan= ...;
int nsmax= ...;
int ncmax= ...;
int ntmax= ...;
float tsi= ...;
int D= ...;
int hD= ...;
int Nf= ...;
float Zprima[K]= ...;
float gamma= ...;

//Variables

{i_t} indices_a = {<i,t> | i in E, t in Preplan: t in Ai[i]};
{t_j_k} indices_b = {<t,j,k> | t in Preplan, j in J, k in K:j in Ck[k]};

dvar float x[i in indices_a];
dvar float y[i in indices_b];
dvar float+ v[E];
dvar float+ d[Preplan][K];
dvar int sit[i in indices_a] in 0..1;
```

```
dvar int wit[i in indices_a] in 0..1;
dvar int cambio[i in indices_a] in 0..1;
dvar float xcpos[i in indices_a];
dvar float xcneg[i in indices_a];
dvar int xfuerte[i in indices_a] in 0..1;

//Función objetivo

minimize
   (sum (i in E) betai[i] * v[i])
   +
    (sum (k in K) gammak[k]* ( sum (t in Preplan) d[t][k]))
   +
        (landa * sum (t in Preplan) sum ( k in K) sum (j in Ck[k])
         penalizacion[j][k]* y[<t,j,k>] )

 ;

//Restricciones

subject to {

//Restricción 2: horas requeridas se satisfacen entre los empleados de la
empresa y los externos
      forall (t in Preplan)
        forall (k in K)
          sum(j in Ck[k]) (eficiencia[j][k]*y[<t,j,k>])
           + d[t][k] >= rtknueva[t][k];

//Restricción 3: cota inferior de horas semanales
   forall (i in E)
  forall (t in Preplan: t in Ai[i])
   x[<i,t>] >= hm;

//Restricción 3bis: cota superior de horas semanales
  forall (i in E)
  forall (t in Preplan: t in Ai[i])
    x[<i,t>] <= hM;

//Restricción 4: cota del número de horas extras
   forall (i in E)
   v[i] <= alfa * Hi[i] ;

//Restricción 5 (horas trabajadas=horas extra+horas estipuladas)
      forall (i in E)
         sum(t in Preplan: t in Ai[i]) x[<i,t>] == HRi[i]+v[i];
```

```
//Restricción 6
      forall(t in Preplan, j in J)
           sum(k in Fj[j]) y[<t,j,k>] ==
           sum(i in Ej[j]:t in Ai[i])  x[<i,t>] ;

//Restricción 7


     forall(i in E, tau in L..item(vac[i],0)-1)
     if (tau-L+1>=9){
     sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
     }
     else {
     sum (t in 9..tau) x[<i,t>] + sum (t in tau-L+1..8) x1[i][t]<= L * hL;
       }

     forall(i in E, tau in item(vac[i],1)+L..item(vac[i],2)-1)
     if (tau-L+1>=9){
     sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
     }
     else {
     sum (t in 9..tau) x[<i,t>]+ sum (t in tau-L+1..8) x1[i][t]<= L * hL;
       }

     forall(i in E, tau in item(vac[i],5)+L..T)
      if (tau-L+1>=9){
     sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
     }
     else {
     sum (t in 9..tau) x[<i,t>]+ sum (t in tau-L+1..8) x1[i][t]<= L * hL;
       }
//Restricción 8,9: semanas fuertes
     forall(<i,t> in indices_a: t in Preplan)
     x[<i,t>]<=(hS+(hM-hS)*sit[<i,t>]);

     forall(i in E)
     (sum(t in Ai[i]: t in Preplan)(sit[<i,t>])
     + sum (t in Ai[i]: t in Pfijo)(sit1[i][t])
       <=S);

//Restricción 10,11: semanas flojas
      forall(<i,t> in indices_a: t in Preplan)
      x[<i,t>]<=(hM-(hM-hW)*wit[<i,t>]);

      forall(i in E)
      (sum(t in Ai[i]: t in Preplan) wit[<i,t>]
      + sum(t in Ai[i]: t in Pfijo) wit1[i][t]
```

```
        ) >=W;

//Restricción 15 : y(t,j,k) siempre positiva
     forall(<t,j,k> in indices_b: t in Preplan)
     y[<t,j,k>]>=0;

}
```

A.3.2 Modelo de la fase II (R2)

```
//Tipos para índices

tuple i_t {
  int i;
  int t;
};

tuple t_j_k {
  int t;
  int j;
  int k;
};

//Datos

int T= ...;
{int} Periodos = asSet (1..T);
{int} E= ...;
{int} J= ...;
{int} K= ...;
{int} Ej[J]= ...;
float rtk[Periodos][K]= ...;
{int} Ai[E]= ...;
{int} vac[i in E]= Periodos diff Ai[i];

float Hi[E]= ...;
float penalizacion[J][K]= ...;
float eficiencia[J][K]= ...;
{int} Ck[K]= ...;
{int} Fj[J]= ...;
float landa= ...;
float alfa= ...;
float betai[E]= ...;
float gammak[K]= ...;
float hm= ...;
float hM= ...;
```

```
int L= ...;
int hL= ...;
int S= ...;
int hS= ...;
int W= ...;
int hW= ...;

float sitdato[E][Periodos]= ...;
float witdato[E][Periodos]= ...;
float vdato[E]= ...;

float z1= ...;
float epsilon= ...;
float W1= ...;
float W2= ...;
float xmedia[E]= ...;
float dmedia[K]= ...;
int treplan= ...;
int Trigidez=...;
{int} Preplan = asSet (treplan+Trigidez..T);
{int} Pfijo = asSet (1..treplan+Trigidez-1);

float x1[E][Pfijo]= ...;
float x2[E][Preplan]= ...;
float sit1[E][Pfijo]= ...;
float sit2[E][Preplan]= ...;
float wit1[E][Pfijo]= ...;
float wit2[E][Preplan]= ...;
float HRi[i in E]= Hi[i] - (sum (t in Pfijo) x1[i][t]);
float rtknueva[Preplan][K]= ...;
float epsilon_replan= ...;
int nsmax= ...;
int ncmax= ...;
int ntmax= ...;
float tsi= ...;
int D= ...;
int hD= ...;
int Nf= ...;


float Zprima[K]= ...;
float gamma= ...;


//Variables

{i_t} indices_a = {<i,t> | i in E, t in Preplan: t in Ai[i]};
```

```
{t_j_k} indices_b = {<t,j,k> | t in Preplan, j in J, k in K:j in Ck[k]};

dvar float x[i in indices_a];
dvar float y[i in indices_b];
dvar float+ v[E];
dvar float+ d[Preplan][K];
dvar int sit[i in indices_a] in 0..1;
dvar int wit[i in indices_a] in 0..1;
dvar int cambio[i in indices_a] in 0..1;
dvar float xcpos[i in indices_a];
dvar float xcneg[i in indices_a];
dvar int xfuerte[i in indices_a] in 0..1;

//Función objetivo

minimize
   (sum (i in E) betai[i] * v[i])
   +
    (sum (k in K) gammak[k]* ( sum (t in Preplan) d[t][k]))
   +
       (landa * sum (t in Preplan) sum ( k in K) sum (j in Ck[k])
        penalizacion[j][k]* y[<t,j,k>] )

 ;

//Restricciones

subject to {

//Restricción 2: horas requeridas se satisfacen entre los empleados de la
empresa y los externos
      forall (t in Preplan)
        forall (k in K)
          sum(j in Ck[k]) (eficiencia[j][k]*y[<t,j,k>])
           + d[t][k] >= rtknueva[t][k];

//Restricción 3: cota inferior de horas semanales
  forall (i in E)
 forall (t in Preplan: t in Ai[i])
  x[<i,t>] >= hm;

//Restricción 3bis: cota superior de horas semanales
 forall (i in E)
 forall (t in Preplan: t in Ai[i])
   x[<i,t>] <= hM;

//Restricción 4: cota del número de horas extras
```

```
  forall (i in E)
  v[i] <= alfa * Hi[i] ;

//Restricción 5 (horas trabajadas=horas extra+horas estipuladas)
    forall (i in E)
      sum(t in Preplan: t in Ai[i]) x[<i,t>] == HRi[i]+v[i];

//Restricción 6
     forall(t in Preplan, j in J)
        sum(k in Fj[j]) y[<t,j,k>] ==
        sum(i in Ej[j]:t in Ai[i])  x[<i,t>] ;

//Restricción 7


    forall(i in E, tau in L..item(vac[i],0)-1)
    if (tau-L+1>=9){
    sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
    }
    else {
    sum (t in 9..tau) x[<i,t>] + sum (t in tau-L+1..8) x1[i][t]<= L * hL;
      }

    forall(i in E, tau in item(vac[i],1)+L..item(vac[i],2)-1)
    if (tau-L+1>=9){
    sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
    }
    else {
    sum (t in 9..tau) x[<i,t>]+ sum (t in tau-L+1..8) x1[i][t]<= L * hL;
      }



    forall(i in E, tau in item(vac[i],5)+L..T)
     if (tau-L+1>=9){
    sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
    }
    else {
    sum (t in 9..tau) x[<i,t>]+ sum (t in tau-L+1..8) x1[i][t]<= L * hL;
      }
//Restricción 8,9: semanas fuertes
    forall(<i,t> in indices_a: t in Preplan)
    x[<i,t>]<=(hS+(hM-hS)*sit[<i,t>]);

    forall(i in E)
    (sum(t in Ai[i]: t in Preplan)(sit[<i,t>])
    + sum (t in Ai[i]: t in Pfijo)(sit1[i][t])
```

```
        <=S);

//Restricción 10,11: semanas flojas
        forall(<i,t> in indices_a: t in Preplan)
        x[<i,t>]<=(hM-(hM-hW)*wit[<i,t>]);

        forall(i in E)
        (sum(t in Ai[i]: t in Preplan) wit[<i,t>]
        + sum(t in Ai[i]: t in Pfijo) wit1[i][t]
        ) >=W;

//Restricción 15 : y(t,j,k) siempre positiva
     forall(<t,j,k> in indices_b: t in Preplan)
     y[<t,j,k>]>=0;


//Definición de xcpos y xcneg:

forall(i in E, t in Preplan: t in Ai[i] )
xcpos[<i,t>] >= x[<i,t>]-x2[i][t];

forall(i in E, t in Preplan: t in Ai[i] )
xcneg[<i,t>] >= x2[i][t] - x[<i,t>];

forall(i in E, t in Preplan: t in Ai[i] )
xcpos[<i,t>] >= 0;

forall(i in E, t in Preplan: t in Ai[i] )
xcneg[<i,t>] >= 0;

//Definición de cambio:

forall(i in E, t in Preplan: t in Ai[i] ){
x[<i,t>]<= x2[i][t] + epsilon_replan + (hM-x2[i][t]-epsilon_replan) *
cambio[<i,t>];
x[<i,t>] >= x2[i][t] -epsilon_replan + (hm-x2[i][t]+epsilon_replan) *
cambio[<i,t>];
}

//R1:

forall (i in E)
  sum(t in Preplan: t in Ai[i]) cambio[<i,t>] <= nsmax;

//R2: Número máximo de horas por trabajador y semana a modificar

forall(i in E, t in Preplan: t in Ai[i] )
```

```
xcpos[<i,t>] + xcneg[<i,t>] <= ncmax;

//R6: Límite de semanas que pasan a ser fuertes

forall ( i in E)
  sum(t in Preplan: t in Ai[i])
  xfuerte[<i,t>] <= Nf;

  forall (i in E, t in Preplan: t in Ai[i])
  xfuerte[<i,t>] >= sit[<i,t>]-sit2[i][t];

}
```

A.3.3 Modelo de la fase III (R3)

```
//Tipos para índices

tuple i_t {
  int i;
  int t;
};

tuple t_j_k {
  int t;
  int j;
  int k;
};

//Datos

int T= ...;
{int} Periodos = asSet (1..T);
{int} E= ...;
{int} J= ...;
{int} K= ...;
{int} Ej[J]= ...;
float rtk[Periodos][K]= ...;
{int} Ai[E]= ...;
{int} vac[i in E]= Periodos diff Ai[i];

float Hi[E]= ...;
float penalizacion[J][K]= ...;
float eficiencia[J][K]= ...;
{int} Ck[K]= ...;
{int} Fj[J]= ...;
float landa= ...;
```

```
float alfa= ...;
float betai[E]= ...;
float gammak[K]= ...;
float hm= ...;
float hM= ...;
int L= ...;
int hL= ...;
int S= ...;
int hS= ...;
int W= ...;
int hW= ...;

float sitdato[E][Periodos]= ...;
float witdato[E][Periodos]= ...;
float vdato[E]= ...;

float z1= ...;
float epsilon= ...;
float W1= ...;
float W2= ...;
float xmedia[E]= ...;
float dmedia[K]= ...;
int treplan= ...;
int Trigidez=...;
{int} Preplan = asSet (treplan+Trigidez..T);
{int} Pfijo = asSet (1..treplan+Trigidez-1);

float x1[E][Pfijo]= ...;
float x2[E][Preplan]= ...;
float sit1[E][Pfijo]= ...;
float sit2[E][Preplan]= ...;
float wit1[E][Pfijo]= ...;
float wit2[E][Preplan]= ...;
float HRi[i in E]= Hi[i] - (sum (t in Pfijo) x1[i][t]);
float rtknueva[Preplan][K]= ...;
float epsilon_replan= ...;
int nsmax= ...;
int ncmax= ...;
int ntmax= ...;
float tsi= ...;
int D= ...;
int hD= ...;
int Nf= ...;


float Zprima[K]= ...;
float gamma= ...;
```

```
//Variables

{i_t} indices_a = {<i,t> | i in E, t in Preplan: t in Ai[i]};
{t_j_k} indices_b = {<t,j,k> | t in Preplan, j in J, k in K:j in Ck[k]};

dvar float x[i in indices_a];
dvar float y[i in indices_b];
dvar float+ v[E];
dvar float+ d[Preplan][K];
dvar int sit[i in indices_a] in 0..1;
dvar int wit[i in indices_a] in 0..1;
dvar int cambio[i in indices_a] in 0..1;
dvar float xcpos[i in indices_a];
dvar float xcneg[i in indices_a];
dvar int xfuerte[i in indices_a] in 0..1;

//Función objetivo

   minimize
 sum (i in E, t in Preplan: t in Ai[i])  ( tsi * xcpos[<i,t>]+  xcneg[<i,t>])  ;


//Restricciones

subject to {

//Restricción 2: horas requeridas se satisfacen entre los empleados de la
empresa y los externos
      forall (t in Preplan)
        forall (k in K)
          sum(j in Ck[k]) (eficiencia[j][k]*y[<t,j,k>])
           + d[t][k] >= rtknueva[t][k];

//Restricción 3: cota inferior de horas semanales
  forall (i in E)
 forall (t in Preplan: t in Ai[i])
  x[<i,t>] >= hm;

//Restricción 3bis: cota superior de horas semanales
 forall (i in E)
 forall (t in Preplan: t in Ai[i])
   x[<i,t>] <= hM;

//Restricción 4: cota del número de horas extras
  forall (i in E)
  v[i] <= alfa * Hi[i] ;
```

```
//Restricción 5 (horas trabajadas=horas extra+horas estipuladas)
      forall (i in E)
         sum(t in Preplan: t in Ai[i]) x[<i,t>] == HRi[i]+v[i];

//Restricción 6
       forall(t in Preplan, j in J)
            sum(k in Fj[j]) y[<t,j,k>] ==
            sum(i in Ej[j]:t in Ai[i])  x[<i,t>] ;

//Restricción 7


      forall(i in E, tau in L..item(vac[i],0)-1)
      if (tau-L+1>=9){
      sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
      }
      else {
      sum (t in 9..tau) x[<i,t>] + sum (t in tau-L+1..8) x1[i][t]<= L * hL;
        }

      forall(i in E, tau in item(vac[i],1)+L..item(vac[i],2)-1)
      if (tau-L+1>=9){
      sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
      }
      else {
      sum (t in 9..tau) x[<i,t>]+ sum (t in tau-L+1..8) x1[i][t]<= L * hL;
        }



      forall(i in E, tau in item(vac[i],5)+L..T)
       if (tau-L+1>=9){
      sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
      }
      else {
      sum (t in 9..tau) x[<i,t>]+ sum (t in tau-L+1..8) x1[i][t]<= L * hL;
        }

//Restricción 8,9: semanas fuertes
      forall(<i,t> in indices_a: t in Preplan)
      x[<i,t>]<=(hS+(hM-hS)*sit[<i,t>]);

      forall(i in E)
      (sum(t in Ai[i]: t in Preplan)(sit[<i,t>])
      + sum (t in Ai[i]: t in Pfijo)(sit1[i][t])
        <=S);
```

```
//Restricción 10,11: semanas flojas
      forall(<i,t> in indices_a: t in Preplan)
      x[<i,t>]<=(hM-(hM-hW)*wit[<i,t>]);

      forall(i in E)
      (sum(t in Ai[i]: t in Preplan) wit[<i,t>]
      + sum(t in Ai[i]: t in Pfijo) wit1[i][t]
      ) >=W;

//Restricción 15 : y(t,j,k) siempre positiva
    forall(<t,j,k> in indices_b: t in Preplan)
    y[<t,j,k>]>=0;


//Definición de xcpos y xcneg:

forall(i in E, t in Preplan: t in Ai[i] )
xcpos[<i,t>] >= x[<i,t>]-x2[i][t];

forall(i in E, t in Preplan: t in Ai[i] )
xcneg[<i,t>] >= x2[i][t] - x[<i,t>];

forall(i in E, t in Preplan: t in Ai[i] )
xcpos[<i,t>] >= 0;

forall(i in E, t in Preplan: t in Ai[i] )
xcneg[<i,t>] >= 0;

//Definición de cambio:


forall(i in E, t in Preplan: t in Ai[i] ){
x[<i,t>]<= x2[i][t] + epsilon_replan + (hM-x2[i][t]-epsilon_replan) *
cambio[<i,t>];
x[<i,t>] >= x2[i][t] -epsilon_replan + (hm-x2[i][t]+epsilon_replan) *
cambio[<i,t>];

}

//R1:


forall (i in E)
  sum(t in Preplan: t in Ai[i]) cambio[<i,t>] <= nsmax;


//R2: Número máximo de horas por trabajador y semana a modificar
```

```
forall(i in E, t in Preplan: t in Ai[i] )
xcpos[<i,t>] + xcneg[<i,t>] <= ncmax;

//R6: Límite de semanas que pasan a ser fuertes

forall ( i in E)
  sum(t in Preplan: t in Ai[i])
  xfuerte[<i,t>] <= Nf;

  forall (i in E, t in Preplan: t in Ai[i])
  xfuerte[<i,t>] >= sit[<i,t>]-sit2[i][t];

//Restricción de la etapa 3

sum (i in E) betai[i] * v[i]
+ sum (k in K, t in Preplan ) ( gammak[k] * d[t][k])
+ landa * sum (t in Preplan, k in K, j in Ck[k]) penalizacion[j][k] *
y[<t,j,k>]
<= 0.001 + Zprima[1];


}
```

A.3.4 Modelo de la fase IV (R4)

```
//Tipos para índices

tuple i_t {
  int i;
  int t;
};

tuple t_j_k {
  int t;
  int j;
  int k;
};

//Datos

int T= ...;
{int} Periodos = asSet (1..T);
{int} E= ...;
{int} J= ...;
{int} K= ...;
```

```
{int} Ej[J]= ...;
float rtk[Periodos][K]= ...;
{int} Ai[E]= ...;
{int} vac[i in E]= Periodos diff Ai[i];

float Hi[E]= ...;
float penalizacion[J][K]= ...;
float eficiencia[J][K]= ...;
{int} Ck[K]= ...;
{int} Fj[J]= ...;
float landa= ...;
float alfa= ...;
float betai[E]= ...;
float gammak[K]= ...;
float hm= ...;
float hM= ...;
int L= ...;
int hL= ...;
int S= ...;
int hS= ...;
int W= ...;
int hW= ...;

float sitdato[E][Periodos]= ...;
float witdato[E][Periodos]= ...;
float vdato[E]= ...;

float z1= ...;
float epsilon= ...;
float W1= ...;
float W2= ...;
float xmedia[E]= ...;
float dmedia[K]= ...;
int treplan= ...;
int Trigidez=...;
{int} Preplan = asSet (treplan+Trigidez..T);
{int} Pfijo = asSet (1..treplan+Trigidez-1);

float x1[E][Pfijo]= ...;
float x2[E][Preplan]= ...;
float sit1[E][Pfijo]= ...;
float sit2[E][Preplan]= ...;
float wit1[E][Pfijo]= ...;
float wit2[E][Preplan]= ...;
float HRi[i in E]= Hi[i] - (sum (t in Pfijo) x1[i][t]);
float rtknueva[Preplan][K]= ...;
float epsilon_replan= ...;
```

```
int nsmax= ...;
int ncmax= ...;
int ntmax= ...;
float tsi= ...;
int D= ...;
int hD= ...;
int Nf= ...;


float Zprima[K]= ...;
float gamma= ...;


//Variables

{i_t} indices_a = {<i,t> | i in E, t in Preplan: t in Ai[i]};
{t_j_k} indices_b = {<t,j,k> | t in Preplan, j in J, k in K:j in Ck[k]};

dvar float x[i in indices_a];
dvar float y[i in indices_b];
dvar float+ v[E];
dvar float+ d[Preplan][K];
dvar int sit[i in indices_a] in 0..1;
dvar int wit[i in indices_a] in 0..1;
dvar int cambio[i in indices_a] in 0..1;
dvar float xcpos[i in indices_a];
dvar float xcneg[i in indices_a];
dvar int xfuerte[i in indices_a] in 0..1;

//Función objetivo

   minimize
 sum (i in E, t in Preplan: t in Ai[i])  ( tsi * xcpos[<i,t>]+  xcneg[<i,t>])  ;


//Restricciones

subject to {

//Restricción 2: horas requeridas se satisfacen entre los empleados de la
empresa y los externos
      forall (t in Preplan)
        forall (k in K)
          sum(j in Ck[k]) (eficiencia[j][k]*y[<t,j,k>])
           + d[t][k] >= rtknueva[t][k];

//Restricción 3: cota inferior de horas semanales
```

```
  forall (i in E)
 forall (t in Preplan: t in Ai[i])
  x[<i,t>] >= hm;

//Restricción 3bis: cota superior de horas semanales
 forall (i in E)
 forall (t in Preplan: t in Ai[i])
   x[<i,t>] <= hM;

//Restricción 4: cota del número de horas extras
  forall (i in E)
  v[i] <= alfa * Hi[i] ;

//Restricción 5 (horas trabajadas=horas extra+horas estipuladas)
     forall (i in E)
        sum(t in Preplan: t in Ai[i]) x[<i,t>] == HRi[i]+v[i];

//Restricción 6
      forall(t in Preplan, j in J)
           sum(k in Fj[j]) y[<t,j,k>] ==
           sum(i in Ej[j]:t in Ai[i])  x[<i,t>] ;

//Restricción 7


     forall(i in E, tau in L..item(vac[i],0)-1)
     if (tau-L+1>=9){
     sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
     }
     else {
     sum (t in 9..tau) x[<i,t>] + sum (t in tau-L+1..8) x1[i][t]<= L * hL;
       }

     forall(i in E, tau in item(vac[i],1)+L..item(vac[i],2)-1)
     if (tau-L+1>=9){
     sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
     }
     else {
     sum (t in 9..tau) x[<i,t>]+ sum (t in tau-L+1..8) x1[i][t]<= L * hL;
       }

     forall(i in E, tau in item(vac[i],5)+L..T)
      if (tau-L+1>=9){
     sum (t in tau-L+1..tau) x[<i,t>]<= L * hL;
     }
     else {
     sum (t in 9..tau) x[<i,t>]+ sum (t in tau-L+1..8) x1[i][t]<= L * hL;
```

```
        }
//Restricción 8,9: semanas fuertes
      forall(<i,t> in indices_a: t in Preplan)
      x[<i,t>]<=(hS+(hM-hS)*sit[<i,t>]);

      forall(i in E)
      (sum(t in Ai[i]: t in Preplan)(sit[<i,t>])
      + sum (t in Ai[i]: t in Pfijo)(sit1[i][t])
       <=S);

//Restricción 10,11: semanas flojas
       forall(<i,t> in indices_a: t in Preplan)
       x[<i,t>]<=(hM-(hM-hW)*wit[<i,t>]);

       forall(i in E)
       (sum(t in Ai[i]: t in Preplan) wit[<i,t>]
       + sum(t in Ai[i]: t in Pfijo) wit1[i][t]
       ) >=W;

//Restricción 15 : y(t,j,k) siempre positiva
     forall(<t,j,k> in indices_b: t in Preplan)
     y[<t,j,k>]>=0;


//Restricciones que limitan los cambios

//Definición de xcpos y xcneg:

forall(i in E, t in Preplan: t in Ai[i] )
xcpos[<i,t>] >= x[<i,t>]-x2[i][t];

forall(i in E, t in Preplan: t in Ai[i] )
xcneg[<i,t>] >= x2[i][t] - x[<i,t>];

forall(i in E, t in Preplan: t in Ai[i] )
xcpos[<i,t>] >= 0;

forall(i in E, t in Preplan: t in Ai[i] )
xcneg[<i,t>] >= 0;

//Definición de cambio:

forall(i in E, t in Preplan: t in Ai[i] ){
x[<i,t>]<= x2[i][t] + epsilon_replan + (hM-x2[i][t]-epsilon_replan) *
cambio[<i,t>];
x[<i,t>] >= x2[i][t] -epsilon_replan + (hm-x2[i][t]+epsilon_replan) *
cambio[<i,t>];
```

```
}

//R1:

forall (i in E)
  sum(t in Preplan: t in Ai[i]) cambio[<i,t>] <= nsmax;


//R2: Número máximo de horas por trabajador y semana a modificar

forall(i in E, t in Preplan: t in Ai[i] )
xcpos[<i,t>] + xcneg[<i,t>] <= ncmax;


//R6: Límite de semanas que pasan a ser fuertes

forall ( i in E)
  sum(t in Preplan: t in Ai[i])
  xfuerte[<i,t>] <= Nf;

  forall (i in E, t in Preplan: t in Ai[i])
  xfuerte[<i,t>] >= sit[<i,t>]-sit2[i][t];

//--------Restricción de la etapa 4----------


sum (i in E) betai[i] * v[i]
+ sum (k in K, t in Preplan ) ( gammak[k] * d[t][k])
+ landa * sum (t in Preplan, k in K, j in Ck[k]) penalizacion[j][k] *
y[<t,j,k>]
<= gamma * Zprima[1];

}
```

A.4 Programa de unión de los modelos

A continuación se presenta el programa que permite, también en lenguaje OPL, la ejecución consecutiva de los modelos de planificación y replanificación. Este programa está diseñado de tal manera que sólo introduciendo la lista de ejemplares a resolver, para cada uno de ellos se sucedan los dos modelos de planificación y las cuatro fases de replanificación. La lista de ejemplares se ejecuta en el orden dado, y los resultados

deseados de la resolución de los ejemplares se almacenan en un único fichero de resultados.

```
main {


var basis = new IloOplCplexBasis();
var source_M1 = new IloOplModelSource(MODELO_1);
var def_M1 = new IloOplModelDefinition(source_M1);
var cplex_M1 = new IloCplex();
var source_M2 = new IloOplModelSource(MODELO_2);
var def_M2 = new IloOplModelDefinition(source_M2);
var cplex_M2 = new IloCplex();
var source_R1 = new IloOplModelSource(MODELO_R1);
var def_R1 = new IloOplModelDefinition(source_R1);
var cplex_R1 = new IloCplex();
var source_R2 = new IloOplModelSource(MODELO_R2);
var def_R2 = new IloOplModelDefinition(source_R2);
var cplex_R2 = new IloCplex();
var source_R3 = new IloOplModelSource(MODELO_R3);
var def_R3 = new IloOplModelDefinition(source_R3);
var cplex_R3 = new IloCplex();
var source_R4 = new IloOplModelSource(MODELO_R4);
var def_R4 = new IloOplModelDefinition(source_R4);
var cplex_R4 = new IloCplex();

//Inicializaciones

cplex_M1.tiLim = 3600; cplex_M2.tiLim = 3600;
cplex_R1.tiLim = 3600; cplex_R2.tiLim = 3600; cplex_R3.tiLim = 3600;
cplex_R4.tiLim = 3600;

cplex_M1.epGap = 0.01; cplex_M2.epGap = 0.01;
cplex_R1.epGap = 0.01; cplex_R2.epGap = 0.01;cplex_R3.epGap = 0.01;
cplex_R4.epGap = 0.01;

cplex_M1.nodeFileInd = 3; cplex_M2.nodeFileInd = 3;
cplex_R1.nodeFileInd = 3; cplex_R2.nodeFileInd =
3;cplex_R3.nodeFileInd = 3; cplex_R4.nodeFileInd = 3;

cplex_M1.treLim = 128; cplex_M2.treLim = 128;
cplex_R1.treLim = 128; cplex_R2.treLim = 128;cplex_R3.treLim = 128;
cplex_R4.treLim = 128;

cplex_M1.workMem = 30000; cplex_M2.workMem = 30000;
```

```
cplex_R1.workMem = 30000; cplex_R2.workMem = 30000;
cplex_R3.workMem = 30000; cplex_R4.workMem = 30000;

//Fichero de resultados

var ofile = new
IloOplOutputFile("C:\\MONTSE\\Resultados\\resultados_R1R2R6_O3.txt")
;


//Crear primer modelo y resolver para cada ejemplar

for (var i=0; i < data_names.length; i++)

{
  var tiempoM1;
  var time_o;
  var time_m2;
  var time_r1;
  var time_r2;
  var time_r3;
  var time_r4;

  var opl_M1 = new IloOplModel(def_M1, cplex_M1);

  var data = new IloOplDataSource("C:\\MONTSE\\Ejemplares_montse\\"
+ data_names[i]);

  opl_M1.addDataSource(data);
  opl_M1.generate();
  basis.getBasis(cplex_M1);

//------Obtener el tiempo que ha tardado en resolver M1(en segundos, si
trabajáramos con UNIx el tiempo sería en CPU)------

time_o = ((new Date()).getTime()) / 1000.0;
if ( cplex_M1.solve() ) {
  tiempoM1 = (((new Date()).getTime()) / 1000.0) - time_o;
  writeln("Tiempo ejecución Modelo 1: " + tiempoM1);

  ofile.writeln("ejemplar:"+ i + ";" + tiempoM1 + ";" );
  ofile.writeln( cplex_M1.getObjValue() + ";" );
    }
else {
  writeln("Modelo 1 no resuelto para " + data_names[i]);
  ofile.writeln("Modelo 1 no resuelto para " + data_names[i]);
    }
```

```
//Calcular los valores medios

for (tr in opl_M1.E) {
  var horas_total = 0;
      for (t in opl_M1.Ai[tr]) {
        horas_total += opl_M1.x[opl_M1.indices_a.find(tr, t)];
        }
        var xmedia = (horas_total + opl_M1.v[tr]) / 46;
        //     ofile.writeln ("Promedio de horas: " + xmedia);
   }

for(k in opl_M1.K){
  var horas_sub = 0;
      for (t in opl_M1.Periodos) {
        horas_sub += opl_M1.d[t][k];
        }
        var dmedia = horas_sub / 46;
       // ofile.writeln ("Promedio de externos:" + dmedia);
  }


//Volcar resultados de M1 en datos de M2

    var opl_M2 = new IloOplModel(def_M2, cplex_M2);
    var data2= opl_M1.dataElements;
    opl_M2.addDataSource(data2);
    opl_M2.generate();
    data2 = opl_M2.dataElements;

data2.z1 = cplex_M1.getObjValue();

for (tr in opl_M1.E){
opl_M2.vdato[tr] = opl_M1.v[tr];
opl_M2.xmedia[tr]= xmedia;
}

for (kk in opl_M1.K){
opl_M2.dmedia[kk] = dmedia;
}

for(tr in opl_M1.E){for(t in opl_M1.Ai[tr]){
opl_M2.witdato[tr][t] = opl_M1.wit[opl_M1.indices_a.find(tr, t)];
}}

for(tr in opl_M1.E){for(t in opl_M1.Ai[tr]){
opl_M2.sitdato[tr][t] = opl_M1.sit[opl_M1.indices_a.find(tr, t)];
```

```
}}

//Crear M2 y resolver

opl_M2 = new IloOplModel(def_M2, cplex_M2);
opl_M2.addDataSource(data2);
opl_M2.generate();
basis.getBasis(cplex_M2);

//Determinar tiempo que tarda M2

time_o = ((new Date()).getTime()) / 1000.0;
if ( !cplex_M2.solve() ) {
  writeln("Modelo 2 no resuelto para " + data_names2[i]);
  ofile.writeln("Modelo 2 no resuelto para " + data_names2[i]);
    }
else {
  time_m2 = (((new Date()).getTime()) / 1000.0) - time_o;
  writeln ("Tiempo ejecución Modelo 2: " + time_m2);
  ofile.writeln (time_m2+ ";");
  ofile.writeln( cplex_M2.getObjValue() + ";");

}

//Replanificación Fase 1

  var opl_R1 = new IloOplModel(def_R1, cplex_R1);
  var datar1= opl_M2.dataElements;
  opl_R1.addDataSource(datar1);
  opl_R1.generate();
  datar1 = opl_R1.dataElements;

for (tr in opl_M2.E){
  for (t in opl_M2.Ai[tr]){
  if(t < (opl_R1.treplan + opl_R1.Trigidez)){
    opl_R1.x1[tr][t] = opl_M2.x[opl_M2.indices_a.find(tr, t)];}
  else{}
}}


for (tr in opl_M2.E){
  for (t in opl_M2.Ai[tr]){
  if(t >= (opl_R1.treplan + opl_R1.Trigidez)){
    opl_R1.x2[tr][t] = opl_M2.x[opl_M2.indices_a.find(tr, t)];}
  else{}
}}
```

```
for (tr in opl_M2.E){
  for (t in opl_M2.Ai[tr]){
  if(t < (opl_R1.treplan + opl_R1.Trigidez)){
    opl_R1.wit1[tr][t] = opl_M2.witdato[tr][t];}
  else{}
}}

for (tr in opl_M2.E){
  for (t in opl_M2.Ai[tr]){
  if(t >= (opl_R1.treplan + opl_R1.Trigidez)){
    opl_R1.wit2[tr][t] = opl_M2.witdato[tr][t];}
  else{}
}}

for (tr in opl_M2.E){
  for (t in opl_M2.Ai[tr]){
  if(t < (opl_R1.treplan + opl_R1.Trigidez)){
    opl_R1.sit1[tr][t] = opl_M2.sitdato[tr][t];}
  else{}
}}

for (tr in opl_M2.E){
  for (t in opl_M2.Ai[tr]){
  if(t >= (opl_R1.treplan + opl_R1.Trigidez)){
    opl_R1.sit2[tr][t] = opl_M2.sitdato[tr][t];}
  else{}
}}

//Crear r1 y resolver

opl_R1 = new IloOplModel(def_R1, cplex_R1);
opl_R1.addDataSource(datar1);
opl_R1.generate();
basis.getBasis(cplex_R1);

//Determinar tiempo que tarda r1

time_0 = ((new Date()).getTime()) / 1000.0;
if ( !cplex_R1.solve() ) {
  writeln("Replanificación-etapa1 no resuelto para " + data_names[i] );
  ofile.writeln("Replanificación-etapa1 no resuelto para " +
data_names[i]);
    }
else {
  time_r1 = (((new Date()).getTime()) / 1000.0) - time_0;
  writeln ("Tiempo ejecución Replanificación-1: " + time_r1);
```

```
  ofile.writeln ( time_r1 + ";");
  ofile.writeln(cplex_R1.getObjValue()+ ";");
  }

//Crear r2

  var opl_R2 = new IloOplModel(def_R2, cplex_R2);
  var datar2 = opl_R1.dataElements;
  opl_R2.addDataSource(datar2);
  opl_R2.generate();
  basis.getBasis(cplex_R2);

//Determinar tiempo que tarda r2

time_0 = ((new Date()).getTime()) / 1000.0;
if ( !cplex_R2.solve() ) {
  writeln("Replanificación-etapa2 no resuelto para " + data_names[i]);
  ofile.writeln("Replanificación-etapa2 no resuelto para " +
data_names[i]) ;
    }
else {
  time_r2 = (((new Date()).getTime()) / 1000.0) - time_0;
  writeln ("Tiempo ejecución Replanificación-2: " + time_r2);
  ofile.writeln (time_r2 + ";");
  ofile.writeln(cplex_R2.getObjValue() + ";");
}

//Crear r3

  var opl_R3 = new IloOplModel(def_R3, cplex_R3);
  var datar3 = opl_R1.dataElements;

  opl_R3.addDataSource(datar3);
  opl_R3.generate();
  basis.getBasis(cplex_R3);

  opl_R3.Zprima[1] = cplex_R2.getObjValue();

//Determinar tiempo que tarda r3

time_0 = ((new Date()).getTime()) / 1000.0;
if ( !cplex_R3.solve() ) {
  writeln("Replanificación-etapa3 no resuelto para " + data_names[i]);
  ofile.writeln("Replanificación-etapa3 no resuelto para " +
data_names[i]);
    }
```

```
else {
  time_r3 = (((new Date()).getTime()) / 1000.0) - time_0;
  writeln ("Tiempo ejecución Replanificación-3: " + time_r3);
  ofile.writeln (time_r3 + ";" );
  ofile.writeln(cplex_R3.getObjValue() + ";");

}

//Crear r4

  var opl_R4 = new IloOplModel(def_R4, cplex_R4);
  var datar4 = opl_R1.dataElements;
  opl_R4.addDataSource(datar4);
  opl_R4.generate();

  opl_R4.Zprima[1] = cplex_R2.getObjValue();

//Determinar tiempo que tarda r4

time_0 = ((new Date()).getTime()) / 1000.0;
if ( !cplex_R4.solve() ) {
  writeln("Replanificación-etapa4 no resuelto para " + data_names[i]);
  ofile.writeln("Replanificación-etapa4 no resuelto para " +
data_names[i]);
    }
else {
  time_r4 = (((new Date()).getTime()) / 1000.0) - time_0;
  writeln ("Tiempo ejecución Replanificación-4: " + time_r4);
  ofile.writeln ( time_r4 + ";");
  ofile.writeln( cplex_R4.getObjValue() + ";");
}


//Liberar memoria

opl_M1.endAll();
opl_M2.endAll();
opl_R1.endAll();
opl_R2.endAll();
opl_R3.endAll();
opl_R4.endAll();
data.end();
data2.end();
datar1.end();
datar2.end();
datar3.end();
datar4.end();
```

```
}

//Cerrar fichero de resultados

ofile.close();
def_M1.end();source_M1.end();cplex_M1.end();
def_M2.end();source_M2.end();cplex_M2.end();
def_R1.end();source_R1.end();cplex_R1.end();
def_R2.end();source_R2.end();cplex_R2.end();
def_R3.end();source_R3.end();cplex_R3.end();
def_R4.end();source_R4.end();cplex_R4.end();
}
```

B Programa para la generación de los ejemplares

A continuación se encuentra el programa informático utilizado para la generación de los 675 ejemplares utilizados en la experimentación. El lenguaje de programación utilizado es Java.

```
//LIBRERÍAS

import java.io.*;
import java.util.*;
import tuccojava.utiles.mates.*;
import tuccojava.utiles.estructuras.*;
import tuccojava.estructuras.*;

public class genera_ejemplares_plan {

private static class trabajador {
      public int id;
      public int categoria;

      public trabajador (int id, int categoria) {
         this.id = id;
         this.categoria = categoria;
      }
}

//DEFINICIÓN DE LOS DATOS

private static class data {

      public static final int T = 52;
      public static final int Hi =1760;

      public static final int TABLA1_EFIC = 1;
      public static final int TABLA2_EFIC = 2;
      public static final int TABLA3_EFIC = 3;
      private static final double[][] tabla1_eficiencia = {
         {1, 0.9, 0},
         {0, 1, 0.9},
         {0, 0, 1}
                                                  };
      private static final double[][] tabla2_eficiencia = {
         {1, 0.9, 0.8},
```

```
    {0, 1, 0.9},
    {0, 0, 1}};
private static final double[][] tabla3_eficiencia = {
    {1, 0, 0},
    {0.9, 1, 0},
    {0.8, 0, 1}};
public static final int TABLA1_PENAL = 1;
public static final int TABLA2_PENAL = 2;
public static final int TABLA3_PENAL = 3;
private static final double[][] tabla1_penalizacion = {
    {0, 5, Double.MAX_VALUE},
    {Double.MAX_VALUE, 0, 2},
    {Double.MAX_VALUE, Double.MAX_VALUE, 0}
                                        };
private static final double[][] tabla2_penalizacion = {
    {0, 5, 10},
    {Double.MAX_VALUE, 0, 2},
    {Double.MAX_VALUE, Double.MAX_VALUE, 0}
                                        };
private static final double[][] tabla3_penalizacion = {
    {0, Double.MAX_VALUE, Double.MAX_VALUE},
    {2, 0, Double.MAX_VALUE},
    {5, Double.MAX_VALUE, 0}};


public static final int DEMANDA_TIPO1 = 1;
public static final int DEMANDA_TIPO2 = 2;
public static final int DEMANDA_TIPO3 = 3;
public double[][] eficiencia;
public double[][] penalizacion;
public int demanda;
public static final double ruido = 0.05;
private static final double PORCENTAJE_PICO = 0.25;
public static final int hm =20;
public static final int hM =50;
public static final int L =12;
public static final int hL =45;
public static final int S =10;
public static final int hS =48;
public static final int W =10;
public static final int hW =28;
public static final double alfa =0.05;
public static final double landa =0.001;
public static final double betai =1;
public static final double gammak =1.5;
```

```
public trabajador E[];
public int K[];   //[1..K]  tareas
public int J[];   //[1..J]  categorías de trabajadores
private static final double[] porcentaje_J = {0.5, 0.3, 0.2};
public static final double W1=1;
public static final double W2=1;
public static final double epsilon=5;
public static final double z1=0;
public double[] vdato;
public double[][] sitdato;
public double[][] witdato;
public double[] xmedia;
public double[] dmedia;
public static final int treplan = 5;
public static final int Trigidez = 4;
public static final double epsilon_replan = 0.5;
public static final double nsmax = 18;
public static final double ncmax = 15;
public static final double ntmax = 5;
public static final double tsi = 0.7;
public static final double D = 4;
public static final double hD=8;
public static final double Nf=6;
public static final double gamma = 1.05;
public double [][] rtknueva;
public double[] Zprima;
public double [][] x1;
public double [][] x2;
public double [][] wit1;
public double [][] wit2;
public double [][] sit1;
public double [][] sit2;


//Asignación de datos

public data (int E, int tabla_efic, int tabla_penal, int demanda) {
   this.J = new int[this.porcentaje_J.length];
   for (int i=0; i< this.J.length; i++)
      this.J[i] = i+1;

   int aux[] = new int[E];
   int contador = 0;

   for (int i=0; i<J.length; i++) {
      int limite;
      if (i < (J.length -1))
```

```
            limite = (int)(E * porcentaje_J[i]);
        else
            limite = E-contador;

                for (int j=0; j<limite; j++) {
                    aux[contador] = J[i];
                    contador++;
                }
        }

    this.E = new trabajador[E];

    for (int i=0; i< E; i++)
        this.E[i] = new trabajador(i+1, aux[i]);

    switch( tabla_efic ) {
        case TABLA1_EFIC:
            this.eficiencia = tabla1_eficiencia;
            break;
        case TABLA2_EFIC:
            this.eficiencia = tabla2_eficiencia;
            break;
        case TABLA3_EFIC:
            this.eficiencia = tabla3_eficiencia;
            break;
    }

    this.K = new int[this.eficiencia[0].length];

    switch ( tabla_penal) {
        case TABLA1_PENAL:
            this.penalizacion = tabla1_penalizacion;
            break;
        case TABLA2_PENAL:
            this.penalizacion = tabla2_penalizacion;
            break;
        case TABLA3_PENAL:
            this.penalizacion = tabla3_penalizacion;
            break;
    }

    this.K = new int[this.penalizacion[0].length];

    for (int i=0; i< this.K.length; i++)
        this.K[i] = i+1;

    this.demanda = demanda; }}
```

```
//Programa principal

public static void main(String[] args) {

String PATH = "Ejemplares_segundo\\";
int num_repeticiones = 25;
int E[] = {25, 50, 100};
int tipo_eficiencia[] = {data.TABLA1_EFIC, data.TABLA2_EFIC,
data.TABLA3_EFIC};
int tipo_penalizacion[] = {data.TABLA1_PENAL,
data.TABLA2_PENAL, data.TABLA3_PENAL};
int tipo_demanda[] = {data.DEMANDA_TIPO1,
data.DEMANDA_TIPO2, data.DEMANDA_TIPO3};
   int contador = 1
        for (int iw=0; iw<E.length; iw++) {
 for (int ite=0; ite<tipo_eficiencia.length; ite++) {
        for (int itd=0; itd<tipo_demanda.length; itd++) {
           for (int ini=1; ini<=num_repeticiones; ini++) {
              try {
        data jp = new data (E[iw], tipo_eficiencia[ite],
tipo_penalizacion[ite],tipo_demanda[itd]);
        String nombre_fichero = "EF_PEN"+ tipo_eficiencia[ite]
+"_DEM" + tipo_demanda[itd] +  "_E" + E[iw]  + "_" + (ini)+ ".dat" ;
        int total =
E.length*tipo_eficiencia.length*tipo_demanda.length*num_repeticiones;
        FileOutputStream fsalida = new FileOutputStream(PATH +
nombre_fichero);
        System.out.print("Generando " + nombre_fichero + "...");
        generar(jp, new PrintStream(fsalida));
        System.out.println("OK (" + contador + "/" + total + ")");

        contador++;

        fsalida.close();
                }
                catch (Exception e) {
                   System.out.println(e);
                }
             }
          }
        }
     }
  }
```

```
   //Subprograma que escribe los datos

private static void generar (data jp, PrintStream salida) {
 salida.println("T = " + jp.T + ";");
salida.println("hm = " + jp.hm + ";");
salida.println("hM = " + jp.hM + ";");
salida.println("L = " + jp.L + ";");
salida.println("hL = " + jp.hL + ";");
salida.println("S = " + jp.S + ";");
salida.println("hS = " + jp.hS + ";");
salida.println("W = " + jp.W + ";");
salida.println("hW = " + jp.hW + ";");
salida.println("alfa = " + jp.alfa + ";");
salida.println("landa = " + jp.landa + ";");
escribirE(salida, "E", jp.E.length);
tuccoArrays.print(salida, "K = {", "};\n", " ", jp.K);
tuccoArrays.print(salida, "J = {", "};\n", " ", jp.J);
escribirEj(salida, jp.E, jp.J.length);
escribirDatoCompartido(salida, "gammak", jp.K.length, jp.gammak);
escribirDatoCompartido(salida, "betai", jp.E.length, jp.betai);
escribirDatoCompartido(salida, "Hi", jp.E.length, jp.Hi);
escribirEficiencia(salida, jp.eficiencia);
escribirPenalizacion(salida, jp.penalizacion);
escribirCk(salida, jp.eficiencia, jp.K.length);
escribirFj(salida, jp.eficiencia, jp.J.length);
escribirAi(salida, jp.E.length, jp.T);

            switch( jp.demanda ) {
              case data.DEMANDA_TIPO1:
               escribirDemanda1(salida, jp.Hi, jp.E.length, jp.K.length, jp.T,
jp.ruido);
              break;
              case data.DEMANDA_TIPO2:
               escribirDemanda2(salida, jp.Hi, jp.E.length,jp.K.length, jp.T,
jp.ruido);
              break;
              case data.DEMANDA_TIPO3:
              escribirDemanda3(salida, jp.Hi, jp.E.length, jp.K.length, jp.T,
jp.ruido);
              break;
            }

            salida.println("//Datos del modelo M2");
            salida.println("W1= " + jp.W1 + ";");
            salida.println("W2= " + jp.W2 + ";");
            salida.println("epsilon= " + jp.epsilon + ";");
            salida.println("z1= " + jp.z1 + ";");
```

```
salida.println("vdato=#[ " );
for(int i=1; i<(jp.E.length + 1);i++){
salida.print(i + ": 0,");
}
salida.println("]#; " );
salida.println("sitdato=#[ " );
for(int i=1; i<(jp.E.length + 1);i++){
salida.println(i + ":#[");
   for(int j=1; j<(jp.T+1); j++){
      salida.print(j + ":0,");
      }
salida.println("]#,");
}
salida.println("]#; ");
salida.println("witdato=#[ " );
 for(int i=1; i<(jp.E.length + 1);i++){
salida.println(i + ":#[");
   for(int j=1; j<(jp.T+1); j++){
      salida.print(j + ":0,");
      }
salida.println("]#,");
}
salida.println("]#; ");

salida.println("xmedia=#[");
for(int i=1; i<(jp.E.length + 1);i++){
salida.print(i + ": 0,");
}
salida.println("]#; ");

salida.println("dmedia=#[");
for(int i=1; i<(jp.K.length+1);i++){
salida.print(i + ": 0,");
}
salida.println("]#; ");
salida.println("treplan= " + jp.treplan + ";");
salida.println("Trigidez= " + jp.Trigidez + ";");
salida.println("epsilon_replan= " + jp.epsilon_replan+ ";");
salida.println("nsmax= " + jp.nsmax + ";");
salida.println("ncmax= " + jp.ncmax + ";");
salida.println("ntmax= " + jp.ntmax + ";");
salida.println("tsi= " + jp.tsi + ";");
salida.println("D= " + jp.D + ";");
salida.println("hD= " + jp.hD + ";");
salida.println("Nf= " + jp.Nf + ";");
salida.println("gamma= " + jp.gamma + ";");
```

```
salida.println("Zprima=#[");
for(int i=1; i<(jp.K.length+1);i++){
salida.print(i + ": 0,");
}
salida.println("]#; ");


salida.println("x1=#[ " );
for(int i=1; i<(jp.E.length + 1);i++){
salida.println(i + ":#[");
   for(int j=1; j<(jp.treplan+jp.Trigidez); j++){
      salida.print(j + ":0,");
      }
salida.println("]#,");
}
salida.println("]#; ");
salida.println("x2=#[ " );
for(int i=1; i<(jp.E.length + 1);i++){
salida.println(i + ":#[");
   for(int j=(jp.treplan + jp.Trigidez); j<(jp.T+1); j++){
      salida.print(j + ":0,");
      }
salida.println("]#,");
}
salida.println("]#; ");
salida.println("wit1=#[ " );
for(int i=1; i<(jp.E.length + 1);i++){
salida.println(i + ":#[");
   for(int j=1; j<(jp.treplan+jp.Trigidez); j++){
      salida.print(j + ":0,");
      }
salida.println("]#,");
}
salida.println("]#; ");
salida.println("wit2=#[ " );
for(int i=1; i<(jp.E.length + 1);i++){
salida.println(i + ":#[");
   for(int j=(jp.treplan + jp.Trigidez); j<(jp.T+1); j++){
      salida.print(j + ":0,");
      }
salida.println("]#,");
}
salida.println("]#; ");


salida.println("sit1=#[ " );
```

```
            for(int i=1; i<(jp.E.length + 1);i++){
            salida.println(i + ":#[");
               for(int j=1; j<(jp.treplan+jp.Trigidez); j++){
                  salida.print(j + ":0,");
                  }
            salida.println("]#,");
            }
            salida.println("]#; ");
            salida.println("sit2=#[ " );
            for(int i=1; i<(jp.E.length + 1);i++){
            salida.println(i + ":#[");
               for(int j=(jp.treplan + jp.Trigidez); j<(jp.T+1); j++){
                  salida.print(j + ":0,");
                  }
            salida.println("]#,");
            }
            salida.println("]#; ");
            switch( jp.demanda ) {
               case data.DEMANDA_TIPO1:
                escribirDemanda1nueva(salida, jp.Hi, jp.E.length, jp.K.length,
jp.T, jp.ruido);
               break;
               case data.DEMANDA_TIPO2:
                escribirDemanda2nueva(salida, jp.Hi, jp.E.length,jp.K.length,
jp.T, jp.ruido);
               break;
               case data.DEMANDA_TIPO3:
               escribirDemanda3nueva(salida, jp.Hi, jp.E.length, jp.K.length,
jp.T, jp.ruido);
               break;
            }


            }

   //Subprograma que escribe datos compartidos

      private static void escribirDatoCompartido(PrintStream salida, String
nombre, int N, double valor) {
      salida.print(nombre + " = #[");
      for (int i=1; i<=N; i++) {
         salida.print("      " + i + ": " + valor);
         if (i < N)
            salida.print(",");
         salida.print("");
      }
      salida.println("]#;");
```

```
    }

    //Subprograma que escribe E

        private static void escribirE(PrintStream salida, String nombre, int N)
{
        salida.print(nombre + " = {"); for (int i=1; i<=N; i++) {
           salida.print(i + " " );
        }
        salida.println("};");
    }

    // Subprograma que escribela eficiencia

        private static void escribirEficiencia(PrintStream salida, double[][]
eficiencia) {
        salida.println("eficiencia = #[");
        for (int i=0; i<eficiencia.length; i++) {
           salida.print("         " + (i+1) + ": ");
           salida.println("#[");
           for (int j=0; j<eficiencia[i].length; j++) {
              salida.print("             " + (j+1) + ": " + eficiencia[i][j]);
              if (j < (eficiencia[i].length-1))
                 salida.println(",");
           }
           salida.print("]#");
           if (i < (eficiencia.length-1))
              salida.print(",");
           salida.println("");
        }
        salida.println("]#;");
    }


    // Subprograma que escribela penalización

    private static void escribirPenalizacion(PrintStream salida, double[][]
penalizacion) {
        salida.println("penalizacion = #[");
        for (int i=0; i<penalizacion.length; i++) {
           salida.print("         " + (i+1) + ": ");
           salida.println("#[");
           for (int j=0; j<penalizacion[i].length; j++) {
              salida.print("             " + (j+1) + ": " + penalizacion[i][j]);
              if (j < (penalizacion[i].length-1))
                 salida.println(",");
           }
```

```
            salida.print("]#");
            if (i < (penalizacion.length-1))
               salida.print(",");
            salida.println("");
         }
         salida.println("]#;");
      }

      // Subprogramas que escriben los tres tipos de demanda

         private static void escribirDemanda1(PrintStream salida, int Hi, int
num_trabajadores,int num_tareas, int T, double ruido) {
         double demanda_anual = (num_trabajadores*Hi);
            double demanda_semanal = (demanda_anual)/46;
            double demanda_semanal_tarea = demanda_semanal / num_tareas;
           salida.println("rtk=#[");

            for (int t=1; t<(T+1); t++) {
          salida.println( t  + ":#[");
             for(int k=1; k<(num_tareas+1); k++){
               double  na= Math.random();
                double na_margen = (na * 10)-5;
                double demanda_aleat = demanda_semanal_tarea + na_margen;
                salida.println(k + ":" + demanda_aleat + ",");

               }
            salida.println("]#,");
            }

            salida.println("]#;");
         }


         //Escribir la demanda de tipo 2:

         private static void escribirDemanda2(PrintStream salida, int Hi, int
num_trabajadores,int num_tareas, int T, double ruido) {

            double demanda_anual = (num_trabajadores*Hi);
            double demanda_semanal = (demanda_anual)/46;
            double demanda_semanal_tarea = demanda_semanal/ num_tareas;

            salida.println("rtk=#[");

            for (int t=1; t<(T+1); t++) {

            salida.println( t  + ":#[");
```

```
            for(int k=1; k<(num_tareas+1); k++){
            double  na= Math.random();
            double na_margen = (na * 10)-5;    //hace que el número
aleatorio esté entre -5 y 5

            double demanda_seno = ((Math.sin ( ((t-1)* 2 * Math.PI  /52) -
1))*0.25*demanda_semanal_tarea)+ demanda_semanal_tarea;

            double demanda_aleat = demanda_seno + na_margen;
            salida.println(k + ":" + demanda_aleat + ",");

            }
          salida.println("]#,");
          }

          salida.println("]#;");

      }

      //Escribir la demanda del tipo 3:

      private static void escribirDemanda3(PrintStream salida, int Hi, int
num_trabajadores,int num_tareas, int T, double ruido) {

          double demanda_anual = (num_trabajadores*Hi);
          double demanda_semanal = (demanda_anual)/46;
          double demanda_semanal_tarea = demanda_semanal/ num_tareas;
         salida.println("rtk=#[");
         for (int t=1; t<(T+1); t++) {
        salida.println( t  + ":#[");
           for(int k=1; k<(num_tareas+1); k++){
            double  na= Math.random();
            double na_margen = (na * 10)-5;
            double demanda_seno = ((Math.sin ( ((t-1)* 2 * Math.PI  /26) +
5))*0.25*demanda_semanal_tarea)+ demanda_semanal_tarea;

            double demanda_aleat = demanda_seno + na_margen;
            salida.println(k + ":" + demanda_aleat + ",");

            }
          salida.println("]#,");
          }

          salida.println("]#;");

      }
```

```
   // Subprograma que escribeEj

   private static void escribirEj(PrintStream salida, trabajador[] E, int
num_categorias) {

      salida.print("Ej = #[");

      for (int l=0; l<num_categorias; l++) {
         salida.print("      " + (l+1) + ":{");

         for(int i=1; i< E.length; i++){
            if(E[i].categoria==(l+1))
            salida.print(i+",");
            }
          if (l < (num_categorias-1))
          salida.print("},");
          salida.println("");
      }

      salida.println("}]#;");
   }

   // Subprograma que escribe Ck

      private static void escribirCk(PrintStream salida, double[][]
eficiencia, int num_tareas) {
      salida.println("Ck = #[");
      for (int i=0; i<num_tareas; i++) {
      salida.print("      " + (i+1) + ":{");
         for(int j=0; j<eficiencia.length;j++){
           if(eficiencia[j][i]>0)
           salida.print((j+1) + ",");
            }
         if (i < (num_tareas-1))
            salida.print("}");
         salida.println("");
      }
      salida.println("}]#;");
   }

   // Subprograma que escribeFj

   private static void escribirFj(PrintStream salida, double[][] eficiencia,
int num_categorias) {

      salida.println("Fj = #[");
```

```
    for (int i=0; i<num_categorias; i++) {
    salida.print("      " + (i+1) + ":{");
       for(int j=0; j<eficiencia.length;j++){
         if(eficiencia[i][j]>0)
         salida.print((j+1) + ",");
           }
       if (i < (num_categorias-1))
          salida.print("}");
       salida.println("");
    }
    salida.println("}]#;");
  }


  // Subprograma que escribeel conjunto de semanas disponibles

  private static void escribirAi(PrintStream salida, int
numero_trabajadores, int T) {
  salida.println("Ai=#[");
  for(int i=1; i<(numero_trabajadores+1);i++){
  salida.println(i+ ":{");
  double  nasummer= Math.random();
  double ini_summer_aleat =nasummer*9.00+28.00;
  int ini_summer = (int)Math.floor(ini_summer_aleat);

  double nawinter= Math.random();
  double ini_winter_aleat= nawinter *7.00+15.00;
  int ini_winter= (int)Math.floor(ini_winter_aleat);

    for(int j=1; j<(T+1);j++){

       if(j!=ini_summer){
          if(j!=(ini_summer + 1)){
             if(j!=(ini_summer + 2)){
                if((j!=ini_summer + 3)){
                   if(j!=ini_winter){
                      if(j!=(ini_winter + 1)){
                      salida.print(j + ",");
                          }
                   }
             }  }
          }
       }


    }
  salida.println("},");
```

```
    }
    salida.println("]#;");
    }

     // Subprograma que escribe los tres tipos de nueva demanda


          private static void escribirDemanda1nueva(PrintStream salida, int
Hi, int num_trabajadores,int num_tareas, int T, double ruido) {
        double demanda_anual = (num_trabajadores*Hi);
           double demanda_semanal = (demanda_anual)/46;
           double demanda_semanal_tarea = demanda_semanal / num_tareas;
           salida.println("rtknueva=#[");
           for (int t=1; t<(T+1); t++) {
          salida.println( t  + ":#[");
           for(int k=1; k<(num_tareas+1); k++){
              double  na= Math.random();
              double na_margen = (na * 10)-5;
              double demanda_aleat = demanda_semanal_tarea + na_margen;
              salida.println(k + ":" + demanda_aleat + ",");

              }
           salida.println("]#,");
           }

           salida.println("]#;");

        }

        //Escribir la demanda de tipo 2-nueva:

        private static void escribirDemanda2nueva(PrintStream salida, int Hi,
int num_trabajadores,int num_tareas, int T, double ruido) {

           double demanda_anual = (num_trabajadores*Hi);
           double demanda_semanal = (demanda_anual)/46;
           double demanda_semanal_tarea = demanda_semanal/ num_tareas;

           salida.println("rtknueva=#[");

           for (int t=1; t<(T+1); t++) {

           salida.println( t  + ":#[");

              for(int k=1; k<(num_tareas+1); k++){
              double  na= Math.random();
```

```
            double na_margen = (na * 10)-5;    //hace que el número
aleatorio esté entre -5 y 5

            double demanda_seno = ((Math.sin ( ((t-1-4)* 2 * Math.PI  /52) -
1 ))*0.25*demanda_semanal_tarea)+ demanda_semanal_tarea;

            double demanda_aleat = demanda_seno + na_margen;
            salida.println(k + ":" + demanda_aleat + ",");
           }
         salida.println("]#,");
         }
       salida.println("]#;");

      }

      //Escribir la demanda del tipo 3-nueva:

      private static void escribirDemanda3nueva(PrintStream salida, int Hi,
int num_trabajadores,int num_tareas, int T, double ruido) {

         double demanda_anual = (num_trabajadores*Hi);
         double demanda_semanal = (demanda_anual)/46;
         double demanda_semanal_tarea = demanda_semanal/ num_tareas;
         salida.println("rtknueva=#[");
         for (int t=1; t<(T+1); t++) {
         salida.println( t  + ":#[");
          for(int k=1; k<(num_tareas+1); k++){
          double  na= Math.random();
           double na_margen = (na * 10)-5;
             double demanda_seno = ((Math.sin ( ((t-1-4)* 2 * Math.PI  /26)
+ 5))*0.25*demanda_semanal_tarea)+ demanda_semanal_tarea;

            double demanda_aleat = demanda_seno + na_margen;
            salida.println(k + ":" + demanda_aleat + ",");

            }
         salida.println("]#,");
         }

         salida.println("]#;");

      }
}
```

C Resultados experimentales

A continuación se muestran los tiempos de ejecución necesarios para resolver los modelos de planificación y replanificación de los 675 ejemplares resueltos. Se encuentran ordenados en las tablas c.1 a la c.25, cada una de las cuales presenta un grupo de ejemplares con todas las posibles combinaciones del número de empleados (E), demanda y eficiencia.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	1	0,313	0,218	0,250	1,125	2,375	155,174
1	1	50	1	0,516	0,500	0,438	2,781	5,719	260,832
1	1	100	1	1,438	1,297	0,922	139,518	15,453	230,186
1	2	25	1	0,343	0,359	0,250	0,968	1,719	1,078
1	2	50	1	0,672	0,812	0,547	1,937	2,954	1,828
1	2	100	1	1,579	2,859	1,312	5,875	10,531	5,251
1	3	25	1	0,328	0,468	0,281	2,562	1,703	1,516
1	3	50	1	0,672	0,797	0,438	1,813	4,703	3,204
1	3	100	1	1,578	2,594	1,000	4,235	16,297	10,750
2	1	25	1	0,344	0,203	0,250	1,422	2,844	3,594
2	1	50	1	0,515	0,453	0,437	2,515	5,422	8,157
2	1	100	1	1,281	1,375	0,906	200,545	16,031	36,375
2	2	25	1	0,562	0,437	0,282	1,125	1,562	1,157
2	2	50	1	0,750	1,188	0,531	1,938	4,422	1,734
2	2	100	1	1,656	2,860	1,250	4,500	10,187	4,250
2	3	25	1	0,328	0,391	0,266	50,703	1,844	1,125
2	3	50	1	0,672	1,141	0,500	3,188	4,640	2,375
2	3	100	1	1,578	2,750	0,984	4,485	13,813	6,719
3	1	25	1	0,219	0,141	0,203	0,766	1,954	1,531
3	1	50	1	0,437	0,250	0,375	55,547	4,047	4,031
3	1	100	1	0,828	0,610	0,656	6,251	14,141	11,656
3	2	25	1	0,234	0,156	0,187	1,500	1,454	1,422
3	2	50	1	0,422	0,297	0,328	2,235	4,484	3,094
3	2	100	1	0,954	1,000	0,765	4,109	14,828	6,860
3	3	25	1	0,234	0,172	0,187	1,438	1,672	1,391
3	3	50	1	0,437	0,343	0,313	2,016	4,500	2,766
3	3	100	1	0,922	0,828	0,703	7,766	11,703	7,047

Tabla c.1. Valores de los tiempos de resolución de los modelos para el grupo 1.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	2	4,094	0,235	0,328	1,750	2,141	91,502
1	1	50	2	0,563	0,531	0,500	7,656	5,265	7,782
1	1	100	2	1,109	1,390	1,110	9,172	15,625	35,797
1	2	25	2	0,391	0,516	0,312	1,297	2,109	1,110
1	2	50	2	0,797	0,906	0,578	2,078	5,563	1,859
1	2	100	2	1,594	2,781	1,281	5,328	20,454	4,985
1	3	25	2	0,344	0,485	0,297	0,984	1,422	1,890
1	3	50	2	0,625	1,079	0,547	3,046	4,891	2,640
1	3	100	2	1,593	2,594	0,984	5,640	15,844	7,500
2	1	25	2	5,937	1,844	0,984	1,579	3,203	4,703
2	1	50	2	0,671	0,609	0,500	2,188	5,344	8,656
2	1	100	2	1,344	0,985	1,000	6,859	16,188	33,516
2	2	25	2	0,359	0,344	0,297	1,031	1,563	0,859
2	2	50	2	0,766	0,875	0,547	2,156	4,797	1,875
2	2	100	2	1,532	2,562	1,125	3,812	143,721	4,812
2	3	25	2	0,344	0,532	0,282	2,891	1,625	1,266
2	3	50	2	0,594	1,015	0,531	3,657	4,797	3,219
2	3	100	2	1,656	2,813	1,063	8,000	15,219	14,469
3	1	25	2	0,250	0,172	0,219	57,247	16,282	16,157
3	1	50	2	0,453	0,265	0,360	78,255	51,438	51,672
3	1	100	2	0,844	0,657	0,687	5,344	15,204	12,844
3	2	25	2	0,234	0,172	0,188	98,355	0,985	1,000
3	2	50	2	0,437	0,328	0,344	1,500	4,312	3,406
3	2	100	2	1,015	0,938	0,750	54,658	35,346	43,355
3	3	25	2	0,219	0,188	0,188	9,297	1,360	1,219
3	3	50	2	0,422	0,328	0,344	2,140	4,312	2,984
3	3	100	2	0,953	0,875	0,703	5,109	14,094	7,344

Tabla c.2. Valores de los tiempos de resolución de los modelos para el grupo 2.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	3	0,313	0,204	0,250	1,703	2,469	3,610
1	1	50	3	0,516	0,500	0,437	3,938	5,093	10,672
1	1	100	3	1,953	3,516	7,531	7,579	15,907	580,171
1	2	25	3	0,359	0,407	0,297	1,140	1,547	1,000
1	2	50	3	0,719	0,797	0,579	1,891	3,438	1,844
1	2	100	3	1,500	2,938	1,047	4,016	8,375	4,391
1	3	25	3	0,344	0,422	0,281	3,016	1,296	1,188
1	3	50	3	0,672	0,875	0,468	2,829	3,859	2,734
1	3	100	3	1,594	2,469	1,016	5,812	15,469	10,906
2	1	25	3	0,281	0,188	0,250	1,594	2,515	3,156
2	1	50	3	0,687	0,516	0,578	189,940	5,078	9,156
2	1	100	3	1,656	1,422	0,953	82,123	23,813	30,657
2	2	25	3	0,359	0,390	0,281	1,046	1,719	0,844
2	2	50	3	0,703	0,813	0,562	2,141	6,906	1,750
2	2	100	3	1,344	2,875	1,047	3,875	10,047	4,922
2	3	25	3	0,344	0,375	0,281	0,969	1,281	1,094
2	3	50	3	0,641	0,985	0,484	1,812	4,563	2,078
2	3	100	3	1,578	2,875	1,032	4,937	14,547	6,266
3	1	25	3	0,218	0,156	0,188	64,217	14,047	12,657
3	1	50	3	0,391	0,219	0,375	3,141	5,187	49,297
3	1	100	3	0,938	0,672	0,828	6,281	15,203	167,205
3	2	25	3	0,375	0,234	0,234	3,078	1,718	1,187
3	2	50	3	0,484	0,390	0,391	2,625	5,000	3,360
3	2	100	3	1,046	0,797	0,812	4,953	13,937	8,594
3	3	25	3	0,266	0,219	0,250	21,214	1,156	1,156
3	3	50	3	0,468	0,390	0,375	105,358	58,349	49,355
3	3	100	3	0,953	0,891	0,734	5,594	13,828	8,625

Tabla c.3. Valores de los tiempos de resolución de los modelos para el grupo 3.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	4	0,312	0,250	0,297	1,859	2,656	3,891
1	1	50	4	0,625	0,594	0,500	5,219	5,438	10,609
1	1	100	4	1,203	1,391	0,953	7,422	18,609	360,155
1	2	25	4	0,360	0,453	0,312	1,047	1,375	1,204
1	2	50	4	0,859	0,954	0,610	2,235	5,266	1,797
1	2	100	4	1,531	2,844	1,219	4,000	10,110	4,907
1	3	25	4	0,406	0,454	0,312	1,141	1,391	1,406
1	3	50	4	0,672	0,906	0,500	2,516	4,469	2,843
1	3	100	4	1,531	3,047	1,047	6,281	14,063	11,641
2	1	25	4	0,313	0,266	0,281	1,078	2,203	3,110
2	1	50	4	0,578	0,547	0,484	5,750	5,922	12,406
2	1	100	4	1,125	1,281	1,000	6,687	14,500	360,139
2	2	25	4	0,469	0,531	0,312	1,000	1,500	0,891
2	2	50	4	0,719	1,016	0,609	2,172	4,469	1,766
2	2	100	4	1,532	2,750	1,297	5,671	463,943	5,390
2	3	25	4	0,343	0,531	0,281	1,156	1,437	1,406
2	3	50	4	0,672	0,907	0,500	1,625	4,469	2,984
2	3	100	4	1,578	2,719	1,000	5,172	16,438	7,532
3	1	25	4	0,250	0,188	0,219	94,650	1,156	1,156
3	1	50	4	0,438	0,281	0,360	3,063	5,328	3,297
3	1	100	4	0,875	0,656	0,766	7,687	20,048	15,797
3	2	25	4	0,890	0,453	1,125	68,687	1,203	1,188
3	2	50	4	0,516	0,437	0,391	65,217	3,594	3,547
3	2	100	4	0,984	1,000	0,797	5,094	14,375	7,156
3	3	25	4	0,250	0,203	0,219	2,688	1,468	4,282
3	3	50	4	0,469	0,359	0,375	2,156	4,484	3,360
3	3	100	4	0,953	0,906	0,750	4,390	14,516	8,157

Tabla c.4. Valores de los tiempos de resolución de los modelos para el grupo 4.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	5	0,343	0,297	0,297	2,719	2,125	3,781
1	1	50	5	0,562	0,563	0,484	2,625	5,766	68,154
1	1	100	5	0,265	0,360	153,256	58,369	236,126	45,275
1	2	25	5	0,360	0,336	0,359	1,515	1,406	0,969
1	2	50	5	0,797	0,859	0,562	2,172	4,796	1,813
1	2	100	5	1,640	2,953	1,235	4,719	17,219	5,141
1	3	25	5	0,375	0,500	0,328	1,203	1,765	1,047
1	3	50	5	0,704	0,891	0,531	2,500	4,750	2,312
1	3	100	5	1,657	3,125	1,016	5,297	15,110	9,922
2	1	25	5	0,343	0,266	0,282	1,594	2,219	2,656
2	1	50	5	0,578	0,578	0,484	4,031	5,985	9,484
2	1	100	5	1,344	1,375	1,062	7,141	19,687	213,248
2	2	25	5	0,391	0,484	0,313	0,907	1,532	0,938
2	2	50	5	0,797	1,093	0,593	1,891	6,250	1,750
2	2	100	5	1,500	2,954	1,235	4,812	754,916	5,172
2	3	25	5	0,390	0,454	0,328	1,297	1,781	1,282
2	3	50	5	0,703	0,859	0,469	2,000	4,406	2,984
2	3	100	5	1,610	2,578	1,062	6,547	14,047	10,453
3	1	25	5	0,250	0,187	0,219	54,238	16,859	16,734
3	1	50	5	0,453	0,265	0,375	1,547	6,109	4,187
3	1	100	5	0,922	0,609	0,797	6,563	19,266	127,299
3	2	25	5	0,344	0,329	0,437	34,355	1,297	1,218
3	2	50	5	0,516	0,407	0,469	2,235	4,484	3,375
3	2	100	5	1,625	1,187	2,094	5,656	13,907	7,484
3	3	25	5	0,266	0,234	0,250	2,891	1,625	1,406
3	3	50	5	0,484	0,375	0,375	6,750	4,360	3,125
3	3	100	5	0,985	1,110	1,297	5,500	13,531	8,390

Tabla c.5. Valores de los tiempos de resolución de los modelos para el grupo 5.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	6	0,359	0,297	0,281	1,969	2,844	3,266
1	1	50	6	0,609	0,516	0,469	2,735	5,250	43,354
1	1	100	6	0,312	0,698	0,357	7,225	4,240	6,548
1	2	25	6	0,391	0,406	0,296	1,250	1,407	0,859
1	2	50	6	0,812	0,922	0,578	2,140	5,531	1,812
1	2	100	6	1,547	3,031	1,156	5,391	14,500	5,282
1	3	25	6	0,344	0,515	0,297	2,047	1,266	1,282
1	3	50	6	0,672	1,031	0,547	1,937	4,765	2,172
1	3	100	6	1,531	3,875	1,078	5,079	16,313	14,375
2	1	25	6	0,360	0,282	0,312	1,843	2,719	3,203
2	1	50	6	0,641	0,750	0,485	3,484	5,859	9,297
2	1	100	6	1,360	1,516	1,141	29,546	6,242	50,355
2	2	25	6	0,531	0,437	0,312	0,922	1,969	1,187
2	2	50	6	0,734	0,906	0,562	1,922	4,781	1,750
2	2	100	6	1,531	3,219	1,234	3,875	11,688	5,125
2	3	25	6	0,359	0,500	0,328	2,781	1,531	1,250
2	3	50	6	0,656	0,922	0,515	2,782	4,562	2,813
2	3	100	6	1,531	2,828	1,110	5,765	14,375	7,001
3	1	25	6	0,250	0,172	0,219	130,189	1,844	17,797
3	1	50	6	0,453	0,266	0,375	2,500	5,281	50,953
3	1	100	6	1,015	0,672	0,766	5,937	17,453	152,939
3	2	25	6	0,859	0,219	0,281	2,532	1,891	1,297
3	2	50	6	0,719	0,453	0,407	2,704	4,547	3,329
3	2	100	6	1,562	0,984	0,891	5,609	14,484	8,328
3	3	25	6	0,266	0,219	0,219	1,735	1,594	1,594
3	3	50	6	0,453	0,344	0,359	1,468	4,625	3,046
3	3	100	6	1,015	0,954	1,266	59,000	8,159	29,000

Tabla c.6. Valores de los tiempos de resolución de los modelos para el grupo 6.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	7	0,735	0,250	0,297	2,047	3,188	3,609
1	1	50	7	0,547	0,516	0,500	3,515	6,016	142,425
1	1	100	7	0,267	0,469	1,658	57,245	156,357	264,246
1	2	25	7	0,359	0,437	0,312	0,922	1,516	0,953
1	2	50	7	0,891	0,937	0,609	2,984	3,282	2,015
1	2	100	7	1,625	2,843	1,156	5,672	8,219	4,922
1	3	25	7	0,375	0,468	0,312	2,344	1,672	1,219
1	3	50	7	0,703	1,016	0,485	1,829	4,453	2,969
1	3	100	7	1,640	2,578	1,000	5,828	15,641	6,875
2	1	25	7	0,328	0,266	0,297	2,204	2,531	2,969
2	1	50	7	0,563	0,516	0,531	1,906	5,593	6,844
2	1	100	7	1,156	1,390	1,032	22,844	19,328	102,355
2	2	25	7	0,422	0,656	0,297	1,781	1,985	1,032
2	2	50	7	0,734	1,047	0,547	1,844	36,141	2,094
2	2	100	7	1,532	2,890	1,235	3,969	9,531	5,125
2	3	25	7	0,390	0,484	0,281	1,063	1,329	1,532
2	3	50	7	0,703	0,844	0,547	1,969	4,578	3,078
2	3	100	7	1,562	2,750	1,125	7,937	15,938	11,516
3	1	25	7	0,266	3,570	1,124	43,355	8,539	69,357
3	1	50	7	0,438	0,235	0,375	2,140	5,047	45,720
3	1	100	7	0,984	0,687	0,734	4,813	18,735	122,002
3	2	25	7	0,250	0,218	0,234	65,687	1,032	1,016
3	2	50	7	0,484	0,375	0,375	24,245	2,781	2,797
3	2	100	7	1,047	1,094	0,828	5,266	13,953	8,578
3	3	25	7	0,265	0,218	0,235	1,453	1,828	1,375
3	3	50	7	0,453	0,391	0,360	3,281	4,547	3,469
3	3	100	7	1,000	0,969	0,750	5,640	20,984	13,781

Tabla c.7. Valores de los tiempos de resolución de los modelos para el grupo 7.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	8	2,188	0,375	0,579	3,109	3,141	3,578
1	1	50	8	0,562	0,625	0,563	2,687	5,375	9,703
1	1	100	8	1,500	1,406	1,359	7,094	20,281	36,987
1	2	25	8	0,360	0,438	0,312	0,953	1,703	0,938
1	2	50	8	0,688	1,203	0,547	2,109	5,531	2,063
1	2	100	8	1,453	2,812	1,219	4,985	505,772	5,360
1	3	25	8	0,391	0,453	0,297	1,093	1,625	1,360
1	3	50	8	0,672	0,796	0,547	1,859	4,219	2,704
1	3	100	8	1,672	2,859	1,250	5,125	16,172	11,953
2	1	25	8	0,313	0,250	0,266	1,453	2,609	3,047
2	1	50	8	0,594	0,562	0,484	8,766	5,515	8,594
2	1	100	8	1,250	1,750	1,000	9,625	18,234	32,266
2	2	25	8	0,531	0,484	0,360	2,594	1,968	1,078
2	2	50	8	0,750	0,891	0,640	1,937	326,551	2,234
2	2	100	8	1,859	4,375	3,265	5,094	281,301	5,640
2	3	25	8	0,391	0,531	0,391	2,015	2,157	1,360
2	3	50	8	0,703	1,000	0,609	3,672	6,016	3,313
2	3	100	8	1,625	3,234	1,157	7,281	13,828	6,251
3	1	25	8	0,250	0,203	0,219	98,355	14,547	14,407
3	1	50	8	0,453	0,326	0,213	109,235	16,545	305,245
3	1	100	8	1,047	0,672	0,782	5,578	13,688	163,939
3	2	25	8	0,265	0,325	9,234	26,355	10,357	36,245
3	2	50	8	0,469	0,375	0,375	1,969	5,125	3,016
3	2	100	8	1,016	0,953	0,781	4,313	15,454	7,719
3	3	25	8	0,265	0,219	0,234	55,204	1,501	4,594
3	3	50	8	0,469	0,375	0,360	2,781	4,015	2,844
3	3	100	8	0,954	0,828	0,750	4,562	14,203	8,578

Tabla c.8. Valores de los tiempos de resolución de los modelos para el grupo 8.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	9	0,329	0,281	0,312	1,625	2,391	3,906
1	1	50	9	0,625	0,563	0,484	2,563	5,563	849,282
1	1	100	9	1,141	1,094	1,094	7,735	16,265	269,163
1	2	25	9	0,359	0,500	0,344	0,906	1,610	1,031
1	2	50	9	0,734	1,016	0,610	2,109	20,469	1,797
1	2	100	9	1,453	2,765	1,187	7,531	809,401	5,360
1	3	25	9	0,375	0,562	0,297	1,110	1,235	1,203
1	3	50	9	0,688	0,844	0,500	3,718	4,937	2,984
1	3	100	9	1,609	2,750	1,016	4,688	15,953	12,219
2	1	25	9	0,313	0,297	0,282	1,641	2,109	2,891
2	1	50	9	0,610	0,578	0,468	3,891	5,187	7,812
2	1	100	9	1,234	1,281	1,016	8,454	16,844	29,532
2	2	25	9	0,390	0,500	0,359	1,437	1,438	1,047
2	2	50	9	0,719	1,203	0,562	1,781	4,031	1,843
2	2	100	9	1,469	2,875	1,188	3,922	23,469	6,422
2	3	25	9	1,375	0,656	0,625	2,547	1,562	1,219
2	3	50	9	0,750	0,954	0,484	2,328	4,985	1,969
2	3	100	9	2,531	2,891	2,093	6,625	16,828	7,625
3	1	25	9	0,281	0,187	0,219	32,547	17,532	17,329
3	1	50	9	0,453	0,281	0,375	2,609	5,266	55,766
3	1	100	9	0,984	0,579	0,766	5,937	18,437	197,237
3	2	25	9	0,296	0,235	0,234	1,359	1,735	1,437
3	2	50	9	0,500	0,359	0,391	1,500	4,469	2,969
3	2	100	9	0,984	1,438	1,343	6,437	12,953	9,578
3	3	25	9	0,281	0,203	0,234	1,187	1,813	1,406
3	3	50	9	0,469	0,391	0,390	329,346	61,266	592,100
3	3	100	9	0,985	0,906	0,734	4,797	12,407	7,937

Tabla c.9. Valores de los tiempos de resolución de los modelos para el grupo 9.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	10	0,313	0,250	0,297	1,891	2,250	3,954
1	1	50	10	0,563	0,547	0,484	0,599	0,988	36,588
1	1	100	10	1,531	1,344	0,953	6,625	17,140	136,085
1	2	25	10	0,391	0,485	0,313	1,235	1,531	0,875
1	2	50	10	0,750	0,828	0,578	3,328	5,297	1,782
1	2	100	10	1,593	2,813	1,250	4,219	748,806	4,906
1	3	25	10	0,359	0,484	0,343	2,187	1,938	1,171
1	3	50	10	0,703	1,218	0,515	1,969	4,672	2,859
1	3	100	10	1,531	2,781	1,000	4,656	16,015	8,079
2	1	25	10	0,312	0,266	0,281	1,579	2,843	2,969
2	1	50	10	0,547	0,563	0,516	1,985	5,531	6,719
2	1	100	10	1,360	1,313	1,250	7,109	15,157	192,513
2	2	25	10	0,375	0,437	0,297	1,031	1,578	1,125
2	2	50	10	0,765	1,063	0,594	2,079	4,312	1,734
2	2	100	10	1,500	2,843	1,187	3,734	7,126	4,859
2	3	25	10	0,375	0,515	0,313	2,047	1,563	1,985
2	3	50	10	0,625	0,984	0,484	1,875	4,610	3,219
2	3	100	10	1,656	2,625	1,062	4,750	14,828	6,453
3	1	25	10	0,250	0,203	0,219	4,985	1,469	1,454
3	1	50	10	0,438	0,266	0,344	4,250	5,797	40,282
3	1	100	10	0,890	0,610	0,735	5,938	24,922	93,079
3	2	25	10	1,047	0,359	2,796	5,655	3,955	153,652
3	2	50	10	0,438	0,359	0,375	2,031	4,329	3,031
3	2	100	10	0,984	1,093	0,781	5,016	12,016	7,750
3	3	25	10	0,265	0,204	0,234	6,655	4,252	196,641
3	3	50	10	0,469	0,375	0,375	2,172	4,765	3,266
3	3	100	10	0,984	0,968	0,750	4,891	12,641	8,188

Tabla c.10. Valores de los tiempos de resolución de los modelos para el grupo 10.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	11	0,328	0,266	0,281	1,922	2,531	56,689
1	1	50	11	0,563	0,547	0,453	2,656	5,453	1205,039
1	1	100	11	1,281	1,266	1,093	8,078	19,297	1211,102
1	2	25	11	0,391	0,422	0,344	1,172	1,578	1,031
1	2	50	11	0,703	0,922	0,671	13,891	6,313	1,828
1	2	100	11	1,453	3,047	1,172	4,282	254,941	5,390
1	3	25	11	0,343	0,516	0,297	1,813	1,515	1,703
1	3	50	11	0,688	1,000	0,563	1,719	4,390	3,297
1	3	100	11	1,485	2,860	1,078	5,219	14,281	6,235
2	1	25	11	0,313	0,281	0,297	1,953	2,344	2,969
2	1	50	11	0,672	0,562	0,485	2,859	5,266	9,969
2	1	100	11	1,563	1,453	1,797	7,453	15,313	232,078
2	2	25	11	0,375	0,547	0,313	0,953	1,516	1,109
2	2	50	11	0,735	0,969	0,610	2,266	5,578	1,969
2	2	100	11	1,547	2,938	1,172	5,125	11,907	4,937
2	3	25	11	0,360	0,469	0,312	2,297	2,031	1,375
2	3	50	11	0,688	0,875	0,578	1,891	4,641	2,234
2	3	100	11	1,672	3,016	0,953	4,672	15,531	7,485
3	1	25	11	0,250	0,188	0,218	23,655	15,172	15,063
3	1	50	11	0,438	0,344	0,375	2,968	5,235	63,673
3	1	100	11	0,969	0,781	0,734	5,078	23,032	13,890
3	2	25	11	0,516	0,265	0,282	23,259	1,312	1,203
3	2	50	11	0,453	0,406	0,375	2,688	4,422	3,438
3	2	100	11	1,563	0,937	1,031	5,797	14,078	9,265
3	3	25	11	0,313	0,218	0,250	1,828	1,437	1,188
3	3	50	11	0,469	0,328	0,360	2,032	4,360	2,876
3	3	100	11	1,000	0,875	0,782	7,360	14,875	12,079

Tabla c.11. Valores de los tiempos de resolución de los modelos para el grupo 11.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	12	1,531	0,312	0,719	1,735	2,531	68,345
1	1	50	12	0,547	0,563	0,531	2,906	5,438	10,188
1	1	100	12	1,188	1,282	1,094	8,078	14,969	569,478
1	2	25	12	0,547	0,500	0,312	0,844	1,578	1,047
1	2	50	12	0,751	0,921	0,563	1,985	4,453	1,953
1	2	100	12	1,579	2,765	1,219	4,797	164,456	4,890
1	3	25	12	0,359	0,500	0,313	1,297	1,375	1,359
1	3	50	12	0,672	1,125	0,515	3,078	4,828	2,656
1	3	100	12	1,485	2,985	1,000	4,797	14,922	10,750
2	1	25	12	0,329	0,266	0,281	0,735	2,515	2,828
2	1	50	12	0,593	0,500	0,516	2,797	5,593	9,187
2	1	100	12	1,219	1,031	1,032	10,063	20,172	28,735
2	2	25	12	0,579	0,468	0,406	2,062	1,828	1,094
2	2	50	12	0,718	1,203	0,672	2,094	109,986	1,969
2	2	100	12	1,531	2,891	1,765	4,843	8,187	5,953
2	3	25	12	0,375	0,469	0,328	2,078	1,766	1,344
2	3	50	12	0,672	0,875	0,500	1,734	4,578	2,469
2	3	100	12	1,625	2,750	1,094	6,750	16,501	7,985
3	1	25	12	0,297	0,203	0,235	1,562	1,844	1,594
3	1	50	12	0,453	0,328	0,406	3,000	5,235	3,438
3	1	100	12	0,906	0,625	0,750	5,156	20,156	137,783
3	2	25	12	0,312	0,219	0,235	1,000	1,500	1,250
3	2	50	12	0,469	0,422	0,375	2,219	4,485	4,125
3	2	100	12	0,954	0,937	0,797	5,063	13,282	7,047
3	3	25	12	0,250	0,219	0,219	26,615	1,344	1,360
3	3	50	12	0,453	0,328	0,360	2,329	4,562	3,344
3	3	100	12	0,984	0,938	0,719	4,875	12,578	7,359

Tabla c.12. Valores de los tiempos de resolución de los modelos para el grupo 12.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	13	0,360	0,281	0,281	1,703	2,391	141,456
1	1	50	13	0,594	0,563	0,453	5,015	5,562	9,313
1	1	100	13	1,422	1,390	1,000	6,078	19,219	378,241
1	2	25	13	0,469	0,406	0,297	0,797	1,391	0,782
1	2	50	13	0,703	0,859	0,579	2,016	5,156	1,672
1	2	100	13	1,609	2,875	1,203	5,813	133,206	4,625
1	3	25	13	0,359	0,531	0,312	1,250	1,344	1,187
1	3	50	13	0,672	0,860	0,485	2,454	4,859	3,001
1	3	100	13	1,547	2,640	1,015	5,047	16,110	12,672
2	1	25	13	0,328	0,281	0,282	1,469	2,360	2,359
2	1	50	13	0,547	0,609	0,531	1,250	5,719	10,250
2	1	100	13	1,297	1,109	1,000	6,015	21,907	38,001
2	2	25	13	1,219	1,516	0,562	2,297	1,594	1,000
2	2	50	13	0,766	1,063	0,610	2,015	4,625	1,813
2	2	100	13	2,250	2,938	1,625	7,594	20,062	6,172
2	3	25	13	0,422	0,547	0,344	1,219	1,360	1,391
2	3	50	13	0,750	1,156	0,594	2,234	4,516	3,000
2	3	100	13	1,687	2,781	1,782	16,484	14,640	6,797
3	1	25	13	0,281	0,187	0,234	1,875	1,953	1,562
3	1	50	13	0,515	1,347	16,358	26,217	6,354	369,346
3	1	100	13	0,859	0,704	0,703	4,813	18,266	138,034
3	2	25	13	0,235	0,188	0,203	1,079	1,657	1,328
3	2	50	13	0,438	1,265	0,547	36,547	26,656	536,355
3	2	100	13	1,016	1,016	0,781	10,546	15,255	236,355
3	3	25	13	0,265	0,218	0,235	13,547	13,654	135,235
3	3	50	13	0,500	0,391	0,390	2,328	4,922	3,031
3	3	100	13	0,984	0,985	0,734	5,281	12,548	8,437

Tabla c.13. Valores de los tiempos de resolución de los modelos para el grupo 13.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	14	0,359	0,266	0,343	0,313	1,734	1,594
1	1	50	14	0,516	0,469	0,391	0,516	69,782	95,096
1	1	100	14	1,235	1,313	1,266	3,125	110,971	11,297
1	2	25	14	0,359	0,360	0,235	0,250	0,641	0,625
1	2	50	14	0,704	0,859	0,406	0,438	1,547	1,484
1	2	100	14	1,422	2,938	0,891	0,984	3,516	3,875
1	3	25	14	0,328	0,375	0,219	0,234	0,609	5,688
1	3	50	14	0,703	1,172	0,391	0,437	1,407	1,438
1	3	100	14	5,391	6,313	5,485	2,703	5,938	6,937
2	1	25	14	2,906	2,140	2,875	10,797	4,562	6,422
2	1	50	14	4,906	2,266	2,656	3,938	52,423	77,220
2	1	100	14	0,343	0,360	0,234	0,265	0,718	0,703
2	2	25	14	0,343	0,360	0,234	0,265	0,718	0,703
2	2	50	14	0,735	0,875	0,375	0,438	1,328	1,266
2	2	100	14	1,454	2,750	0,922	0,875	3,859	3,422
2	3	25	14	0,328	0,437	0,203	0,250	0,734	0,813
2	3	50	14	0,672	0,782	0,422	0,484	1,454	1,546
2	3	100	14	1,625	2,484	0,890	0,891	3,000	3,297
3	1	25	14	3,078	0,312	0,218	0,531	2,500	2,594
3	1	50	14	0,937	0,250	0,343	1,093	3,782	3,703
3	1	100	14	1,438	1,235	1,234	4,156	12,437	14,563
3	2	25	14	0,437	0,187	0,203	0,250	1,532	1,500
3	2	50	14	0,437	0,328	0,344	0,484	3,188	3,000
3	2	100	14	1,000	0,829	1,266	5,781	9,954	8,906
3	3	25	14	0,219	0,172	0,203	0,266	1,829	1,813
3	3	50	14	0,437	0,344	0,328	0,469	3,079	3,282
3	3	100	14	0,953	0,891	1,906	1,078	13,376	8,359

Tabla c.14. Valores de los tiempos de resolución de los modelos para el grupo 14.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	15	0,375	0,266	0,297	1,093	2,391	111,081
1	1	50	15	0,531	0,593	0,562	2,813	5,406	290,553
1	1	100	15	1,375	1,359	0,984	5,860	17,000	36,325
1	2	25	15	0,407	0,469	0,312	0,953	1,657	0,984
1	2	50	15	0,719	0,890	0,594	2,704	3,594	2,015
1	2	100	15	1,594	3,032	1,312	4,109	129,549	6,015
1	3	25	15	0,375	0,531	0,297	1,140	1,469	1,312
1	3	50	15	0,672	0,984	0,516	78,439	4,875	2,750
1	3	100	15	1,609	2,985	1,016	6,391	15,813	11,281
2	1	25	15	0,329	0,266	0,296	2,187	2,219	3,656
2	1	50	15	0,609	0,641	0,500	2,281	5,266	6,625
2	1	100	15	1,203	1,500	0,985	9,329	20,485	38,376
2	2	25	15	0,657	0,500	0,328	1,984	1,625	1,454
2	2	50	15	0,765	1,094	0,563	1,937	57,251	2,031
2	2	100	15	1,657	2,937	1,876	4,750	11,829	5,375
2	3	25	15	0,406	0,594	0,312	1,968	1,515	1,235
2	3	50	15	0,718	0,843	0,531	1,907	5,500	2,015
2	3	100	15	1,750	2,656	1,641	7,063	16,594	6,812
3	1	25	15	0,344	0,203	0,250	6,124	14,532	14,298
3	1	50	15	0,453	0,297	0,375	4,127	3,343	3,359
3	1	100	15	1,562	0,355	0,658	2,033	3,343	5,137
3	2	25	15	0,297	0,219	0,250	0,953	1,734	1,609
3	2	50	15	0,469	0,391	0,391	2,141	4,531	3,360
3	2	100	15	0,985	1,062	0,812	4,375	14,297	7,110
3	3	25	15	0,265	0,235	0,250	1,609	1,641	1,375
3	3	50	15	0,484	0,343	0,375	2,187	4,532	3,063
3	3	100	15	0,969	0,907	0,781	4,297	15,188	7,594

Tabla c.15. Valores de los tiempos de resolución de los modelos para el grupo 15.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	16	0,328	0,282	0,281	1,297	2,891	55,392
1	1	50	16	0,578	0,547	0,500	1,859	5,672	263,333
1	1	100	16	1,234	1,063	0,969	7,391	16,844	132,456
1	2	25	16	0,390	0,532	0,328	0,953	1,516	0,891
1	2	50	16	0,735	1,000	0,578	2,187	3,953	1,813
1	2	100	16	1,593	3,141	1,219	5,266	8,656	4,985
1	3	25	16	0,375	0,562	0,312	1,094	1,422	1,297
1	3	50	16	0,672	0,937	0,531	2,641	5,140	3,328
1	3	100	16	1,625	2,797	1,094	8,203	15,000	10,781
2	1	25	16	0,344	0,265	0,297	1,735	2,734	3,484
2	1	50	16	0,562	0,610	0,484	4,141	5,172	6,500
2	1	100	16	1,219	1,578	1,015	8,797	19,672	30,235
2	2	25	16	0,532	0,500	0,391	2,376	2,031	1,172
2	2	50	16	0,766	1,094	0,641	2,078	4,735	1,859
2	2	100	16	2,141	3,266	1,922	5,875	9,579	5,391
2	3	25	16	0,469	0,610	0,359	2,844	1,547	1,250
2	3	50	16	0,750	0,906	0,547	1,735	4,125	2,109
2	3	100	16	1,640	2,719	1,563	5,641	17,609	7,594
3	1	25	16	0,297	0,203	0,235	8,325	15,594	15,907
3	1	50	16	0,469	0,297	0,375	2,531	5,406	4,719
3	1	100	16	0,922	0,640	0,718	6,641	14,782	131,377
3	2	25	16	0,250	0,234	0,250	1,907	1,813	1,407
3	2	50	16	0,516	0,406	0,406	10,355	7,324	6,244
3	2	100	16	1,078	0,985	0,829	5,391	12,469	7,891
3	3	25	16	0,266	0,235	0,234	2,031	1,578	1,344
3	3	50	16	0,500	0,422	0,390	2,813	4,265	3,688
3	3	100	16	1,063	1,000	0,735	4,797	13,578	6,953

Tabla c.16. Valores de los tiempos de resolución de los modelos para el grupo 16.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	17	0,343	0,297	0,297	1,078	2,469	70,783
1	1	50	17	0,593	0,532	0,547	2,687	5,485	422,227
1	1	100	17	1,297	1,062	1,063	7,516	21,922	31,485
1	2	25	17	0,515	0,515	0,406	2,437	1,547	1,094
1	2	50	17	0,766	0,953	0,703	2,141	2,829	2,359
1	2	100	17	2,078	3,219	1,578	6,500	179,707	5,265
1	3	25	17	0,422	0,562	0,344	2,000	1,484	1,297
1	3	50	17	0,718	0,875	0,531	1,640	4,656	2,781
1	3	100	17	1,812	2,703	1,093	6,234	13,344	6,437
2	1	25	17	0,422	0,296	0,328	1,094	2,204	2,719
2	1	50	17	0,578	0,562	0,563	2,781	5,625	8,907
2	1	100	17	1,188	1,313	0,953	7,359	15,719	27,845
2	2	25	17	0,359	0,438	0,297	0,906	1,328	1,032
2	2	50	17	0,688	0,890	0,578	2,187	5,594	1,891
2	2	100	17	1,437	3,140	1,203	6,437	9,188	5,906
2	3	25	17	0,375	0,500	0,281	1,719	1,359	1,703
2	3	50	17	0,735	0,859	0,453	2,094	4,656	1,906
2	3	100	17	1,641	3,078	1,031	51,569	15,641	8,172
3	1	25	17	0,266	0,172	0,219	6,365	23,641	22,485
3	1	50	17	0,500	0,312	0,500	1,766	6,469	5,454
3	1	100	17	0,953	0,625	0,735	5,031	19,453	11,563
3	2	25	17	0,250	0,187	0,234	1,322	4,223	6,325
3	2	50	17	0,469	0,328	0,375	2,078	4,969	42,985
3	2	100	17	0,984	0,922	0,797	4,641	13,610	6,953
3	3	25	17	0,250	0,188	0,219	1,156	1,547	1,172
3	3	50	17	0,453	0,296	0,360	1,985	4,328	3,375
3	3	100	17	0,953	0,906	0,750	4,734	12,953	8,703

Tabla c.17. Valores de los tiempos de resolución de los modelos para el grupo 17.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	18	0,312	0,218	0,266	1,625	2,282	199,532
1	1	50	18	0,563	0,594	0,532	8,126	5,093	296,767
1	1	100	18	1,313	1,359	0,953	10,406	15,266	331,802
1	2	25	18	0,359	0,453	0,328	0,922	1,562	0,781
1	2	50	18	0,718	0,797	0,563	1,812	4,360	1,906
1	2	100	18	1,531	2,906	1,125	4,078	7,375	4,813
1	3	25	18	0,344	0,484	0,297	1,953	1,547	1,172
1	3	50	18	0,703	1,062	0,484	2,469	4,734	2,797
1	3	100	18	1,578	2,672	0,985	5,171	19,422	12,938
2	1	25	18	1,343	0,453	0,422	2,859	2,406	3,125
2	1	50	18	0,594	0,547	0,469	4,954	5,672	9,000
2	1	100	18	1,844	1,156	1,563	8,469	18,719	889,178
2	2	25	18	0,422	0,516	0,297	1,062	1,656	0,844
2	2	50	18	0,750	1,156	0,562	1,750	5,453	2,016
2	2	100	18	1,515	2,703	1,140	5,172	717,317	6,391
2	3	25	18	0,875	0,516	0,344	1,640	1,375	1,265
2	3	50	18	0,672	0,781	0,547	2,812	4,547	3,391
2	3	100	18	1,656	2,828	1,109	343,721	13,391	6,609
3	1	25	18	0,235	0,171	0,219	25,579	2,063	6,938
3	1	50	18	0,437	0,250	0,344	2,063	5,359	35,391
3	1	100	18	0,953	0,641	0,735	5,937	18,141	95,365
3	2	25	18	0,369	1,225	0,369	6,313	13,355	60,365
3	2	50	18	0,441	0,999	1,658	8,324	3,354	6,325
3	2	100	18	0,125	0,547	1,657	6,325	9,324	21,685
3	3	25	18	0,366	0,364	0,845	1,355	10,354	89,871
3	3	50	18	0,584	0,690	0,548	10,635	63,654	236,635
3	3	100	18	0,554	0,558	0,651	5,355	26,354	16,354

Tabla c.18. Valores de los tiempos de resolución de los modelos para el grupo 18.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	19	0,426	0,556	1,355	102,335	16,355	132,546
1	1	50	19	0,542	0,987	2,325	96,355	9,324	111,217
1	1	100	19	0,355	0,790	3,355	290,635	6,349	95,355
1	2	25	19	0,375	0,406	0,297	0,921	1,828	1,000
1	2	50	19	0,688	1,079	0,610	2,141	4,437	1,781
1	2	100	19	1,609	2,547	1,140	4,500	9,812	4,563
1	3	25	19	0,359	0,531	0,297	1,876	1,985	1,266
1	3	50	19	0,703	0,953	0,640	2,422	4,484	2,640
1	3	100	19	1,578	2,860	1,000	4,938	13,766	6,406
2	1	25	19	0,313	0,250	0,281	52,016	2,688	2,984
2	1	50	19	0,531	0,547	0,500	2,718	5,125	6,750
2	1	100	19	1,438	1,375	1,109	6,656	18,860	165,550
2	2	25	19	0,406	0,438	0,329	1,078	1,797	0,859
2	2	50	19	0,719	0,890	0,609	2,062	171,673	1,984
2	2	100	19	1,594	2,969	1,296	5,547	148,579	5,984
2	3	25	19	0,359	0,390	0,344	1,391	1,703	1,250
2	3	50	19	0,703	1,031	0,500	2,171	4,922	2,953
2	3	100	19	1,718	3,047	1,266	5,062	16,250	10,813
3	1	25	19	0,250	0,172	0,219	56,518	17,297	16,922
3	1	50	19	0,453	0,,5468	0,263	66,655	12,654	20,242
3	1	100	19	1,141	0,657	0,765	4,594	20,328	93,578
3	2	25	19	0,266	0,563	0,594	0,564	16,517	49,655
3	2	50	19	0,438	0,375	0,375	1,969	4,313	3,141
3	2	100	19	1,015	0,890	0,781	4,953	13,094	7,093
3	3	25	19	0,250	0,203	0,218	0,797	1,875	1,032
3	3	50	19	0,437	0,344	0,360	1,438	4,453	2,766
3	3	100	19	1,203	0,985	0,718	4,907	12,531	7,750

Tabla c.19. Valores de los tiempos de resolución de los modelos para el grupo 19.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	20	0,328	0,250	0,265	1,375	2,515	148,486
1	1	50	20	0,562	0,547	0,578	3,156	5,375	10,172
1	1	100	20	1,390	1,296	1,141	7,891	2,655	9,655
1	2	25	20	0,406	0,438	0,281	0,859	1,406	0,812
1	2	50	20	0,843	0,875	0,625	1,984	5,031	1,828
1	2	100	20	1,484	3,031	1,328	5,313	205,033	6,282
1	3	25	20	0,359	0,484	0,328	3,015	1,719	1,281
1	3	50	20	0,766	0,922	0,594	2,032	4,656	3,125
1	3	100	20	1,594	2,844	1,015	6,890	14,954	14,062
2	1	25	20	0,328	0,313	0,297	1,797	2,390	3,265
2	1	50	20	0,594	0,594	0,484	3,063	5,454	10,234
2	1	100	20	1,453	1,437	1,078	5,359	20,625	34,078
2	2	25	20	0,703	0,438	0,562	2,453	1,484	0,844
2	2	50	20	0,703	0,922	0,578	2,110	6,062	1,812
2	2	100	20	2,032	2,719	2,359	5,484	13,516	6,375
2	3	25	20	0,391	0,438	0,282	0,969	1,484	1,516
2	3	50	20	0,703	1,000	0,516	1,782	4,547	3,391
2	3	100	20	1,531	2,843	1,516	7,844	16,859	10,641
3	1	25	20	0,265	0,172	0,219	64,987	1,359	1,375
3	1	50	20	0,437	0,265	0,359	7,266	5,734	4,375
3	1	100	20	1,078	0,671	0,719	12,078	16,094	217,158
3	2	25	20	0,297	0,235	0,219	78,684	1,156	1,141
3	2	50	20	0,500	0,422	0,375	2,453	4,406	2,625
3	2	100	20	1,031	1,000	0,750	4,828	12,141	8,531
3	3	25	20	0,266	0,187	0,203	1,172	1,812	1,344
3	3	50	20	0,500	0,360	0,359	2,250	4,844	2,766
3	3	100	20	1,016	0,843	0,766	36,654	5,698	165,545

Tabla c.20. Valores de los tiempos de resolución de los modelos para el grupo 20.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	21	4,578	0,328	0,343	2,765	3,172	3,234
1	1	50	21	0,641	0,547	0,532	3,359	5,829	779,115
1	1	100	21	1,468	0,367	0,570	3,699	10,237	6,149
1	2	25	21	0,375	0,531	0,313	0,937	1,531	0,813
1	2	50	21	0,782	0,890	0,594	2,110	3,859	1,844
1	2	100	21	1,594	3,110	1,406	4,219	179,314	5,250
1	3	25	21	0,344	0,625	0,359	2,812	1,922	1,876
1	3	50	21	1,453	1,187	0,594	1,954	4,921	2,219
1	3	100	21	1,828	2,735	1,109	4,969	13,688	10,000
2	1	25	21	0,328	0,250	0,297	3,390	2,375	3,343
2	1	50	21	0,656	0,563	0,500	1,875	5,610	11,062
2	1	100	21	1,360	1,703	1,063	10,078	29,266	36,407
2	2	25	21	2,312	1,016	0,812	6,156	1,468	1,031
2	2	50	21	0,860	0,890	0,594	99,703	4,500	1,735
2	2	100	21	1,609	2,766	1,765	6,719	9,766	4,953
2	3	25	21	0,375	0,531	0,343	2,875	1,328	1,953
2	3	50	21	0,719	1,109	0,485	1,797	4,906	2,187
2	3	100	21	1,579	3,265	1,109	7,531	17,250	7,281
3	1	25	21	0,297	0,219	0,219	1,125	1,938	1,515
3	1	50	21	0,438	0,213	0,848	10,655	1,355	162,217
3	1	100	21	0,368	0,365	1,355	8,654	5,635	250,660
3	2	25	21	0,235	0,188	0,219	1,203	1,485	1,235
3	2	50	21	0,484	0,344	0,375	2,047	5,078	3,875
3	2	100	21	1,000	0,984	0,750	4,735	15,313	7,812
3	3	25	21	0,250	0,203	0,204	1,062	1,688	1,235
3	3	50	21	0,469	0,360	0,344	2,125	4,656	2,828
3	3	100	21	0,984	1,031	0,735	4,063	14,016	8,047

Tabla c.21. Valores de los tiempos de resolución de los modelos para el grupo 21.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	22	0,313	0,234	0,266	2,359	2,969	87,625
1	1	50	22	0,578	0,515	0,484	1,953	6,140	230,354
1	1	100	22	0,347	0,655	0,559	2,355	8,655	65,354
1	2	25	22	0,375	0,422	0,313	0,735	1,250	0,766
1	2	50	22	0,671	1,110	0,532	2,360	32,485	2,094
1	2	100	22	1,641	2,922	1,235	4,000	10,984	4,922
1	3	25	22	0,328	0,468	0,297	2,547	1,734	1,172
1	3	50	22	0,656	0,859	0,516	1,688	5,187	3,063
1	3	100	22	1,719	2,469	1,062	7,657	15,938	6,703
2	1	25	22	0,296	0,219	0,265	1,093	2,641	3,032
2	1	50	22	0,500	0,531	0,453	2,671	5,172	8,563
2	1	100	22	1,281	1,141	1,016	10,953	38,344	63,562
2	2	25	22	0,355	1,343	0,989	20,673	10,635	13,354
2	2	50	22	0,337	1,355	0,355	6,655	26,342	26,654
2	2	100	22	0,255	0,697	0,687	6,355	33,210	165,651
2	3	25	22	0,137	0,355	0,355	3,655	65,355	135,654
2	3	50	22	0,222	1,659	0,999	8,355	16,324	26,651
2	3	100	22	0,693	1,368	0,655	5,355	13,355	36,541
3	1	25	22	0,466	1,784	0,664	2,355	6,325	26,651
3	1	50	22	0,567	1,844	0,648	1,370	3,322	16,651
3	1	100	22	0,654	1,517	0,635	6,322	6,355	13,654
3	2	25	22	0,250	0,187	0,203	1,250	1,515	169,654
3	2	50	22	0,437	0,344	0,359	2,219	5,047	3,219
3	2	100	22	1,015	0,906	0,812	54,685	3,255	236,256
3	3	25	22	0,250	0,204	0,203	132,355	1,438	1,469
3	3	50	22	0,453	0,375	0,344	3,109	4,328	3,344
3	3	100	22	0,922	0,922	0,734	4,984	13,078	7,656

Tabla c.22. Valores de los tiempos de resolución de los modelos para el grupo 22.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	23	0,313	0,281	0,266	0,953	2,281	83,516
1	1	50	23	0,641	0,531	0,531	2,390	5,344	46,355
1	1	100	23	0,367	0,965	0,256	3,000	8,984	9,517
1	2	25	23	0,360	0,469	0,328	0,906	1,641	1,391
1	2	50	23	0,734	1,219	0,563	2,985	4,891	1,750
1	2	100	23	1,501	3,000	1,125	4,235	11,219	5,157
1	3	25	23	0,328	0,422	0,312	2,125	1,687	1,219
1	3	50	23	0,656	0,953	0,531	1,813	4,141	2,985
1	3	100	23	1,719	2,313	1,047	6,312	15,469	7,781
2	1	25	23	0,328	0,234	0,297	1,438	2,672	3,141
2	1	50	23	0,516	0,500	0,437	4,938	5,500	9,953
2	1	100	23	1,485	1,359	1,031	7,750	22,860	39,655
2	2	25	23	0,328	0,485	0,328	1,110	1,704	0,859
2	2	50	23	0,766	1,047	0,578	1,828	4,609	2,078
2	2	100	23	1,687	3,016	1,187	4,500	42,322	8,355
2	3	25	23	0,328	0,453	0,266	2,391	1,719	1,281
2	3	50	23	0,703	0,891	0,531	1,625	5,187	2,094
2	3	100	23	1,547	2,531	1,047	6,641	14,235	10,454
3	1	25	23	0,250	0,172	0,234	50,540	11,719	11,375
3	1	50	23	0,453	0,297	0,360	50,565	73,720	73,813
3	1	100	23	0,907	0,593	0,766	5,438	14,609	139,408
3	2	25	23	0,250	0,203	0,218	87,872	1,047	1,046
3	2	50	23	0,453	0,453	0,375	2,016	4,594	2,766
3	2	100	23	0,969	1,015	0,828	7,454	25,922	11,390
3	3	25	23	0,891	0,813	1,391	2,985	2,047	1,437
3	3	50	23	0,485	0,359	0,360	2,234	4,578	3,172
3	3	100	23	1,610	1,421	2,078	6,797	12,922	8,109

Tabla c.23. Valores de los tiempos de resolución de los modelos para el grupo 23.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	24	0,313	0,219	0,266	0,687	2,406	263,925
1	1	50	24	0,531	0,531	0,516	4,891	5,438	9,906
1	1	100	24	0,487	1,211	0,984	56,655	98,651	100,367
1	2	25	24	0,390	0,437	0,297	1,109	1,516	0,766
1	2	50	24	0,719	0,812	0,594	1,890	4,922	1,906
1	2	100	24	1,594	3,000	1,141	5,218	8,500	5,047
1	3	25	24	0,359	0,438	0,282	1,718	1,984	1,328
1	3	50	24	0,656	0,860	0,547	3,281	3,922	3,047
1	3	100	24	2,734	0,355	0,687	3,324	6,352	66,559
2	1	25	24	0,422	0,266	0,265	0,953	2,828	3,016
2	1	50	24	0,547	0,563	0,469	2,860	5,297	9,063
2	1	100	24	1,343	1,328	1,188	9,610	17,969	26,797
2	2	25	24	0,391	0,391	0,297	1,157	1,735	1,062
2	2	50	24	0,750	1,016	0,578	3,219	5,140	1,782
2	2	100	24	1,610	2,750	1,218	6,735	24,438	4,953
2	3	25	24	0,360	0,453	0,297	2,406	1,703	1,485
2	3	50	24	0,672	0,938	0,531	2,454	4,437	2,953
2	3	100	24	1,656	2,609	0,938	23,266	18,703	8,688
3	1	25	24	0,344	0,344	1,796	8,126	5,219	1,672
3	1	50	24	0,547	0,938	1,953	5,126	41,563	41,626
3	1	100	24	1,000	0,640	0,781	5,328	19,157	119,721
3	2	25	24	0,265	0,188	0,218	6,355	1,156	1,110
3	2	50	24	0,453	0,390	0,454	3,359	5,969	16,891
3	2	100	24	1,047	1,437	0,813	5,344	15,422	12,235
3	3	25	24	0,500	0,218	0,235	8,125	1,344	1,265
3	3	50	24	0,469	0,375	0,375	1,703	4,469	3,422
3	3	100	24	1,062	0,907	0,703	3,860	12,297	7,250

Tabla c.24. Valores de los tiempos de resolución de los modelos para el grupo 24.

EF.	DEM.	E	GRUPO	Tiempo de ejecución [segundos]					
				M1	M2	R1	R2	R3	R4
1	1	25	25	0,313	0,250	0,281	1,234	2,422	90,548
1	1	50	25	0,625	0,531	0,500	4,781	5,375	457,475
1	1	100	25	0,354	0,500	0,369	6,215	20,244	79,255
1	2	25	25	0,359	0,453	0,312	1,422	1,500	0,844
1	2	50	25	0,765	1,078	0,609	2,985	5,375	2,062
1	2	100	25	1,734	3,000	1,219	5,093	44,422	5,688
1	3	25	25	0,360	0,562	0,343	1,719	1,672	1,234
1	3	50	25	0,719	0,906	0,532	111,377	5,094	2,750
1	3	100	25	1,891	3,031	1,109	5,250	21,172	8,125
2	1	25	25	0,812	0,953	1,719	78,641	2,578	2,547
2	1	50	25	0,625	0,563	0,547	3,547	5,375	10,109
2	1	100	25	1,266	1,422	1,312	7,610	18,000	22,357
2	2	25	25	0,625	0,406	0,282	0,906	1,609	0,891
2	2	50	25	0,672	0,812	0,578	2,016	92,361	1,953
2	2	100	25	1,593	2,688	1,187	4,000	74,220	6,156
2	3	25	25	0,375	0,641	0,297	1,500	1,672	1,578
2	3	50	25	0,734	1,188	0,547	1,828	4,672	3,250
2	3	100	25	1,641	3,250	0,937	5,375	14,829	7,282
3	1	25	25	0,234	0,172	0,204	3,226	15,344	14,828
3	1	50	25	0,437	0,281	0,359	6,255	10,354	10,370
3	1	100	25	0,875	0,531	0,734	5,891	14,156	11,501
3	2	25	25	0,250	0,188	0,219	1,610	1,719	1,719
3	2	50	25	0,469	0,359	0,376	2,031	4,969	3,688
3	2	100	25	1,047	0,922	0,765	5,047	12,235	6,922
3	3	25	25	0,234	0,188	0,204	7,456	1,235	1,344
3	3	50	25	0,437	0,344	0,375	3,109	5,391	3,422
3	3	100	25	1,094	1,015	0,984	6,322	14,987	6,655

Tabla c.25. Valores de los tiempos de resolución de los modelos para el grupo 25.