Gueret's Models

Tony Hürlimann
info@matmod.ch

March 13, 2025
(First version: Dec 01, 2001)

Abstract

This paper contains all 65 models from the French book of Guéret Christelle, Prins Christian and Sevaux Marc (see [2]). The book has been translated to English by Dash Optimization and is available at [4]. It can be downloaded from the Internet free of charge. The last chapter 14 of [2] has not been translated, so I give the original French version of the models. Some models that I used in my master courses have additional comments and explanation, but most of them only contain the bare bone LPL implementation that can also be executed directly through the Internet. The reader needs the book for more explanation of the models.

Contents

1 Introduction

The text contains the LPL source code from all models from the French book [2]. Most models have been translated to English and are explained in the free book [4] – also downloadable at XPress. Each model can be executed directly through the Internet by clicking the red link at the top title of each model. For the model explanation the reader will need the book itself.

2 Animal food production (guer04-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 4.2. English translation in [4] chap 6.2.

Listing 1: The Complete Model implemented in LPL [5]

```
model guer04_2 "Animal food production";
  set
    n  := [1..2] "Foods";
    m  := [1..3] "Raw material";
    o  := [1..4] "Operations";
    b  := [1..3] "Nutrients";
  parameter
    BN{b} := [9.5 2 6]              "Required contents";
    P{m,b}:= [13.6 7.1 7 , 4.1 2.4 3.7 , 5 0.3 25]
                          "Nut. contents in percent";
    QMax{m} := [11900 23500 750] "Availability";
    Pr{m} := [0.8 1 0.75]         "Price";
    C{o}  := [1.5 0.3 2.5 1]      "Operation cost";
    D{n}  := [9000 12000]         "Demand";
  variable
    Q{n}   "Quantity produced";
    X{n,m} "Quantity raw material";
  constraint
    PRO{n}: sum{m} X = Q;
    ATL{n,b|b<#b}: sum{m} P*X >= BN*Q;
    MIN{n}: sum{m} P[m,#b]*X <= BN[#b]*Q;
    MAX{m}: sum{n} X<=QMax;
    DEM{n}: Q >= D;
  minimize cost: sum{n,m} Pr*X
       + sum{n,m|m<#m} C[1]*X + sum{n,m} C[2]*X
       + sum{m} C[3]*X[1,m] + sum{m} C[4]*X[2,m];
  Writep(cost,Q,X);
end
```

3 Production of alloys (guer04-3)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 4.3. English translation in [4] chap 6.1.

Listing 2: The Complete Model implemented in LPL [5]

```
model guer04_3 "Production of alloys";
set
  m  := [1..7] "Raw material";
  e  := [1..3] "Chemical elements";
parameter
  C{m} := [1.2, 1.5, 0.9, 1.3, 1.45, 1.2, 1] "Cost of m";
  Qmax{m} := [40 30 60 50 20 30 25]       "Availability";
  P{m,e} "Percent content" :=
    [2.5 0 1.3 , 3   0 0.8 , 0 0.3   0 , 0  90   0 ,
     0  96   4 , 0 0.4 1.2 , 0 0.6   0];
  PMin{e} := [2, 0.4, 1.2]  "Min. content in product";
  PMax{e} := [3, 0.6, 1.65] "Max. content in product";
  D := 5000                 "Demand";
variable
  X{m} [0..100*Qmax] "Quantity in end product";
  Q              "Quantity produced";
constraint
  QteFabriquee: sum{m} X = Q;
  Demande: Q >= D;
  Pourcent{e}: PMin*Q <= sum{m} P*X <= PMax*Q;
minimize Couts: sum{m} C*X;
Writep(Couts,X,Q);
end
```

4 Refinery (guer04-4)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 4.4. English translation in [4] chap 6.3.

Listing 3: The Complete Model implemented in LPL [5]

```
model guer04_4 "Refinery";
parameter
  B1:=250000 "Crude 1 in stock";
  B2:=500000 "Crude 2 in stock";
variable
  -- end products
  B    "Butane";
  C    "Petrol";
  G    "Diesel oil";
  F    "Heating oil";
  -- Intermediary products
  BC; EC; DC; GG; DG; GF; DF;
constraint
  MaxButane  : B+BC     <= 0.03*B1+0.05*B2;
  MaxEssence : EC       <= 0.15*B1+0.20*B2;
  MaxGazole  : GG+GF    <= 0.40*B1+0.35*B2;
  MaxDistill : DC+DG+DF <= 0.15*B1+0.10*B2;
  CompoCarb  : C = EC+BC+DC;
  CompoGaz   : G = GG+DG;
  CompoFuel  : F = GF+DF;
  Octane     : 100*EC+120*BC+ 74*DC >= 94*C;
  Vapeur     : 2.6*EC+ 60*BC+4.1*DC <= 12.7*C;
  Volatilite :   3*EC+105*BC+ 12*DC >= 17*C;
  Soufre     : 1.2*DG+0.3*GG <= 0.5*G;
  Reformage  : EC <= 30000;
  Desulfur   : GG+GF <= 40000;
  Craquage   : DC+DG+DF <= 50000;
  -- Demand
  DemButane  : B >= 20000;
  DemCarb    : C >= 40000;
  DemGaz     : G >= 35000;
  demFuel    : F >= 50000;
minimize Cost: 250*EC+ 450*(GG+GF)+ 350*(DC+DG+DF);
Writep(Cost);
end
```

5 Cane Sugar Production (guer04-5)

—- Run LPL Code , HTML Document –

Problem: "The harvest of cane sugar in Australia is highly mechanized. The sugar cane is immediately transported to a sugarhouse in wagons that run on a network of small rail tracks. The sugar content of a wagonload depends on the field it has been harvested from and on the maturity of the sugar cane. Once harvested, the sugar content decreases rapidly through fermentation and the wagonload will entirely lose its value after a certain time. At this moment, eleven wagons loaded with the same quantity have arrived at the sugarhouse. They have been examined to find out the hourly loss and the remaining life span (in hours) of every wagon." (Cited from [1].

A different formulation due to [1] is given in model guer04-5a

The problem is from [2] chap 4.5. English translation in [4] chap 6.4.

Listing 4: The Complete Model implemented in LPL [5]

```
model guer04_5 "Cane Sugar Production";
parameter
  N   := 3 "Number of production lines";
  DUR := 2 "Duration";
set
  w  := [1..11] "Lots (waggons)";
  c  := 1..Ceil(#w/N) "Nombres of time slots";
parameter
  Perte{w} := [43,26,37,28,13,54,62,49,19,28,30] "Loss";
  Vie{w}   := [8,8,2,8,4,8,8,8,6,8,8]       "Life span";
binary variable X{w,c} "Assign every waggon to a time slot";
constraint
  Affect{w}: sum{c} X = 1;
  WaggonsPerSlot{c}: sum{w} X <= N;
  Limite{w}: sum{c} c*X <= Vie/DUR;
minimize TotalLoss: sum{w,c} c*DUR*Perte*X;
parameter slot{w}:=argmax{c} X;
Write{w}('Waggon␣%2s␣Time␣slot%2d␣␣␣Loss=%3d\n', w, slot, sum{c} c*DUR*
   Perte*X );
Write('Total␣loss:␣%d\n', TotalLoss);
end
```

Questions

1. What is the total loss if the life span of each wagon is infinite?
2. What is the loss on each wagon (give the expression in LPL).
3. Suppose that it is possible to accelerate the traitment. How long should the duration be in order to push the overall loss under 1000 kg? How does the overall loss decrease with the acceleration of the traitment?
4. How does the number of lignes influence the total loss?
5. Create and run a problem with 100 wagons.

6 Cane Sugar Production (guer04-5a)

—- Run LPL Code , HTML Document –

Problem: This is a different formulation of the model guer04-5. It is more appropiate for a constraint programming solver like Kalis. It shows the *global cardinality* constraint.

The problem is from [2] chap 4.5. English translation in [4] chap 6.4.

Listing 5: The Complete Model implemented in LPL [5]

```
model guer04_5a "Cane Sugar Production";
parameter
  N   := 3 "Number of production lines";
  DUR := 2 "Duration";
set
  w  := [1..11] "Lots (waggons)";
  c  := 1..Ceil(#w/N) "Nombres of time slots";
parameter
  Perte{w} := [43,26,37,28,13,54,62,49,19,28,30] "Loss";
  Vie{w}   := [8,8,2,8,4,8,8,8,6,8,8]       "Life span";
  COST{w,c}:=c*DUR*Perte;
integer variable
      slot{w} [1..#c] "waggon processed in time slot";
      loss{w} [0..Perte*Vie] "loss of waggon w";
constraint
  WaggonsParSlot{c}: sum{w} (slot=c) <= N;     // LPL syntax
  --WaggonsPerSlot{c}: Cardinality(slot,c,N,'<='); --Kalis syntax
  --WaggonsPerSlot{c}: Occurrence(slot,c) <= N; -- Mosel syntax
  Loss{w}: Element(slot, {c}COST, loss);
  --Loss{w}: loss = COST[w,slot]; //not yet implemented
minimize TotalLoss: sum{w} loss;
Write{w}('Waggon␣%2s␣Time␣slot%2d␣␣␣Loss=%3d\n', w, slot, loss);
Write('Total␣loss:␣%d\n', TotalLoss);
end
```

7 Opencast Mining (guer04-6)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 4.6. English translation in [4] chap 6.5.

Listing 6: The Complete Model implemented in LPL [5]

```
model guer04_6 "Opencast Mining";
  set
    i,j := [1..18] "Number of blocks";
    A{i,j}  "Block j above i" :=
      [(15,9)(15,10)(15,11)(16,10)(16,11)(16,12)(17,11)
      (17,12)(17,13)(18,12)(18,13)(18,14)(9,1)(9,2)(9,3)
      (10,2)(10,3)(10,4)(11,3)(11,4)(11,5)(12,4)(12,5)
      (12,6)(13,5)(13,6)(13,7)(14,6)(14,7)(14,8)];
  parameter
    C{i} "Extraction cost" :=
      [100, 100, 100, 100, 100, 100, 100, 100,1000,
       200, 200, 200, 200,1000,1000,1000, 300,1000];
    V{i} "Value of bloc" :=
      [200,   0,   0,   0,   0,   0, 300,   0,   0,
       500,   0, 200,   0,   0,   0,   0,1000,1200];
  binary variable X{i} "Extract?";
  constraint Degage{A[i,j]}: X[i] -> X[j];
  maximize Profit: sum{i} (V-C)*X;
  Draw.Scale(50,50);
  Draw.DefFont('Verdana',10);
  {i|i<=8} Draw.Rect(i&'',i,0,1,1,if(X,3,1),0);
  {i|i>8 and i<15} Draw.Rect(i&'',i-7,1,1,1,if(X,3,1),0);
  {i|i>=15} Draw.Rect(i&'',i-12,2,1,1,if(X,3,1),0);
  Draw.Text('Extract:'&sum{i}X&'␣blocks',0,2.3);
  Draw.Text('Total␣Profit:'&Profit,0,2.6);
end
```

8 Production of Electricity (guer04-7)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 4.7. English translation in [4] chap 6.6.

Listing 7: The Complete Model implemented in LPL [5]

```
model guer04_7 "Production of Electricity";
set
  p := [1..5] "Time periods";
  t := [1..3] "Generator types";
parameter
  Larg{p} := [6 3 6 3 6]   "Period length";
  Dema{p} := [15000,30000,25000,40000,27000] "Demand";
  PMin{t} := [ 850,  1250,  1500] "Minimal power";
  PMax{t} := [2000,  1750,  4000] "Maximal power";
  CDem{t} := [2000,  1000,   500] "Starting cost";
  CMin{t} := [1000,  2600,  3000] "Cost at min power";
  CSup{t} := [ 2, 1.3, 3] "Additional cost above min";
  NDis{t} := [ 12, 10,  5] "Number of generators";
variable
  NDem{t,p} "Starting generators";
  PSup{t,p} "Supp. power";
  integer NFon{t,p} [0..NDis] "Generator working";
constraint
  R{p,t} : NDem >= NFon - if(p=1,NFon[t,#p],NFon[t,p-1]);
  Lim{p,t}: PSup <= (PMax-PMin)*NFon;
  SatisDeman{p}: sum{t} PMin*NFon + sum{t} PSup >= Dema;
  Reserve{p}: sum{t} PMax*NFon >= 1.15*Dema;
minimize
  Cost:sum{p,t}(CDem*NDem+Larg*CMin*NFon+Larg*CSup*PSup);
Writep(Cost);
end
```

9 Stadium Construction (guer05-2)

—- Run LPL Code , HTML Document –

Problem: A municipality wants to construct a small stadium. A construction company is awarded the contract an wants to complete the construction in the shortest time possible. All tasks are listed and their duration (in weeks) is estimated and the the order of precedence between them is established, that is, certain tasks must be completetd before others can begin. All these data are collected in a table (see Table 1.

What is the earliest possible date of completing the construction?

The model is from [2] chapter 5.2. and [4] chapter 7.1.

Task	Description	Duration	Predecessors	max reduct.	add. cost
1	Installing the construction site	2	none	0	
2	Terracing	16	1	3	30
3	Constructing the foundations	9	2	1	26
4	Access roads and other networks	8	2	2	12
5	Erecting the basement	10	3	2	17
6	Main floor	6	4,5	1	15
7	Dividing up the changing rooms	2	4	1	8
8	Electrifying the terraces	2	6	0	
9	Constructing the roof	9	4,6	2	42
10	Lighting of the stadium	5	4	1	21
11	Installing the terraces	3	6	1	18
12	Sealing the roof	2	9	0	
13	Finishing the changing rooms	1	7	0	
14	Constructing the ticket office	7	2	2	22
15	Secondary access roads	4	4,14	2	12
16	Means of signalling	3	8,11,14	1	6
17	Lawn and sport accessories	9	12	3	16
18	Handing over the building	1	17	0	

Table 1: Data for the Stadium Construction

Modeling Steps: We introduce a set for the tasks $i, j \in I$. All we need is the duration time of a task p_i and the precedence list of task tuples $Prec_{i,j}$. The tuple entry $Prec_{i,j}$ is true if and only if task i must be completed before task j. The variables are the (earliest) starting time t_i of a task i. The objective is to minimize the latest starting time.

The model is as follows:

$$\begin{array}{ll} \min & \max\limits_{i \in I} (t_i + p_i) \\ \text{subject to} & t_j - t_i \geq p_i, \quad \forall (i,j) \in Prec \\ & t_i \geq 0, \quad \forall i \in I \end{array}$$

Listing 8: The Main Model implemented in LPL [5]

```
model guer05_2 "Stadium Construction";
  set i,j        "tasks";
      Prec{i,j}  "precedences";
  parameter p{i} "duration";
            r{i} "max. reduction";
            c{i} "add. cost per reducted week";
            T    "starting time of last task";
  variable t{i}  "start time of task i";
  constraint Precedence{Prec[i,j]}: t[j] - t[i] >= p[i];
  minimize lasttaskBegin: max{i} (t+p);
```

```
  T:=lasttaskBegin;
  Writep(T,t);
  Draw.Scale(14,20);
  for{i} do
    Draw.Rect(t,i,p,1,i%12+2); Draw.Text((t+p)&'',t+p,i+.5,12);
    Draw.Text(i&'',t+0.1,i+.6,12);
    Draw.Line(t,i+1,t,#i+2); Draw.Text(t&'',t+.1,#i+1.8,12);
  end
  Draw.Rect(0,#i+2,t[#i],.1,0);
  --CostModel;
  EarliestLatest;
  model CostModel;
    parameter P:=30  "Prime for reduced time";
    variable
      A  "reduced weeks";
      s{i}  [0..r] "reduced week per task";
      TT "start time of last task" ;
    constraint
      Precedence{Prec[i,j]}: t[j] - t[i] + s[i] >= p[i];
      DureeTot: TT + A = T;
      LastTask{i}: TT>=(t+p-s);
    maximize Gain: P*A - sum{i} c*s - 0.00001*(sum{i} t);
    Writep(Gain,t,A,s);
    parameter Y:=#i+2;
    for{i} do
      Draw.Rect(t,i+Y,p-s,1,i%12+2);
      Draw.Text(i&'',t+0.1,i+.6+Y,12);
      Draw.Line(t,i+1+Y,t,#i+2+Y); Draw.Text(t&'',t+.1,#i+1.8+Y,12);
    end
    Draw.Rect(0,#i+2+Y,t[#i],.1,0);
  end
  model EarliestLatest;
    constraint Precedence{Prec[i,j]}: t[j] - t[i] >= p[i];
    minimize obj: max{i} (t+p) + 0.00001*(sum{i} t);
    Draw.Save('temp');
    Draw.Scale(14,20);
    for{i} do
      Draw.Rect(t,i,p,1,i%12+2); Draw.Text((t+p)&'',t+p,i+.5,12);
      Draw.Text(i&'',t+0.1,i+.6,12);
      Draw.Line(t,i+1,t,#i+2); Draw.Text(t&'',t+.1,#i+1.8,12);
    end
    Draw.Rect(0,#i+2,t[#i],.1,0);
    parameter Earliest{i}:=t;
    minimize obj1: max{i} (t+p) - 0.00001*(sum{i} t);
    for{i} do
      if t>Earliest then Draw.Rect(Earliest,i+.3,t-Earliest,.4,i
          %12+2,-2,-2,.2); end
      Draw.Rect(t,i,p,1,i%12+2); Draw.Text((t+p)&'',t+p,i+.5,12);
      Draw.Text(i&'',t+0.1,i+.6,12);
      Draw.Line(t,i+1,t,#i+2); Draw.Text(t&'',t+.1,#i+1.8,12);
    end
    Draw.Rect(0,#i+2,t[#i],.1,0);
    parameter Lastest{i}:=t;
  end
end
```

Further Comments: A comment on the Read statement: The first read staemetn reads the data

from an Excel sheet (the statement is outcommented). The second Read reads the same data from the same format in a tab separated text file. Note how similar these statements are. In the text-form we use `'%1'` to indicate that the first line must be skipped, and `'\t\n'` indicates that the only token separators are tabs and line feeds.

Listing 9: The Data Model

```
model data;
  parameter M; string N{i}; PR{i};
  --Read{i}('guer05-2.xls,[Sheet1$A2:F19]',i,N,p,PR,r,c);
  Read{i}('guer05-2.txt,;1,\t\n',i,N,p,PR,r,c);
  {i,j} (M:=Strfloat(PR[i],',',j), if(M,Prec[M,i]:=1));
end
```

Questions

1. What is the earliest, what is the latest time that each task can begin?
2. Draw a Gantt diagram to visualiser the solution.
3. Suppose that the duration of each task could be reduced by one unit. What is now the solution?
4. Suppose that the task number 4 can only begin at the 32 week. What are the implication of the overall plan?
5. Suppose the municipality would like to terminate the construction earlier than projected by the construction company and is ready to pay a bonus of 30 unit of money for each reduced week. The company has additional costs for each task c_i per reduced week (last column in Table 1) and each task can only be reduced a maximal number of weeks, given by r_i (second to the last column in Table 1. How would the construction company proceed?
6. Create and run a project with 100 tasks.

Answers

1. Replace the objective by the following code:

   ```
   minimize obj: max{i} (t+p) + 0.00001*(sum{i} t);
   parameter Earliest{i}:=t;
   minimize obj1: max{i} (t+p) - 0.00001*(sum{i} t);
   parameter Lastest{i}:=t;
   ```

   The parameter `Earliest` gives the eariest possible starting time, and `Lastest` gives the latest possible time. The Submodel `EarliestLatest` shows the results.

2. See answer 1
3. Repace p_i by $p_i - 1$ the run again. The completion time is 55 weeks.
4. Add the constraint `constraint C4: t[4]>=32;` and run the model again.
5. Certainly, the company would try to maximize the additional bonuses. The additional requirements are formulated in the model `CostModel`.

10 Flow-shop Scheduling (guer05-3)

—- Run LPL Code , HTML Document –

Problem: The model is from [2], Chap 5.3. and [4] chapter 7.2.

Listing 10: The Complete Model implemented in LPL [5]

```
model guer05_3 "Flow-shop Scheduling";
  set m,m1   := [1..3] "Machines";
      j,k,k1 := [1..6] "Job j / Rank k,k1";
  parameter
    p{m,j}:=[3 6 3 5 5 7, 5 4 2 4 4 5, 5 2 4 6 3 6] "processing time";
  variable
    E{m,k|k<#k and m>1} "Idle time on machine m between job of rang k
        and k+1";
    A{m,k|m<#m and k>1} "Waiting time between machine m and m+1 of job
        in rang k";
    binary X{j,k}       "1 if job j is in rank k";
  constraint
    C{k}: sum{j} X = 1  "Exactly un job in each rank";
    B{j}: sum{k} X = 1  "Exactly un rank for each job";
    D{m,k|m<#m and k<#k}:
       E[m,k] + sum{j} p[m,j]*X[j,k+1] + A[m,k+1] =
       A[m,k] + sum{j} p[m+1,j]*X[j,k] + E[m+1,k]
      "Relations between end of job at rang k on mach m and begin of
         the following job on mach m+1";
  minimize W: sum{m,j|m<#m} p*X[j,1]+sum{k|k<#k} E[#m,k]
    "Total empty time on the last machine";
  integer parameter
    x{j}:=argmax{k|X} k;
    T{m,k} := sum{m1,j|m1<m} p[m1,j]*X[j,1] +
     sum{k1,j|k1<k} p[m,j]*X[j,k1] + sum{k1|k1<k} E[m,k1]
      "begin on mach m of job with rank k";
    t{m,j}:=T[m,x];
  Write('Total wating time: %4d\n', W);
  Write('Job permutation  : %3d\n', {j}x);
  ---draw the solution
  Draw.Scale(20,25);
  for{m,j} do Draw.Rect(j&'',t,m,p,1,j+3); end;
  set i:=[1..40];
  for{i} do Draw.Line(i,#m+1,i,#m+1.3); end
  for{i} do Draw.Text(i&'',i,#m+1.5,12); end
end
```

Questions

1. Visualize the solution with a Gantt diagram.
2. How could we force to process job 1 just before job 3 ? Add the condition for that. What is the total duration now?
3. Suppose that we must execute the job 6 as the first job. How could this condition be added to the model? What is the consequence on the total processing time?
4. Create and run a problem with 50 jobs.

11 Job Shop: non-generic version (guer05-4a)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 5.4. English translation in [4] chap 7.3.

Listing 11: The Complete Model implemented in LPL [5]

```
model guer05_4a "Job Shop: non-generic version";
set
  n  := [1..3]  "Jobs";
  m  := [1..3]  "Machines";
parameter
  P{m,n}:=[45 20  0   0 10 17  10 34 28] "Process time";
  M := 1000  "Big-M";
variable
  T{m,n}  "Begin of task";
  TEnd    "Make span";
  binary Y{1..5} "Disjuctions";
constraint
  FinJ1     : TEnd >= T[3,1]+P[3,1];
  FinJ2     : TEnd >= T[3,2]+P[3,2];
  FinJ3     : TEnd >= T[3,3]+P[3,3];
  -- Conjonctions : task of the same job on the machines
  J1_M3M1   : T[1,1]+P[1,1] <= T[3,1];
  J2_M1M3   : T[2,2]+P[2,2] <= T[1,2];
  J2_M3M2   : T[1,2]+P[1,2] <= T[3,2];
  J3_M2M3   : T[2,3]+P[2,3] <= T[3,3];
  -- Disjonctions : precedence on the same machine
  M1_J1J2_1 : T[1,1]+P[1,1] <= T[1,2]+M-M*Y[1];
  M1_J1J2_2 : T[1,2]+P[1,2] <= T[1,1]+M*Y[1];
  M2_J2J3_1 : T[2,2]+P[2,2] <= T[2,3]+M-M*Y[2];
  M2_J2J3_2 : T[2,3]+P[2,3] <= T[2,2]+M*Y[2];
  M3_J1J2_1 : T[3,1]+P[3,1] <= T[3,2]+M-M*Y[3];
  M3_J1J2_2 : T[3,2]+P[3,2] <= T[3,1]+M*Y[3];
  M3_J1J3_1 : T[3,1]+P[3,1] <= T[3,3]+M-M*Y[4];
  M3_J1J3_2 : T[3,3]+P[3,3] <= T[3,1]+M*Y[4];
  M3_J2J3_1 : T[3,2]+P[3,2] <= T[3,3]+M-M*Y[5];
  M3_J2J3_2 : T[3,3]+P[3,3] <= T[3,2 ]+M*Y[5];
minimize FinOrdo: TEnd;
Writep(TEnd,T,Y);
end
```

12 Job Shop: generic version (guer05-4b)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 5.4. English translation in [4] chap 7.3.

Listing 12: The Complete Model implemented in LPL [5]

```
model guer05_4b "Job Shop: generic version";
set
  m,n := [1..9]  "Tasks";
  U{m,n} := [1 7, 5 2, 2 8, 6 9] "m before n";
  D{m,n} := [1 2, 5 6, 7 8, 7 9, 8 9] "m before n";
parameter
  P{m} := [45,20,0,0,10,17,10,34,28] "Processing time";
  M := sum{m} P  "Big-M";
variable
  T{m}          "Starting time";
  TEnd [0..M]   "Makespan";
  binary Y{D[m,n]|m<#m and n>m} "Disjonctions";
constraint
  ToutesTerminees{m}: TEnd >= T+P;
  Conj{U[m,n]}: T[m]+P[m] <= T[n];
  Dis1{D[m,n]|m<#m and n>m}: T[m]+P[m] <= T[n]+M*Y;
  Dis2{D[m,n]|m<#m and n>m}: T[n]+P[n] <= T[m]+M-M*Y;
minimize FinOrdo: TEnd;
Writep(FinOrdo,T);
end
```

13 Sequencing jobs on a bottleneck machine (guer05-5)

—- Run LPL Code , HTML Document –

Problem: A set of jobs is to be processed on a single machine. The execution of tasks is non-preemptive (that is, an operation may not be interrupted before its completion). For every job *i* its release date, duration, and due date are given. Implement a model with three different objectifs: minimize the total processing time, the average processing time, and the total tardiness.

For a different model do be solved by constraint programming see guer05-5a.

The problem is from [2] chap 5.5. English translation in [4] chap 7.4.

Listing 13: The Complete Model implemented in LPL [5]

```
model guer05_5 "Sequencing jobs on a bottleneck machine";
set m,n := [1..7] "Job / Rank";
parameter
  r{m}:=[2 5 4 0 0 8 9] "Earlist time";
  p{m}:=[5 6 8 4 2 4 2] "Processing time";
  d{m}:=[10 21 15 10 5 15 22] "Due date";
variable
  t{n}  "Starting time";
  c{n}  "End time";
  T{n}  "Tardiness";
  binary u{m,n} "Job sequence";
constraint
  A{n}: sum{m} u = 1;
  B{m}: sum{n} u = 1;
  Sequence{n|n<#n}: t[n+1] >= t[n] + sum{m} p*u;
  Debut{n}: t >= sum{m} r*u;
  Completion{n}: c = t + sum{m} p*u;
  Retard{n}: T >= c - sum{m} d*u;
minimize Cmax: c[#n];
  integer parameter  M{n}:=argmax{m} u;
  Draw.Scale(20,20);
  Draw.Text('Minimize␣makespan:␣'&Cmax,1,0.7);
  {n} Draw.Rect(M&'',t,1,c-t,1,M+1);
minimize SommeC: sum{n} c;
  M{n}:=argmax{m} u;
  Draw.Text('Minimize␣the␣average␣processing␣time:␣'&SommeC,1,2.7);
  {n} Draw.Rect(M&'',t,3,c-t,1,M+1);
minimize SommeT: sum{n} T;
  M{n}:=argmax{m} u;
  Draw.Text('Minimize␣total␣tardiness:␣'&SommeT,1,4.7);
  {n} Draw.Rect(M&'',t,5,c-t,1,M+1);
  {n} Draw.Rect(r,6.1+n*0.3,d-r,0.3,n+1);
  {n} Draw.Rect(d[M],6.1+M*0.3,T,0.3,0);
Writep(SommeC,Cmax,t);
end
```

Solution: The solutions of all three ojective function is given in Figure 1.

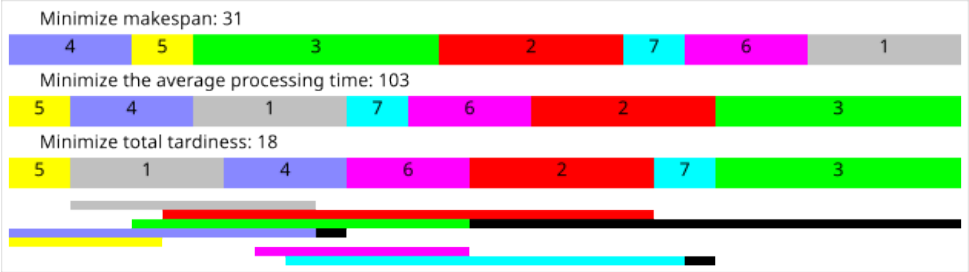

Figure 1: Optimal Sequencings

14 Paint production (guer05-6)

—- Run LPL Code , HTML Document –

Problem: As a part of its weekly production a paint company produces five batches of paints, always the same, for some big clients who have a stable demand. Every paint batch is produced in a single production process, all in the same blender that needs to be cleaned between every two batches. The durations of blending paint are given in minutes. The cleaning times depend on the colors and the paint types. For example, a long cleaning period is required if an oil-based paint is produced after a water-based paint, or to produce white paint after a dark color. In which order the batches should be produced in order to minimize the overall production time. (See [1]).

A different implementation is given in model guer05-6a. This implementation is for a constraint programming solver.

The problem is from [2] chap 5.6. English translation in [4] chap 7.5.

Modeling Steps: Note that this problem is an TSP instance and the implementation is the same as in model tsp-1.

Listing 14: The Complete Model implemented in LPL [5]

```
model guer05_6 "Paint production";
set i,j := [1..5]  "batches/sequence";
parameter
  d{i}   := [40,35,45,32,50]  "Processing time";
  c{i,j}  "Setup time (cleaning time)" :=
   [ 0,11, 7,13,11 , 5, 0,13,15,15
    13,15, 0,23,11 , 9,13, 5, 0, 3 , 3, 7, 7, 7, 0];
variable Y{i};
binary variable X{i,j|i<>j};
constraint
  A{i}: sum{j} X = 1;
  B{j}: sum{i} X = 1;
  C{i,j|i<>j and j>1}: Y[j] >= Y[i]+1 - #i*(1-X);
minimize Cost: sum{i,j} (d+c)*X;
parameter x{i}; n:=1; {i} (n:=argmax{j} X[n,j], x[i]:=n);
Write('Cost: %d\n', Cost);
Write('Order of batches: 1%s\n', {i}Format(' --> %d', x));
end
```

15 Paint production (guer05-6a)

—- Run LPL Code , HTML Document –

Problem: This is the same as model guer05-6. This implementation is for a constraint programming solver.

Listing 15: The Complete Model implemented in LPL [5]

```
model guer05_6a "Paint production";
set i,j := [1..5]  "batches/sequence";
parameter
  d{i}   := [40,35,45,32,50]  "Processing time";
  c{i,j}  "Setup time (cleaning time)" :=
   [ 0,11, 7,13,11 , 5, 0,13,15,15
    13,15, 0,23,11 , 9,13, 5, 0, 3 , 3, 7, 7, 7, 0];
variable clean{i}; y{i} [1..#i];
alldiff variable x{i} [1..#i];
constraint
  A{i}: x<>i;
  B{i}: Element(x, {j}c, clean);
  --B{i}: clean[i] = c[i,x]; //KEltTerm and KElement in Kalis
  C{i,j|i<>j and j>1}: x[i] = j -> y[j] = y[i]+1;
  --C{i,j|i<>j and j>1}: KGard(x[i] = j , y[j] = y[i]+1); //in Kalis
minimize Cost: sum{i} (d+clean);
Write('Cost: %d\n', Cost);
Write('Order of batches: %d%s\n', x[#i],{i}Format(' --> %d', x));
end
```

Further Comments: This model can also be solved by a MIP solver. LPL translates the `Element` global constraint into a linear form. See my paper Various Model Types.

16 Assembly line balancing (guer05-7)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 5.7. English translation in [4] chap 7.6.

Listing 16: The Complete Model implemented in LPL [5]

```
model guer05_7 "Assembly line balancing";
set t := [1..12] "Tasks";
    p := [1..4]  "Machines";
    A{t,t} :=
      [1 2, 1 3, 2 4, 2 5, 2 6, 3 6, 3 7, 4 9, 5 9,
      6 8, 7 11, 8 9, 8 10, 9 12, 10 12, 11 10];
parameter D{t}:=[3,6,7,6,4,8,9,11,2,13,4,3] "Proc. time";
model gantt,g;
  variable T{t};
  constraint prec{A[i,j]}: T[i]+D[i] <= T[j];
  minimize makespan: sum{t} T;
end
gantt;
parameter F{t}:= g.T;
variable
  TT            "Cycle time";
  binary X{t,p} "Assign";
constraint
  OneMaschPerTask{t}: sum{p} X = 1;
  Sequence{A[i,j]}: sum{p} p*X[i,p] <= sum{p} p*X[j,p];
  UpperBound{p}: sum{t} D*X <= TT;
minimize Cycle: TT;
Writep(TT);
parameter FF{p}:=min{t|X} F;
parameter f{t};
 {p,t|X} (f[t]:=sum{t1 in t|X[t1,p] and t1<t} D[t1]+FF);
 Draw.Scale(20,20);
 Draw.DefFont('Verdana',12);
 Draw.Rect(0,0,50,10,1);
 {p} Draw.Text('ligne␣'&p,0,p+.7);
 {p} Draw.Text('␣'&sum{t|X} D,2,p+.7);
 {p,t|X} Draw.Rect(t&'',f+3,p,D,1,2,0,1);
end
```

17 Planning the production of bicycles (guer06-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 6.2. English translation in [4] chap 8.1.

Listing 17: The Complete Model implemented in LPL [5]

```
model guer06_2 "Planning the production of bicycles";
set n := [1..12]  "Time periods";
parameter
  D{n} "Demand" := [30000,15000,15000,25000,33000,40000,
                    45000,45000,26000,14000,25000,30000];
  CN    := 130   "Unit cost, normal time";
  CS    := 160   "Unit cost, over time";
  CSt   := 20    "Storage cost";
  Capa  := 30000 "Production capacilty";
  SInit := 2000  "Inituial storage";
variable
  XN{n} [0..Capa] "Quantity to produce, normal time";
  XS{n}           "Quantity to produce, overtime";
  S{n}            "Storage";
constraint
  Periode{n}: XN + XS + if(n=1,SInit,S[n-1]) = D + S;
minimize Cost: sum{n} (CN*XN + CS*XS + CSt*S);
Writep(Cost);
end
```

18 Production of drinking glasses (guer06-3)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 6.3. English translation in [4] chap 8.2.

Listing 18: The Complete Model implemented in LPL [5]

```
model guer06_3 "Production of drinking glasses";
set
  m := [1..12]  "Week periods";
  n := [1..6]   "Products";
parameter
  CAT :=  390 "Workforce capacity";
  CAM :=  850 "Capacity machines";
  CAZ := 1000 "Storage capacity";
  d{n,m} "Demand" :=
    [20, 22, 18, 35, 17, 19, 23, 20, 29, 30, 28, 32
     17, 19, 23, 20, 11, 10, 12, 34, 21, 23, 30, 12
     18, 35, 17, 10,  9, 21, 23, 15, 10,  0, 13, 17
     31, 45, 24, 38, 41, 20, 19, 37, 28, 12, 30, 37
     23, 20, 23, 15, 10, 22, 18, 30, 28,  7, 15, 10
     22, 18, 20, 19, 18, 35,  0, 28, 12, 30, 21, 23];
  C{n}:=[100 80 110 90 200 140] "Prod cost";
  CS{n}:=[25 28 25 27 10 20] "Stagage cost";
  SI{n}:=[50 20 0 15 0 10]   "Initial storgae";
  SF{n}:=[10 10 10 10 10 10] "Min. storage";
  Tut{n} := [3 3 3 2 4 4]    "Workforce time";
  Tum{n} := [2 1 4 8 11 9]   "Machine time";
  Tzs{n} := [4 5 5 6 4 9]    "Storage space";
variable
  Q{n,m} "Quantity";
  s{n,m} "Storage";
constraint
  Dem{n,m}: if(m=1,SI,s[n,m-1]) + Q - s = d;
  CapHom{m}: sum{n} Tut*Q <= CAT  "Workforce";
  CapMac{m}: sum{n} Tum*Q <= CAM  "Machine";
  CapSto{m}: sum{n} Tzs*s <= CAZ  "Storage";
  StockFin{n}: s[n,#m] >= SF "Final storage";
minimize Cost: sum{n,m} C*Q + sum{n,m} CS*s;
Writep(Cost);
end
```

19 Material Requirement Planning (guer06-4)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 6.4. English translation in [4] chap 8.3.

Listing 19: The Complete Model implemented in LPL [5]

```
model guer06_4 "Material Requirement Planning";
set n,m := [1..17]  "Components";
parameter
  B{n,m} "Quanlity of raw mat." :=
    /4 1 2,  4 2 1,    -- chassis monte
     9 3 2,  9 4 2,  9 5 1,    -- cabine montee
    12 6 1,  12 7 2,  12 8 1,     -- camion bleu
    16 9 1,  16 10 1,  16 12 1,  16 13 1,  16 15 2, -- camion rouge
    17 9 1,  17 11 1,  17 12 1,  17 14 1,  17 15 2/;
  --...CA{n}    "Raw material cost" =
  -- ... modeling not finished ...
end
```

20 Production of elec. components (guer06-5)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 6.5. English translation in [4] chap 8.4.

Listing 20: The Complete Model implemented in LPL [5]

```
model guer06_5 "Production of elec. components";
set
  m := [1..6]   "Time periods";
  n := [1..4]   "Components";
parameter
  CB := 0.5  "Cost to reduce production";
  CA := 1.0  "Cost to augment production";
  d{n,m} "Demand" :=
    [1500 3000 2000 4000 2000 2500
     1300  800  800 1000 1100  900
     2200 1500 2900 1800 1200 2100
     1400 1600 1500 1000 1100 1200];
  C{n} := [20 25 10 15] "Prod cost";
  CS{n}:= [.4 .5 .3 .3] "Storage cost";
  SI{n}:= [10 0 50 0]   "Initial stock";
  SF{n}:= [50 10 30 10] "Stock minimal";
variable
  Q{n,m} "Prod quantity";
  s{n,m} "Storage";
  B{m}   "Lower production";
  A{m}   "Raise production";
constraint
  Bal{n,m}: if(m=1,SI,s[n,m-1]) + Q = d+s;
  Chg{m|m>1}: sum{n} Q + B = sum{n} Q[n,m-1] + A;
  St{n}: s[n,#m] >= SF;
minimize
  Cost: sum{n,m}(C*Q+CS*s) + sum{m|m>1}(CB*B+CA*A);
Writep(Cost);
end
```

21 Production of fiberglass (guer06-6)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 6.6. English translation in [4] chap 8.5.

Listing 21: The Complete Model implemented in LPL [5]

```
model guer06_6 "Production of fiberglass";
set m,n := [1..12] "Periods";
parameter
  Capa{n} :=
    [140,100,100,120,110,100,100,90,120,120,100,110];
  Arcs{n,m} :=
    /1 2 5 , 3 4 8 , 5 6 6 , 7 8 6 , 9 10 7 , 11 12 6,
     2 4 0.2, 4 6 0.3, 6 8 0.2, 8 10 0.25, 10 12 0.3/;
variable x{n,m|Arcs};
constraint
  Demand{m|m%2=0}: sum{n} x[n,m] = Capa + sum{n} x[m,n];
  Cap{n|n%2=1}: sum{m} x <= Capa;
minimize Cost: sum{n,m|Arcs} Arcs*x;
Writep(Cost);
end
```

22 Assign prod batches to machines (guer06-7)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 6.7. English translation in [4] chap 8.6.

Listing 22: The Complete Model implemented in LPL [5]

```
model guer06_7 "Assign prod batches to machines";
set
  n :=[1..10] "Batches";
  m :=[1..5]  "Machines";
binary variable x{m,n} "Assign";
parameter
  c{m,n} "Production Cost"
    := [17 21 22 18 24 15 20 18 19 18
        23 16 21 16 17 16 19 25 18 21
        16 20 16 25 24 16 17 19 19 18
        19 19 22 22 20 16 19 17 21 19
        18 19 15 15 21 25 16 16 23 15];
  v{m,n} "Processing time"
    := [ 8 15 14 23  8 16  8 25  9 17
        15  7 23 22 11 11 12 10 17 16
        21 20  6 22 24 10 24  9 21 14
        20 11  8 14  9  5  6 19 19  7
         8 13 13 13 10 20 25 16 16 17];
  Ca{m} "Capacity"
    := [18 19 25 19 20];
constraint
  Affectation{n}: sum{m} x = 1;
  Capacite{m}: sum{n} v*x <= Ca;
minimize Cost: sum{m,n} c*x;
Writep(Cost);
end
```

23 Wagon load balancing (guer07-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 7-2. English translation in [4] chap 9.1.

Listing 23: The Complete Model implemented in LPL [5]

```
model guer07_2 "Wagon load balancing";
set
  n := [1..16] "Boxes";
  m := [1..3]  "Wagons";
parameter P{n}:=
  [34,6,8,17,16,5,13,21,25,31,14,13,33,9,25,25] "Weight";
variable CMax           "Max charge";
binary variable X{n,m}  "Assign";
constraint
  Affect{n}: sum{m} X = 1;
  RegleCMax{m}: sum{n} P*X <= CMax;
  --Bound: CMax >= 99; // to speed up the solver
minimize MaxCharge: CMax;
Writep(MaxCharge);
end
```

24 Barge loading (guer07-3)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 7.3. English translation in [4] chap 9.2.

Listing 24: The Complete Model implemented in LPL [5]

```
model guer07_3 "Barge loading";
  set n :=[1..7] "Clients";
  parameter C:=1500  "Capacity";
    q{n}:=[  12  31  20  25   50  40   60] "Av. qty";
    t{n}:=[  10   8   6   8   15  10   12] "Lot size";
    p{n}:=[1000 600 600 700 1200 800 1100] "Price";
    c{n}:=[  80  70  85  80   73  70   80] "Trans. cost";
    g{n}:=p-c*t "Profit";
  integer variable x{n} [0..q]  "Lots";
  constraint Capa: sum{n} t*x <= C;
  maximize Gain: sum{n} g*x;
  Writep(Gain);
end
```

25 Tank loading (guer07-4)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 7.4. English translation in [4] chap 9.3.

Listing 25: The Complete Model implemented in LPL [5]

```
model guer07_4 "Tank loading";
set
  q := [1..5]  "Liquids";
  r := [1..9]  "Tanks";
parameter
  Capa{r} := [500,400,400,600,600,900,800,800,800];
  QIni{r} := [0  ,100,  0,  0,  0,  0,300,  0,  0];
  LIni{r} := [0  ,  1,  0,  0,  0,  0,  5,  0,  0];
  Arrive{q} := [1200, 700,1000,450,1200];
binary variable X{q,r|QIni=0} "Assign";
constraint
  PasMixer{r|QIni=0}: sum{q} X <= 1;
  Suffit{q}: sum{r|QIni=0} Capa*X  >=
       Arrive - sum{r|LIni=q} (Capa-QIni);
minimize FreeCapa: sum{q,r|QIni=0} Capa*X;
Writep(FreeCapa);
end
```

26 Backing up files (guer07-5)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 7.5. English translation in [4] chap 9.4.

Listing 26: The Complete Model implemented in LPL [5]

```
model guer07_5 "Backing up files";
set
  n := [1..16]    "Files";
  m := [1..3]     "Disks";
parameter
  A{n} := [26 35 52 77 88 94 137 164 253 364
        372 388 406 432 461 851] "File size";
  B := 1400  "Disk size";
variable NbU   "Number of disks used";
binary variable X{n,m}   "Assign";
constraint
  NbUtilisees{n}: NbU >= sum{m} m*X;
  ObjDansUneSeuleBoite{n}: sum{m} X = 1;
  Capacite{m}: sum{n} A*X <= B;
minimize Nr: NbU;
Writep(Nr);
end
```

27 Cutting sheet metal (guer07-6)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 7.6. English translation in [4] chap 9.5.

Listing 27: The Complete Model implemented in LPL [5]

```
model guer07_6 "Cutting sheet metal";
set
  m := [1..4]  "Ordered sizes";
  n := [1..16] "Patterns";
parameter
  A{m,n}  "Pattern table" :=
    [1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0
     2 1 0 2 1 0 3 2 1 0 5 4 3 2 1 0
     0 0 0 2 2 2 1 1 1 1 0 0 0 0 0 0
     0 1 3 0 1 3 0 2 3 5 0 1 3 5 6 8];
  D{m}:=[8 13 5 15] "Demand";
integer variable X{n};
constraint Demande{m}: sum{n} A*X >= D;
minimize NR: sum{n} X;
Writep(NR);
end
```

28 Cutting steel bars for desk legs (guer07-7)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 7.7. English translation in [4] chap 9.6.

Listing 28: The Complete Model implemented in LPL [5]

```
model guer07_7 "Cutting steel bars for desk legs";
set
  p  := [1..3]  "Leg types";
  b  := [1..2]  "Bar types";
  n  := [1..12] "Patterns";
parameter
  N := 6 "Number per bars";
  A{n,p} "Pattern table" :=
    [0 0 2, 0 1 1, 2 0 1, 0 2 0, 2 1 0, 3 0 0 --1.5m
     0 1 2, 0 2 1, 1 0 2, 3 0 1, 0 3 0, 5 0 0]; --2m
  D{p}:=[432 500 400] "Demand";
  H{p}:=[40 60 70]    "Height";
  L{b}:=[150 200]     "Length";
integer variable X{n} "Quantity";
constraint Demande{p}: sum{n} A*X >= D;
minimize Loss: sum{n} if(n<=N,L[1],L[2])*X -sum{p} D*H;
Writep(Loss);
end
```

29 Car rental (guer08-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 8.2. English translation in [4] chap 10.1.

Listing 29: The Complete Model implemented in LPL [5]

```
model guer08_2 "Car rental";
set n,m := [1..10]  "Agencies";
parameter
  C := 3    "Km unit cost";
  Theo{n} := [10,06,08,11,09,07,15,07,09,12];
  Reel{n} := [08,13,04,08,12,02,14,11,15,07];
  X{n}    := [00,04,04,06,07,07,01,01,02,00];
  Y{n}    := [00,04,02,02,00,05,05,02,00,03];
  S{n}    := Reel - Theo;
  D{n,m}  := (Abs(X[n]-X[m])^2 + Abs(Y[n]-Y[m])^2)^0.5;
variable Z{n,m|S[n]>0 and S[m]<0};
constraint
  Surplus{n|S[n]>0}: sum{m|S[m]<0} Z = S[n];
  Besoins{m|S[m]<0}: sum{n|S[n]>0} Z = -S[m];
minimize Cost: sum{n,m} C*D*Z;
Writep(Cost,Z);
end
```

30 Choosing the mode of transport (guer08-3)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 8.3. English translation in [4] chap 10.2.

Listing 30: The Complete Model implemented in LPL [5]

```
model guer08_3 "Choosing the mode of transport";
  set n,k  "Locations";
  parameter
    m{n,k}      "Lower bound";
    Ca{n,k}     "Upper bound";
    c{n,k}      "Cost";
  Read{n,k}('guer08-3.dat', n,k,m,Ca,c);
  parameter
    S := 1      "Source";
    T := #n     "Sink";
    F := 180    "Quantity to transport";
  variable p{n,k|Ca} [m,Ca];
  constraint
    Flow{n|n<>T and n<>S}: sum{k|Ca[k,n]} p[k,n] =
                           sum{k|Ca[n,k]} p[n,k];
    QTY: sum{n|n<>S and Ca[S,n]} p[S,n] = F;
  minimize Cost: sum{n,k|Ca} c*p;
  Writep(Cost);
end
```

31 Depot location (guer08-4)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 8.4. English translation in [4] chap 10.3.

Listing 31: The Complete Model implemented in LPL [5]

```
model guer08_4 "Depot location";
set
  n := [1..12]  "Locations";
  m := [1..12]  "Costumers";
parameter
  c{n,m} default 1000000 "Cost" :=
    [100 80 50 50 60 100 120 90 60 70 65 110
     120 90 60 70 65 110 140 110 80 80 75 130
     140 110 80 80 75 130 160 125 100 100 80 150
     160 125 100 100 80 150 190 150 130 . . .
     190 150 130 . . . 200 180 150 . . .
     200 180 150 . . . 100 80 50 50 60 100
     100 80 50 50 60 100 120 90 60 70 65 110
     120 90 60 70 65 110 140 110 80 80 75 130
     140 110 80 80 75 130 160 125 100 100 80 150
     160 125 100 100 80 150 190 150 130 . . .
     190 150 130 . . . 200 180 150 . . .
     200 180 150 . . . 100 80 50 50 60 100];
  f{n}:=[35 90 100 40 30 90 90 30 40 100 90 35] "Cost";
  d{m}:=[120 80 75 100 110 100 90 60 30 150 95 120] "Demand";
  Ca{n}:=[300 250 100 180 275 300 200 220 270 250 230 180] "Capacity";
variable x{n,m} [0,1] "Transport (%)";
binary variable y{n}   "Assign";
constraint
  Satisfac{m}: sum{n} x = 1;
  Capacite{n}: sum{m} d*x <= Ca*y;
minimize Cost: sum{n} 100*f*y + sum{n,m} c*x;
Writep(Cost);
end
```

32 Heating oil delivery (guer08-5)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 8.5. English translation in [4] chap 10.4.

Listing 32: The Complete Model implemented in LPL [5]

```
model guer08_5 "Heating oil delivery";
set n,m    := [1..7]  "Clients+depot";
-- Attention, contrairement au livre, les noeuds sont ici
-- indexes dans l'ordre : Donges, Brain, Carquefou, Guerande,
-- Haie Fouassiere, Ponts-de-Ce, Mesanger. Les fichiers de
-- donnees sont reordonnes en consequence.
parameter
  Qmax := 39000;
  D{n,m} "distances" :=
    [0   148 55  32 70  140  73
     148 0   93 180 99   12  72
      55 93  0  85  20  83  28
      32 180 85 0   100 174 99
      70 99  20 100 0   85  49
     140 12  83 174 85  0   73
      73 72  28 99  49  73  0 ];
  Q{n}:=[0 14000 3000 6000 16000 15000 5000] "demand";
variable C{n};
binary variable X{n,m|n<>m};
constraint
  enter{n|n>1}: sum{m} X = 1;
  leave{m|m>1}: sum{n} X = 1;
  First{n|n>1}: C <= Qmax-(Qmax-Q)*X[1,n];
  j_Apres_i{n,m|n<>m and n>1 and m>1}: C[m] >= C[n]+Q[m]-Qmax + Qmax*X[
     n,m]+(Qmax-Q[m]-Q[n])*X[m,n];
  Bound{n|n>1}: Q <= C <= Qmax;
minimize kilometres: sum{n,m|n<>m} D*X;
Writep(kilometres);
end
```

33 Combining diff modes of transport (guer08-6)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 48.6. English translation in [4] chap 10.5.

Listing 33: The Complete Model implemented in LPL [5]

```
model guer08_6 "Combining diff modes of transport";
set
  n := [1..5]  "Locations";
  m,m1 := [1..3]  "Modes (rail, route, air)";
parameter
  C{m,n}:=[30 40 30 60 . , 20 40 50 50 . , 40 10 60 40 .];
  T{m,m1}:=[0 2 1,  2 0 1,  2 1 0] "Mode change mode";
binary variable
  X{m,n|n<#n}  "Assign";
  Y{m,m1,n|n>1 and n<#n}   "Change Mode";
constraint
  UnMode{n|n<#n}: sum{m} X = 1;
  UnTransf{n|n>1 and n<#n}: sum{m,m1} Y = 1;
  Cons{n,m,m1|n>1 and n<#n}: X[m,n-1] + X[m1,n] >= 2*Y;
minimize Cost: sum{m,n|n<#n} C*X + sum{n,m,m1|n<#n} T*Y;
Writep(Cost);
end
```

34 Fleet planning for vans (guer08-7)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 8.7. English translation in [4] chap 10.6.

Listing 34: The Complete Model implemented in LPL [5]

```
model guer08_7 "Fleet planning for vans";
set
  t := [1..3]    "Contracts";
  m,m1 := [1..6] "Months";
parameter
  Z  := 200  "Initial stock";
  B{m} := [ 430,  410,  440, 390, 425, 450]  "Needs";
  C{t} := [1700, 2200, 2600]  "Cost";
variable Y{t,m}  "Number vans";
constraint Cover{m}: if(m<=2,Z) + sum{t,m1|m1 >=
    Max(1,m-t-1) and m1<=Min(m,#m-t-1)} Y[t,m1] >= B;
minimize Cost: sum{t,m} C*Y;
Writep(Cost);
end
```

35 Flight connections at a hub (guer09-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 9.2. English translation in [4] chap 11.1.

Listing 35: The Complete Model implemented in LPL [5]

```
model guer09_2 "Flight connections at a hub";
set n,m := [1..6]  "Locations";
parameter
  P{n,m} "Number of passengers" :=
    [   35  12  16 38  5 2 ,    25     8     9 24  6 8
        12   8  11 27  3 2 ,    38    15    14 30  2 9
     -1000   9   8 25 10 5 , -1000 -1000 -1000 14  6 7];
variable X{n,m};
constraint
  OneIn{n}: sum{m} X = 1;
  OneOut{m}: sum{n} X = 1;
maximize NbPassagers: sum{n,m} P*X;
Writep(NbPassagers);
end
```

36 Composing flight crews (guer09-3)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 9.3. English translation in [4] chap 11.2.

Listing 36: The Complete Model implemented in LPL [5]

```
model guer09_3 "Composing flight crews";
set n,m,m1  "Number of pilots";
parameter
  G{n,m} "Score" :=
    /1 2 30, 1 4 24, 1 5 25, 2 3 27, 2 4 28, 2 6 27,
    3 6 26, 3 7 30,  4 5 29, 4 6 21, 5 6 30, 5 8 30,
    6 7 28, 6 8 36, 7 8 25/;
binary variable X{n,m|G};
constraint Sep{n}: sum{m,m1|m=n or m1=n} X[m,m1] <= 1;
maximize Score: sum{n,m} G*X;
Writep(Score);
end
```

37 Scheduling flight langings (guer09-4)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 9.4. English translation in [4] chap 11.3.

Listing 37: The Complete Model implemented in LPL [5]

```
model guer09_4 "Scheduling flight langings";
set n,m := [1..10] "Aeroplanes";
parameter
  M := 1000   "Big-M";
  inf:=999999;
  Deb{n}:=[129 195  89  96 110 120 124 126 135 160]
       "Begin time window";
  Fin{n}:=[559 744 510 521 555 576 577 573 591 657]
       "End time window";
  Pre{n}:=[155 258  98 106 123 135 138 140 150 180]
       "Prefered time";
  Ava{n}:= [10  10  30  30  30  30  30  30  30  30]
       "Penalty for too early";
  Ret{n}:= [10  10  30  30  30  30  30  30  30  30]
       "Penalty for too late" ;
  Sep{n,m}   "Delay" :=
    [99999 3 15 15 15 15 15 15 15 15
     3 99999 15 15 15 15 15 15 15 15
     15 15 99999 8 8 8 8 8 8 8
     15 15 8 99999 8 8 8 8 8 8
     15 15 8 8 99999 8 8 8 8 8
     15 15 8 8 8 99999 8 8 8 8
     15 15 8 8 8 8 99999 8 8 8
     15 15 8 8 8 8 8 99999 8 8
     15 15 8 8 8 8 8 8 99999 8
     15 15 8 8 8 8 8 8 8 99999];
variable
  T{n} [Deb..Fin]    "Langing time";
  A{n} [0..Pre-Deb]  "Too early";
  R{n} [0..Fin-Pre]  "Too late";
binary variable X{n,m|n<m};
constraint
  cart1{n,m|m<n}: T + Sep <= T[m] + M*X[m,n];
  Ecart2{n,m|m>n}: T + Sep <= T[m] - M*X[n,m] + M;
  Avance{n}: A >= Pre - T;
  Retard{n}: R >= T - Pre;
  AvOuRe{n}: T = Pre -A + R;
minimize Penalty: sum{n} (Ava*A + Ret*R);
Writep(Penalty);
end
```

38 Airline hub location (guer09-5)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 9.5. English translation in [4] chap 11.4.

Listing 38: The Complete Model implemented in LPL [5]

```
model guer09_5 "Airline hub location";
set n,m,k := [1..6]   "Locations";
parameter
  p := 2   "Number of hubs";
  Alpha := 80/100 "Cost reduction";
  D{n,m}          "distances" :=
    [   0,  870,  702, 4420, 4300, 4380
      870,    0,  840, 3660, 3620, 3640
      702,  840,    0, 4200, 4100, 4140
     4420, 3660, 4200,    0,  140,  571
     4300, 3620, 4100,  140,    0,  501
     4380, 3640, 4140,  571,  501,    0];
  Q{n,m}          "transport quantity" :=
    [   0,  500, 1000, 300, 400, 1500
     1500,    0,  250, 630, 360, 1140
      400,  510,    0, 460, 320,  490
      300,  600,  810,   0, 820,  310
      400,  100,  420, 730,   0,  970
      350, 1020,  260, 580, 380,    0];
  C{n,m,k,n1 in n}:=D[n,k]+Alpha*D[k,n1]+D[n1,m];
binary variable
  Z{n,m,k,n1 in n};
  X{n};
constraint
  pHubs: sum{n} X = p  "Exactly p Hubs";
  DeuxVilDeuxHubs{n,m}  : sum{k,n1 in n} Z[n,m,k,n1] = 1;
  kHub{n,m,k,n1 in n}: Z[n,m,k,n1] <= X[k];
  mHub{n,m,k,n1 in n}: Z[n,m,k,n1] <= X[n1];
minimize Cost:
       sum{n,m,k,n1 in n} C[n,m,k,n1]*Q[n,m]*Z[n,m,k,n1];
Writep(Cost);
end
```

39 Planning a flight tour (guer09-6)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 9.6. English translation in [4] chap 11.5.

Listing 39: The Complete Model implemented in LPL [5]

```
model guer09_6 "Planning a flight tour";
set n,m := [1..7]  "Locations";
  --/Bordeaux Lille Lyon Marseille Nantes Paris Toulouse/;
parameter
  d{n,m} "Distance" :=
    /1 2 786, 1 3 549, 1 4 657, 1 5 331, 1 6 559,
     1 7 250, 2 3 668, 2 4 979, 2 5 593, 2 6 224,
     2 7 905, 3 4 316, 3 5 607, 3 6 472, 3 7 467,
     4 5 890, 4 6 769, 4 7 400, 5 6 386, 5 7 559,
     6 7 681/;
binary variable y{n,m};
constraint
  Entree{m}: sum{n|n<>m} y = 1;
  Sortie{n}: sum{m|n<>m} y = 1;
  S1: y[2,6] + y[6,2] <= 1;
  S2: y[1,7] + y[7,1] <= 1;
minimize Trajet: sum{n,m} if(n<m,d[n,m],d[m,n])*y;
Writep(Trajet);
end
```

40 Network reliability (guer10-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 10.2. English translation in [4] chap 12.1.

Listing 40: The Complete Model implemented in LPL [5]

```
model guer10_2 "Network reliability";
  set
    n,m := [1..11]  "nodes";
    G{n,m}  "arcs" :=
      [1 2, 1 3, 1 11, 2 3, 2 8, 2 9, 3 4, 3 9, 3 10,
       3 11, 4 5, 4 6, 4 11, 5 9, 5 11, 6 7, 6 9, 6 10,
       7 8, 7 10, 8 10, 9 10];;
  G{n,m}:=G[n,m] or G[m,n];
  parameter
    s := 10  "Source";
    t := 11  "Sink";
  variable Phi{n,m|G[n,m] or G[m,n]};
  constraint
    A{n|n<>s and n<>t}: sum{m} Phi[m,n] = sum{m} Phi[n,m];
    B{n|n<>s and n<>t}: sum{m} Phi[n,m] <= 1;
    C: sum{n} Phi[n,s] = 0;
  maximize Debit: sum{n} Phi[s,n];
  Writep(Debit);
  parameter x{n} :=[2 4 2 1 2 4 6 6 4 5 1];
            y{n} :=[1 1 2 4 3 4 4 1 3 2 3];
  Draw.Scale(100,100);
  {G[n,m]} Draw.Line(x[n],y[n],x[m],y[m],0,if(Phi[n,m],4,1));
  {n} Draw.Circle(n&'',x,y,.3,1,0);
end
```

41 Dimensioning of a mobile phone network (guer10-3)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 10.3. English translation in [4] chap 12.2.

Listing 41: The Complete Model implemented in LPL [5]

```
model guer10_3 "Dimensioning of a mobile phone network";
set
  c := [1..10] "Cells";
  n := [1..5]  "Hubs nodes";
parameter
  K := 48  "Capacity";
  C{c,n}  "Cost" :=
    [15  8  7 11 10,   9 11  8  5 14,  12  6  7 15 15
     17  5  9 18 24,   8 22 21 19  6,   7 25 15  9 17
     19 25 21 20 22,  20  9 15 18 25,  21 22 14 16 20,
     25 24 13  4 11];
  T{c}:=[22 12 20 12 15 25 15 14  8 22] "traffic";
  D{c}:=[2   2   2   2   3   1   3   2   2   2 ];
binary variable X{c,n}  "Assign";
constraint
  Divers{c}: sum{n} X = D;
  CapaciteAnneau: sum{c,n|n<#n} (T/D)*X <= 2*K;
minimize Cost: sum{c,n} C*X;
Writep(Cost);
end
```

42 Routing telephone calls (guer10-4)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 10.4. English translation in [4] chap 12.3.

Listing 42: The Complete Model implemented in LPL [5]

```
model guer10_4 "Routing telephone calls";
set
  a,b := [1..6]  "Nodes";
  p := [1..5]  "Pair numbers";
  c := [1..17] "Paths";
parameter
  Capa{a}:=[300,120,300,200, 80, 70] "Capacity";
  Chem{c,a} "Paths table" :=
    [2 0 0 0 0 0, 1 3 0 0 0 0, 1 4 6 5 0 0, 1 4 6 0 0 0,
     1 3 5 0 0 0, 2 5 0 0 0 0, 2 3 4 6 0 0, 1 4 0 0 0 0,
     2 3 4 0 0 0, 1 3 5 6 0 0, 2 5 6 0 0 0, 2 1 4 0 0 0,
     3 4 0 0 0 0, 5 6 0 0 0 0, 4 6 0 0 0 0, 1 2 5 0 0 0,
     3 5 0 0 0 0];
  Pi{c}:=[1 1 1 2 2 2 2 3 3 3 3 4 4 4 5 5 5];
  D{p}:=[100  80  75 100  70] "Demand";
integer variable Phi{c}   "Flow";
constraint
  RespD{p}: sum{c|Pi=p} Phi <= D;
  RespC{b}: sum{c,a|Chem[c,a]=b} Phi[c] <= Capa[b];
maximize Flow: sum{c} Phi;
Writep(Flow);
end
```

43 Construction of a cabled networkt (guer10-5)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 10.5. English translation in [4] chap 12.4.

Listing 43: The Complete Model implemented in LPL [5]

```
model guer10_5 "Construction of a cabled networkt";
set n,m := [1..6]   "nodes";
parameter
  D{n,m} "Distances" :=
    [ 0 120 92 265 149 194, 120 0 141 170 93 164, 92 141
      0 218 103 116, 265 170 218 0 110 126, 149 93 103
      110 0 72, 194 164 116 126 72 0];
variable Level{n};
binary variable X{n,m|n<>m};
constraint
  NbConnexions: sum{n,m} X = #n-1;
  Connexe{n,m|m<>n}: Level[m] >= Level[n]+1-#n+#n*X;
minimize Cost: sum{n,m} D*X;
Writep(Cost);
end
```

44 Scheduling of telecomm. via satellite (guer10-6)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 10.6. English translation in [4] chap 12.5.

Listing 44: The Complete Model implemented in LPL [5]

```
model guer10_6 "Scheduling of telecomm. via satellite";
  set n,m := [1..4]    "stations";
  parameter
    T{n,m}:=[7 12 11 15, 15 8 13 9, 17 12 6 10, 6 13 15 11]  "traffic";
    LB;
  binary variable X{n,m|T};
  variable PMin;
  constraint
    UnRecParEme{n}: sum{m} X = 1;
    UnEmeParRec{m}: sum{n} X = 1;
    ReglePMin{n}: sum{m} T*X >= PMin;
  parameter i:=2;
  while i>0 do
    maximize obj: PMin;
    LB:=LB+PMin;
    Writep(PMin,T,X);
    T{n,m}:=if(X=1,T-PMin,T);
    --if(XX,T-PMin,T);
    ClearData(X);
    i:=i-1;
  end;
  Writep(LB);
end
```

45 Localisation of GSM transmitters (guer10-7)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 10.7. English translation in [4] chap 12.6.

Listing 45: The Complete Model implemented in LPL [5]

```
model guer10_7 "Localisation of GSM transmitters";
set
  n := [1..15] "Villages to deserve";
  m := [1..7]  "Locations of transmitters";
  Cv{m,n} "Covering" :=
    [1 1, 1 2, 1 4, 2 2, 2 3, 2 5, 3 4, 3 7, 3 8, 3 10,
     4 5, 4 6, 4 8, 4 9, 5 8, 5 9, 5 12, 6 7, 6 10, 6 11,
     6 12, 6 15, 7 12, 7 13, 7 14, 7 15];
parameter
  B := 10  "Budget maximal";
  p{n}:=[2,4,13,6,9,4,8,12,10,11,6,14,9,3,6]"Population";
  c{m}:=[1.8, 1.3, 4.0, 3.5, 3.8, 2.6, 2.1] "Cost";
binary variable
  x{m} "Transmitter there?";
  y{n} "Couvered?";
constraint
  CouvCom{n}: sum{m|Cv} x >= y;
  Budget: sum{m} c*x <= B;
maximize MaxPop: sum{n} p*y;
Writep(MaxPop);
end
```

46 Choice of loans (guer11-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 11.2. English translation in [4] chap 13.1.

Listing 46: The Complete Model implemented in LPL [5]

```
model guer11_2 "Choice of loans";
set
  i := [1..3]   "Shops";
  j := [1..3]   "Banks";
parameter
  n    := 8         "Time horizon";
  Mmax := 3000000   "Max amount to loan";
  T{j,i}:=[0.05 0.065 0.061, 0.052 0.062 0.062,
           0.055 0.058 0.065] "Interest";
  P{i}:=[2500000,1000000,1700000] "Price of shop";
variable X{j,i};
constraint
  A{i}: sum{j} X = P;
  B{j}: sum{i} X <= Mmax;
minimize Annuities: sum{j,i} X*T/(1-(1+T)^(-n));
Writep(Annuities);
end
```

47 Publicity campaign (guer11-3)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 11.3. English translation in [4] chap 13.2.

Listing 47: The Complete Model implemented in LPL [5]

```
model guer11_3 "Publicity campaign";
set n "Various medias";
string parameter nN{n};
parameter
  ca{n} "Potential";
  c{n}  "Cost";
  nd{n} "Usage";
  q{n}  "Quality index";
  B := 250000  "Max budget";
  C := 100000  "Number of clients to reach";
Read{n}('guer11-3.dat', n,nN,c,ca,q,nd);
integer variable m{n} [0..nd];
constraint
  BudgetMax  : sum{n} c*m <= B;
  CibleMini  : sum{n} ca*m >= C;
maximize QualPercep: sum{n} q*m;
Writep(QualPercep,m);
end
```

48 Portfolio selection (guer11-4)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 11.4. English translation in [4] chap 13.3.

Listing 48: The Complete Model implemented in LPL [5]

```
model guer11_4 "Portfolio selection";
set
  n := [Dash 'Ilog␣France' Eurodecision 'Ilog␣Usa'
        'France␣Telecom' BNP];
  Tech{n} := n<=3;
  France{n}:= ~(n=1 or n=4);
parameter
  C := 100000  "Capital";
  r{n}:=[5.3, 6.2, 5.1, 4.9, 6.5, 3.4] "Interest";
variable v{n} [5000..40000];
constraint
  Technologiques : sum{Tech} v <= 0.3*C;
  Francaises     : sum{France} v >= 0.5*C;
  CapitalMaximum : sum{n} v <= C;
maximize RetourTotal: sum{n} r/100*v;
Writep(RetourTotal,v);
end
```

49 Finacing an early retirment scheme (guer11-5)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 11.5. English translation in [4] chap 13.4.

Listing 49: The Complete Model implemented in LPL [5]

```
model guer11_5 "Finacing an early retirment scheme";
set
  n := [1..7]   "Time horizon";
  e := [1..3]   "Bond types";
parameter
  Taux := 3.2  "Annual interest rate";
  F{n}:=[1000 600 640 480 760 1020 950] "Needs";
  p{e}:=[1.0, 0.8, 0.5]  "Priice";
  t{e}:=[7.0, 7.0, 6.5]  "Interestt";
  T{e} := p*t/100;
  TotalaPayer:=sum{n} F;
variable
  CT     "Capital";
  I{n}   "Annual investment";
  integer N{e} [0..5000]  "Quantity to buy";
constraint
  year1: CT - sum{e} p*N - I[1] = F[1];
  year{n|n>1 and n<6}:sum{e} T*N+(1+Taux/100)*I[n-1]-I=F;
  year6: (p[1]+T[1])*N[1] + (p[2] + T[2])*N[2]
    + T[3]*N[3] + (1+Taux/100)*I[5] - I[6] = F[6];
  year7: (p[3]+T[3])*N[3] + (1+Taux/100)*I[6] = F[7];
minimize Capital: CT;
Writep(CT,TotalaPayer,N);
parameter EE{n}:= if(n<=5,sum{e} p*N,n=6,p[3]*N[3],0);
          II{n}:=I;
          FF{n}:=F;
   Draw.Scale(20,-5);
   {n}  Draw.Rect(n,0,1,EE/100,2,0,1);
   {n}  Draw.Rect(n,EE/100,1,II/100,3,0,1);
   {n}  Draw.Rect(n,EE/100+II/100,1,FF/100,4,0,1);
end
```

50 Family budget (guer11-6)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 11.6. English translation in [4] chap 13.5.

Listing 50: The Complete Model implemented in LPL [5]

```
model guer11_6 "Family budget";
set
  n := [1..12]  "Months";
  i := [VieCourante, Loyer, Voiture, Telephone,
        Impots, 'EDF/GDF']  "Expenses";
parameter
  A :=   900  "Public allocations";
  S := 11500  "Salary";
  D{i} := [3300, 3800, 2000, 800, 600, 5000]  "Expenses";
  m{i} := [1,1,1,2,4,6]  "mensualities";
  d{n,i} := if(n%m,0,D)  "Expenses per month";
variable
  l{n} [1000..9999999] "Leigure";
  e{n}  "Sauvings";
constraint Bal{n}: sum{i} d + l + e <= S + A + e[n-1];
maximize Lei: sum{n} l;
Writep(Lei,l,e);
end
```

51 Choice of expansion projects (guer11-7)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 11.7. English translation in [4] chap 13.6.

Listing 51: The Complete Model implemented in LPL [5]

```
model guer11_7 "Choice of expansion projects";
  set
    a := [1..5]  "Time horizon";
    n := [1..5]  "Projects";
  parameter
    ca{a}:=[4.8 6.0 4.8 4.2 3.5] "Capital";
    c{n,a}  "Project cost" :=
      [1.80 2.40 2.40 1.80 1.50
       1.20 1.80 2.40 0.60 0.50
       1.20 1.00 0    0.48 0
       1.40 1.40 1.20 1.20 1.20
       1.60 2.10 2.50 2.00 1.80];
    b{n}:=[10.8 4.8 3.2 4.44 12.25] "Profit";
  binary variable x{n};
  constraint Invest{a}: sum{n} c*x <= ca;
  maximize Profit: sum{n} b*x;
  Writep(Profit,x);
end
```

52 Mean variance portfolio (guer11-8)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 11.8. English translation in [4] chap 13.7.

Listing 52: The Complete Model implemented in LPL [5]

```
model guer11_8 "Mean variance portfolio";
  set
    i,j := [Hardware Software Show_biz T_bills] "Investment types";
  parameter
    r{i}:=[8 9 12 7] "expected return";
    V{i,j}  "co/variance" :=
      [ 4  3  -1  0 , 3  6   1  0
       -1  1  10  0 , 0  0   0  0];
    Target := 8.5 "minimum target yield";
  variable x{i}  "Fraction of investment";
  binary variable y{i};
  constraint A: sum{i} x = 1;
  constraint B: sum{i} r*x >= Target;
  constraint C: sum{i} y <= 2;
  constraint D{i}: x <= y;
  minimize risk: sum{i,j} x[i]*V[i,j]*x[j];
  Writep(risk,x);
end
```

53 Assigning personnel to machines (guer12-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 12.2. English translation in [4] chap 14.1.

Listing 53: The Complete Model implemented in LPL [5]

```
model guer12_2b "Assigning personnel to machines";
set n,m := [1..6]  "Persons";
parameter
  P{n,m}  "Productivity" :=
    [13 24 31 19 40 29 , 18 25 30 15 43 22
     20 20 27 25 34 33 , 23 26 28 18 37 30
     28 33 34 17 38 20 , 19 36 25 27 45 24];
binary variable X{n,m}  "Assign";
constraint
  UneMachParPers{n}: sum{m} X = 1;
  UnePersParMach{m}: sum{n} X = 1;
maximize ProdTotale: sum{n,m} P*X;
Writep(ProdTotale);
--maximize MinProd: min{n} (sum{m} P*X);
--Writep(MinProd); //maximize a minimal prodictivity !
end
```

54 Scheduling nurses (guer12-3)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 12.3. English translation in [4] chap 14.2.

Listing 54: The Complete Model implemented in LPL [5]

```
model guer12_3b "Scheduling nurses";
set n := [1..12]   "Time windows";
parameter
  B{n}:=[35,40,40,35,30,30,35,30,20,15,15,15] "Demand";
  D  := 80  "Total nurses";
integer variable
  X{n}  "Number of nurses beginning";
  S{n}  "an overtime window";
constraint
  Dispo frozen: sum{n} X <= D;
  NbSup0{n}: S <= 0;
  NbSup1{n} frozen: S <= X;
  tr1 : S[8]+X[9]+X[10]+X[12]+X[1]  >= B[1];
  tr2 : S[9]+X[10]+X[11]+X[1]+X[2]  >= B[2];
  tr3 : S[10]+X[11]+X[12]+X[2]+X[3] >= B[3];
  tr4 : S[11]+X[12]+X[1]+X[3]+X[4]  >= B[4];
  tr5 : S[12]+X[1]+X[2]+X[4]+X[5]   >= B[5];
  tr6 : S[1]+X[2]+X[3]+X[5]+X[6]    >= B[6];
  tr7 : S[2]+X[3]+X[4]+X[6]+X[7]    >= B[7];
  tr8 : S[3]+X[4]+X[5]+X[7]+X[8]    >= B[8];
  tr9 : S[4]+X[5]+X[6]+X[8]+X[9]    >= B[9];
  tr10: S[5]+X[6]+X[7]+X[9]+X[10]   >= B[10];
  tr11: S[6]+X[7]+X[8]+X[10]+X[11]  >= B[11];
  tr12: S[7]+X[8]+X[9]+X[11]+X[12]  >= B[12];
minimize NbNurses: sum{n} X "Total nurses without overtime";
Writep(NbNurses);
Unfreeze(Dispo,NbSup1); Freeze(NbSup0);
minimize TotalSup: sum{n} S  "with overtime";
Writep(TotalSup);
end
```

Questions

1. Formulate the tr* constraints as a single indexed constraint.

55 Establishing a college timetable (guer12-4)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 12.4. English translation in [4] chap 14.3.

Listing 55: The Complete Model implemented in LPL [5]

```
model guer12_4 "Establishing a college timetable";
set
  p := [1..9]  "Teachers";
  c := [1..2]  "Classes";
  td:= [1..20];
parameter
  t   := 4   "Time slots per day";
  d   := 5   "Days in a week";
  NbC{p,c}:=[1 1, 3 3, 2 2, 0 4, 4 0, 3 3, 1 1, 1 0, 0 1] "Number of
     courses";
string Prf{p}:=['a' 'b' 'c' 'd' 'e' 'f' 'g' 'h' 'i'];
variable X{p,c,td} "Assign";
constraint
  TousPlanif{p,c}: sum{td} X = NbC;
  Classe{c,td}: sum{p} X <= 1;
  Prof{p,td}: sum{c} X <= 1;
  Sport1: X[8,1,15] = 1;
  Sport2: X[9,2,15] = 1;
  Devoir{p,c}: X[p,c,1] = 0;
  Math{td|td<=2}: X[4,2,td] = 0;
  Biologie{c,td|c<=2 and td>8 and td<13}: X[2,c,td] = 0;
  AuPlusUn{p,c,td|td<=d}:
    sum{k in td|k>=(td-1)*t+1 and k<=td*t} X[p,c,k] <= 1;
minimize Trou: sum{p,c,td|td<=d} (X[p,c,(td-1)*t+1]
                                  + X[p,c,(td-1)*t+4]);
parameter Res{c,td} := argmin{p|X} p;
string parameter Table{c,td} := Prf[Res];
Writep(Trou);
end
```

56 Exam scheduling (guer12-5)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 12.5. English translation in [4] chap 14.4.

Listing 56: The Complete Model implemented in LPL [5]

```
model guer12_5a "Exam scheduling";
set
  e,f := ['Analyse␣de␣donnees','Analyse␣numerique','C++',
  'Genie␣logiciel','Gestion␣de␣production','Java',
  'Modeles␣et␣Algorithmique','Programmation␣logique',
  'Programmation␣mathematique', 'Statistiques',
  'Systemes␣a␣evenements␣discretes'];
  t := ['1:8-10h','1:10-12','1:14-16','1:16-18','2:8-10h',
        '2:10-12','2:14-16','2:16-18'];
  --e,f := [1..11];
  --t := [1..8];
parameter
  Incomp{e,f} "Incompatibles" :=
    [0 1 0 0 1 0 1 0 0 1 1 , 1 0 0 0 1 0 1 0 0 1 1
     0 0 0 1 1 1 1 0 1 1 1 , 0 0 1 0 1 1 1 0 0 1 1
     1 1 1 1 0 1 1 1 1 1 1 , 0 0 1 1 1 0 1 0 1 1 1
     1 1 1 1 1 1 0 1 1 1 1 , 0 0 0 0 1 0 1 0 0 1 1
     0 0 1 0 1 1 1 0 0 1 1 , 1 1 1 1 1 1 1 1 1 0 1
     1 1 1 1 1 1 1 1 1 1 0];
binary variable X{e,t}  "Assign";
constraint
  AllScheduled{e}: sum{t} X = 1 "Each exam takes place";
  Incomp1{e,f,t|Incomp}: X[e,t] + X[f,t] <= 1;
  --T{t}: Sum{e} X <= 1;
--maximize any: X[1,1]+X[2,2];
minimize length: sum{e,t} t^2 * X;
---binary variable Y{t} <- or{e} X;
---minimize tranche_horaire: sum{t} Y;
--maximize tranche: sum{e} X[e,3];
Write('␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣%10s\n', {t}t);
Write{e}('%31s␣%10s\n', e, {t} if (X,'X','␣'));
model data1 "larger data set (for testing)";
  e := [1..100];
  t := [1..20];
  Incomp{e,f|e<>f}:=if(e<f and Rnd(0,1)<.4,1);
  Incomp{e,f|e>f}:=Incomp[f,e];
end
end
```

57 Personnel assignment (guer12-6)

—- Run LPL Code , HTML Document –

Problem: The model is from [2], Chapter 12.6. and [4].

Listing 57: The Complete Model implemented in LPL [5]

```
model guer12_6 "Personnel assignment";
set
  n     "Products";
  L,L1  "Production lines";
parameter
  g{n}     "Gain";
  d{n,L}   "Processing time";
  m{L}     "Time capacity";
  T{L,L1}  "Transfert possible";
  tm{L}    "Max hours to transfert";
variable
  q{n}      "Quantity";
  h{L}      "Work hours";
  t{L,L1|T} "Hours transfered";
constraint
  ChargeLigne{L}: sum{n} d*q <= m;
  Travail{L}: sum{n} d*q <= h;
  Equilibrage{L1}: h - sum{L|T[L,L1]} t[L,L1] + sum{L|T[L1,L]} t[L1,L]
     = m;
  MaxTranfert{L}: sum{L1|T=1} t <= tm;
expression Profit: sum{n} g*q;
maximize obj1: Profit subject_to ChargeLigne;
Writep(Profit,q);
maximize obj2: Profit subject_to Travail,Equilibrage,MaxTranfert;
Writep(Profit,q);
model data;  --- in the book
  n := [1..4];
  L := [1..5];
  g{n} := [7, 8, 9, 7];
  d{n,L} :=
    [1.3, 0.9, 2.0, 0.3, 0.9
     1.8, 1.7, 1.4, 0.6, 1.1
     1.3, 1.2, 1.3, 1.0, 1.4
     0.9, 1.1, 1.0, 0.9, 1.0];
  m{L} := [4500, 5000, 4500, 1500, 2500];
  T{L,L1} :=  /1 2 1, 1 3 1, 1 4 1, 2 3 1, 2 5 1,
    3 1 1, 3 2 1, 3 4 1, 4 5 1, 5 1 1, 5 2 1, 5 3 1/;
  tm{L} := [400 800 200 500 300];
end
model data1; --- a random data set
  n := [1..20];
  L := [1..5];
  g{n} := Trunc(Rnd(6,10));
  d{n,L} := Trunc(Rnd(10,20))/10;
  m{L} := Trunc(Rnd(1500,6000));
  T{L,L1|L<>L1} := if(Rnd(0,1)<.3,1,0);
  tm{L} := Trunc(Rnd(200,900));
end
end
```

58 Personnel at a construction site (guer12-7)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 12.7. English translation in [4] chap 14.6.

Listing 58: The Complete Model implemented in LPL [5]

```
model guer12_7 "Personnel at a construction site";
set n := [1..6]  "Time horizon";
parameter
  CouArr := 100  "Arrival cost";
  CouDep := 160  "Departure cost";
  CouSur := 200  "Overtime cost";
  CouSou := 200  "Undertime cost";
  EffDeb :=   3  "Initial needs";
  EffFin :=   3  "Final needs";
  Besoin{n} := [4,6,7,4,6,2]   "Needs";
integer variable
  XEff{n};
  XArr{n} [0..3];
  XDep{n};
  XSur{n};
  XSou{n};
constraint
  Satisf{n}: XEff - XSur + XSou = Besoin;
  PremierMois: XEff[1] = EffDeb + XArr[1];
  DernierMois: EffFin  = XEff[#n] - XDep[#n];
  Autres{n|n>1}: XEff = XEff[n-1] + XArr - XDep[n-1];
  Dep{n}: XDep <= (1/3)*XEff;
  Sou{n}: XSou <= (1/4)*XEff;
minimize Cost: sum{n} (CouArr*XArr + CouDep*XDep + CouSur*XSur + CouSou
    *XSou);
Writep(Cost);
end
```

59 Water conveyance management (guer13-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 13.2. English translation in [4] chap 15.1.

Listing 59: The Complete Model implemented in LPL [5]

```
model guer13_2 "Water conveyance management";
set n,m := [1..12] "nodes";
parameter
  s := #n-1      "Source";
  t := #n        "Sink";
  Capa{n,m} :=
    /1 3 20, 1 5 15, 1 6 12, 2 5 6, 2 6 22, 3 4 15,
     3 5 10, 4 8 7, 4 9 10, 5 8 10, 5 9 15, 5 10 15,
     6 7 22, 7 9 10, 7 10 10, 8 12 18, 9 12 15, 10 12 20,
     11 1 35, 11 2 25/;
  variable Phi{n,m|Capa} [0..Capa];
  constraint FL{n|n<>s and n<>t}:
      sum{m} Phi[n,m] = sum{m} Phi[m,n];
  maximize Flow: sum{n} Phi[n,t];
  Writep(Flow);
end
```

60 CCTV surveillance (guer13-3)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 13.3. English translation in [4] chap 15.2.

Listing 60: The Complete Model implemented in LPL [5]

```
model guer13_3 "CCTV surveillance";
  set i,j:=[1..49]; -- the network
    e{i,j}:=[(1 3),(1 2),(2 39),(2 41),(3 4),(4 5),(4 6),
      (4 9),(6 8),(6 7),(9 10),(10 11),(11 21),(12 15),
      (12 13),(13 14),(14 15),(14 18),(15 19),(15 16),
      (16 20),(17 18),(18 19),(19 20),(20 21),(21 22),
      (21 23),(22 25),(23 32),(24 25),(25 26),(25 30),
      (26 27),(26 28),(28 29),(29 31),(30 31),(31 32),
      (31 33),(32 39),(33 34),(33 37),(34 35),(35 36),
      (37 38),(37 43),(38 40),(39 40),(40 41),(41 42),
      (43 44),(44 49),(44 45),(45 46),(45 47),(47 48)];
  parameter X{i}:= [42 170 6 32 71 38 76 49 32 56 50 6
     3 4 19 29 8 16 32 43 65 69 78 40 86 92 91 110 116
     93 114 124 142 130 145 138 163 170 144 177 197 212
     190 186 196 158 183 182 197] "xy-coordinate";
    Y{i}:= [3 55 84 73 88 64 78 34 88 100 103 98 110 133
     120 116 160 155 146 138 125 132 114 188 159 167 175
     172 159 144 152 132 162 185 193 214 169 150 116 130
     136 184 182 194 220 225 216 230 197];;
  e{i,j}:= e[i,j] or e[j,i];
  binary variable x{i};
  constraint A{i,j|e[i,j]}: x[i] + x[j] >= 1;
  minimize Nr: sum{i} x;
  Writep(Nr);
  Draw.Scale(2,2);
  Draw.DefFont('Verdana',10);
  {e[i,j]} Draw.Line(X[i],Y[i],X[j],Y[j]);
  {i} Draw.Circle(i&'',X,Y,4,if(x,3,1),0);
end
```

61 Rigging elections (guer13-4)

—- Run LPL Code , HTML Document –

Problem: A complete problem description and how to model it mathematically is explained in [2], Chapter 13.4. and and [4] Chapter 15.3.

Listing 61: The Complete Model implemented in LPL [5]

```
model guer13_4 "Rigging elections";
set
  q := [1..14]    "Quarters";
  c := [1..46]    "Districts";
parameter
  R := 6;
  M{c}:= [0,0,1,1,0,0,0,1,0,1,1,1,1,1,0,0,0,1,1,0,
     0,0,0,0,0,0,1,1,1,0,0,0,0,0,1,1,1,1,1,1,0,1,1,0,0,1];
  A{c,q} := /*$I 'guer13-4.dat' */ ;
binary variable X{c};
constraint
  Partition{q}: sum{c} A*X = 1;
  NombreRequis: sum{c} X = R;
minimize NombreSieges: sum{c} M*X;
Writep(NombreSieges,X);
end
```

62 Gritting roads (guer13-5)

—- Run LPL Code , HTML Document –

Problem: Leaving from his home post office, a postman needs to visit the households on each block in his route, delivering and collecting letters, and then returning to the post office. He would like to cover this route by travelling the minimum possible distance. The post office is located at the intersection point 1, (see Figure 2).

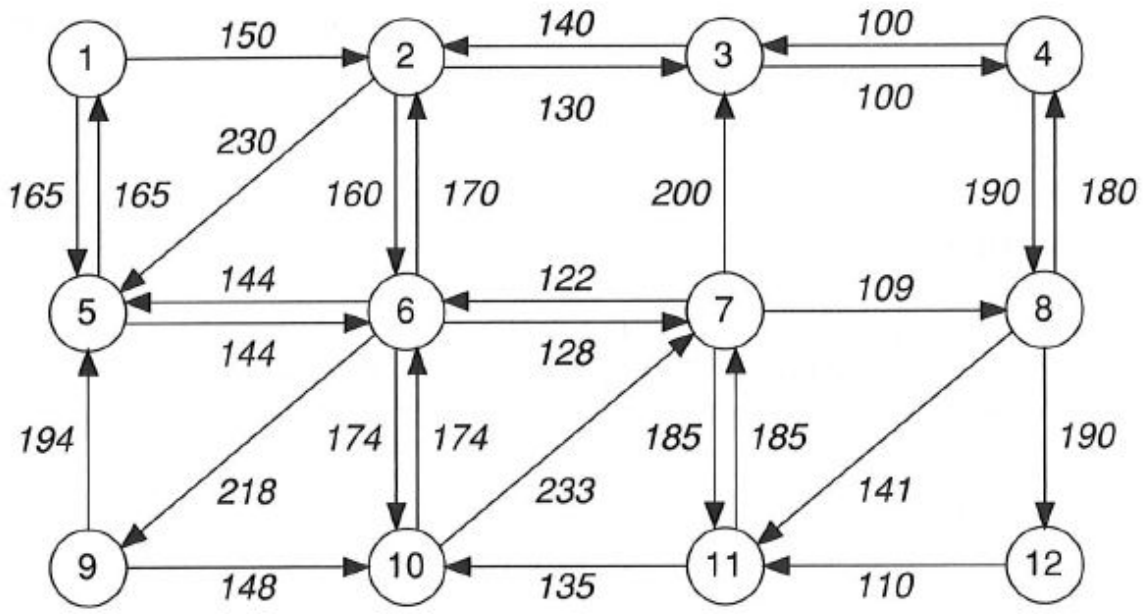

Figure 2: The village streets ([4] p.324)

The problem is from [2] Chap 13.5, and from [4] Chap 15.4.

Modeling Steps: The problem can be formulated as a Min Cost Flow Problem, see mcf-a for the general Min Cost Flow problem. That is, we "simulate" the traversing of an arc as a "flow". Flow=1 means that the arc is traversed once, flow=2 means that the arc is traversed two times, etc. The variables are the number of flows on each arcs ($x_{i,j}$ $i, j \in V$ V is the set of route intersections). The flow on each arc must be 1 (at least on traversal $l_{i,j} \leq x_{i,j}$. We do not want to limit the number of flows on each arc, hence we can set $u_{i,j} = \infty$, however on can show that 2 is enough ($u_{i,j} = 2$). So there is: $l_{i,j} \leq x_{i,j} \leq u_{i,j}$.

Furthermore, the number of entering a node (intersection) must be exactly the number of leaving the node. Hence we have:

$$\sum_j x_{i,j} - \sum_j x_{j,i} = 0 \quad \forall\, i \in V$$

Hence, we have a Min Cost Flow Problem with the following specifications: (1) All lower bounds are 1 ($l_{i,j} = 1$, to make sure that all streets are traversed at least once. All upper bound must be at least 2 ($u_{i,j} \geq 2$). And for completing the formulation (of a Min Cost Flow), $b_i = 0$ for all nodes.

Listing 62: The Complete Model implemented in LPL [5]

```
model guer13_5 "Gritting roads";
  set i,j := [1..12] "Intersection points";
  parameter
    c{i,j} "Steet distances" :=
    /1 2 150, 1 5 165, 2 3 130, 2 5 230, 2 6 160,
     3 2 140, 3 4 100, 4 3 100, 4 8 190, 5 1 165,
     5 6 144, 6 2 170, 6 5 144, 6 7 128, 6 9 218,
     6 10 174,7 3 200, 7 6 122, 7 8 109, 7 11 185,
     8 4 180, 8 11 141,8 12 190,9 5 194, 9 10 148,
     10 6 174,10 7 233,11 7 185,11 10 135,12 11 110/;
    l{i,j}:=1;
    u{i,j}:=999;
  variable x{i,j|c} [l..u] "Nr of passing the street";
  constraint A{i}: sum{j} x[i,j] = sum{j} x[j,i];
  minimize Distance: sum{i,j} c*x;
  Writep(Distance);
  -- draw the graph --
  parameter X{i}:=2*(i-1)%4; Y{i}:=2*Ceil(i/4);
  Draw.Scale(60,60);
  Draw.DefFont('Verdana',12);
  --{c[i,j]} Draw.CArrow('('&1&','&x&',&#8734;)\n'&'c='&c,
  --  X[i],Y[i],X[j],Y[j],.4,.2,0);
  {c[i,j]} Draw.CArrow(c&'',X[i],Y[i],X[j],Y[j],.4,.2,0);
  {c[i,j]|x=2} Draw.CArrow(X[i],Y[i],X[j],Y[j],.2,.2,0);
  {i} Draw.Circle(i&'',X,Y,.2,1,0);
end
```

Solution: The optimal traveled distance is 5990. There are 7 arcs that must be traversed two times: the first time serving the street and the second time without serving. The solution is shown in 2

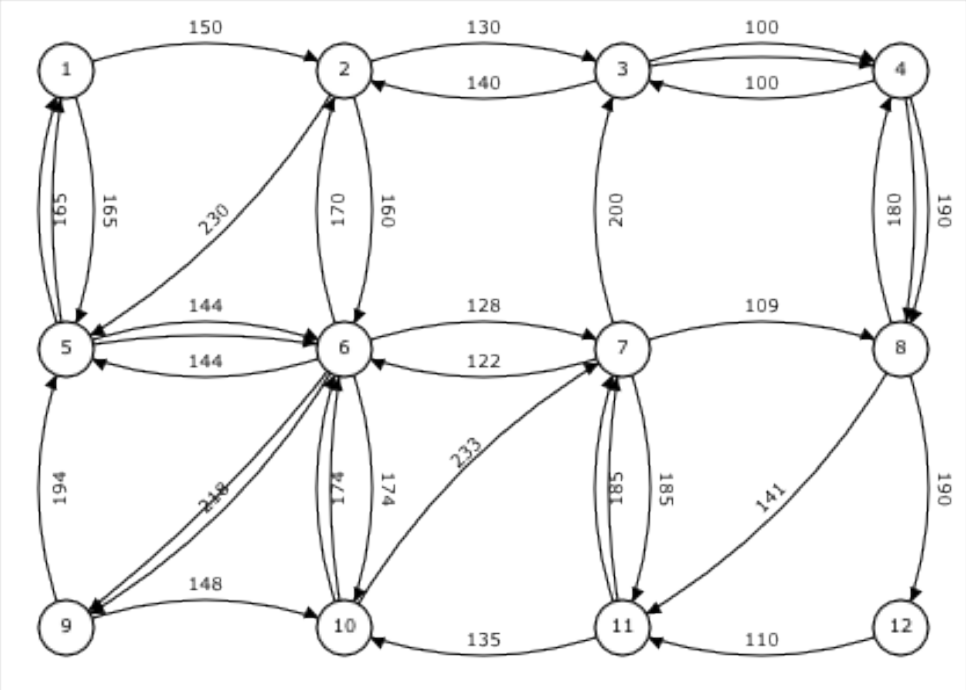

Figure 3: The minimal traversing

63 Location of income tax offices (guer13-6)

—- Run LPL Code , HTML Document –

Problem: The income tax administration is planning to restructure the network of income tax offices in a region. The number of inhabitants of every city and the distances between each pair of cities are known (see table below). The income tax administration has determined that offices should be established in three cities to provide sufficient coverage. Where should these offices be located to minimize the average distance per inhabitant to the closest income tax office? (see [**?**] chap 11.6).

The problem is from [2] chap 13.6. English translation in [4] chap 15.5.

Listing 63: The Complete Model implemented in LPL [5]

```
model guer13_6 "Location of income tax offices";
  set n,m  "Regions";
  parameter
    p := [3] "Number of offices";
    D{n,m}   "Distance";
    H{n}     "Population";
  Read('guer13-6.txt', {n} H);
  Read{n}('%1;3', {m} D);
  binary variable Y{m} "Office in location?";
  variable X{n,m}      "Assign region to office loc";
  constraint
    Affect{n}: sum{m} X = 1;
    NbBureaux: sum{m} Y = p;
    Ajust{n,m}: X <= Y;
  minimize DistPondTot: sum{n,m} H*D*X;
  Write('Objective: %d\n', DistPondTot);
  Write{m|Y}('Office location: %3d | serving:%3d\n', m ,{n|X} n);
end
```

64 Efficiency of hospitals (guer13-7)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 13.7. English translation in [4] chap 15.6.

Listing 64: The Complete Model implemented in LPL [5]

```
model guer13_7b "Efficiency of hospitals";
set
  n := [1..4]  "Hospitals";
  K := [1..4]  "Services indicators";
  L := [1..3]  "Resource indicators";
parameter
  h := 2   "Performance measuring on 2 hospitals";
  s{K,n}  "service indication" :=
    [30.12, 18.54, 20.88, 10.42
     13.54, 14.45,  8.52, 17.74
     13,     7,     8,    26
     79,    55,    47,    50   ];
  r{L,n}  "resource indication" :=
    [90,     87,    51,    66
     38.89, 109.48, 40.43, 48.41
     34,     33,    20,    33   ];
variable
  E      "Efficiency index";
  c{n}   "Coefficients for method DEA";
  sf{K}  "Fictive service indicateurs";
  rf{L}  "Fictive resource indicateurs";
constraint
  DEACoefs: sum{n} c = 1;
  ServFict{K} : sf = sum{n} s*c;
  RessFict{L} : rf = sum{n} r*c;
  ServSup{K}  : sf >= s[K,h];
  RessInf{L}: rf <= r[L,h]*E;
minimize Efficiency: E;
Writep(Efficiency);
end
```

65 MasterMind (guer14-2)

—- Run LPL Code , HTML Document –

Problem: The problem is from [2] chap 14.2.

Listing 65: The Complete Model implemented in LPL [5]

```
model guer14_2 "MasterMind";
set
  c := [Blanc,Bleu,Jaune,Noir,Rouge,Vert];
  p := [1..4]  "Nombre de positions";
binary variable X{p,c}  "=1 si la couleur j est a la position i";
constraint
  -- Une couleur doit etre affectee a chaque position
  UneCoulParPos{p} : sum{c} X = 1;
  --Essai 1: 1 couleur bien placee, 1 couleur mal placee
  E1_bp: X[1,'Rouge']+X[2,'Jaune']+X[3,'Blanc']+X[4,'Vert'] = 1;
  E1_mp: sum{p|p<>1} X[p,'Rouge']+sum{p|p<>2} X[p,'Jaune'] +
   sum{p|p<>3} X[p,'Blanc']+sum{p|p<>4} X[p,'Vert'] = 1;
  --Essai 2: 1 couleur bien placee
  E2_bp: X[1,'Bleu']+X[2,'Vert']+X[3,'Blanc']+X[4,'Bleu'] = 1;
  E2_mp: sum{p|p<>1 and p<>4} X[p,'Bleu']+sum{p|p<>2} X[p,'Vert'] +
    sum{p|p<>3} X[p,'Blanc'] = 0;
  --Essai 3: 1 couleur bien placee, 1 couleur mal placee
  E3_bp: X[1,'Jaune']+X[2,'Noir']+X[3,'Blanc']+X[4,'Noir'] = 1;
  E3_mp: sum{p|p<>1} X[p,'Jaune']+sum{p|p<>2 and p<>4} X[p,'Noir'] +
     sum{p|p<>3} X[p,'Blanc'] = 1;
  --Essai 4: 4 couleurs mal placees
   E4_bp: X[1,'Bleu']+X[2,'Jaune']+X[3,'Rouge']+X[4,'Noir'] = 0;
   E4_mp: sum{p|p<>1} X[p,'Bleu'] + sum{p|p<>2} X[p,'Jaune'] +
     sum{p|p<>3} X[p,'Rouge']+sum{p|p<>4} X[p,'Noir']=4;
solve;
Writep(X);
end
```

66 Probleme de logique: marathon de Paris (guer14-3)

—- Run LPL Code , HTML Document –

Problem: The model is from [2], Chapter 14.3. and [4].

Listing 66: The Complete Model implemented in LPL [5]

```
model guer14_3 "Probleme de logique: marathon de Paris";
set
  n := [1..6] "places";
  c := [Dominique,Ignace,Naren,Olivier,Philippe,Pascal] "Coureurs";
variable X{c,n};
constraint
  UnePlaceParCoureur{c}: sum{n} X = 1;
  UnCoureurParPlace{n}: sum{c} X = 1;
  -- Information a)
  a: X['Olivier',6] = 0;
  -- Information b)
  b: sum{n|n>4} (X['Dominique',n] + X['Ignace',n] + X['Pascal',n]) = 0;
  -- Information c)
  c1: X['Dominique',1] + X['Dominique',2] = 1;
  -- Information d)
  d: sum{n|n<=4} X['Philippe',n] = 1;
  -- Information e)
  e: X['Ignace',2] + X['Ignace',3] = 0;
  -- Information f)
  f1: sum{n|n<=3} X['Naren',n] = 0;
  f2: sum{n|n>=4} X['Pascal',n] = 0;
  -- Information g)
  g: X['Ignace',4] + X['Dominique',4] = 0;
maximize any: X[1];
Writep(X);
end
```

67 Chercheurs dans un Congres (guer14-4)

—- Run LPL Code , HTML Document –

Problem: Probleme de logique consistant a retrouver la nationalite et la date d'expose de 5 chercheurs participant a un congres. The model is from [2], Chapter 14.4.

Listing 67: The Complete Model implemented in LPL [5]

```
model guer14_4 "Chercheurs dans un Congres";
set
  Chercheurs,c:=[Arabinda,Eric,Hitoshi,Michael,Zhicheng];
  n := [Americain,Chinois,Francais,Indien,Japonais];
  Dates,d := ['7','8','9','10','11'];
variable X{c,n,d};
constraint
  Chercheur {c}: sum{n,d} X = 1;
  Nationalite{n}: sum{c,d} X = 1;
  Dat{d}: sum{c,n} X = 1;
  -- Information a)
  a: sum{d} X['Michael','Japonais',d] = 0;
  -- Information b)
  b: sum{d|d<=3} X['Eric','Francais',d] = 1;
  -- Information c)
  c1: sum{n} X['Arabinda',n,'9'] = 1;
  -- Information d)
  d_1: sum{d} (X['Hitoshi','Chinois',d] + X['Michael','Chinois',d]) =
     0;
  d_2: sum{c} X[c,'Chinois','8'] = 1;
  d_3: sum{n,d|d>2} X['Michael',n,d] = 1;
  -- Information e)
  e_1: sum{n} (X['Hitoshi',n,'7'] + X['Hitoshi',n,'11']) = 0;
  e_2: sum{c,d|d>=4} X[c,'Indien',d] = 0;
  e_3: sum{c,d|d<=2} X[c,'Americain',c] = 0;
  e_4: sum{d} (X['Hitoshi','Americain',d] + X['Hitoshi','Indien',d]) =
     0;
solve;
Writep(X);
end
```

68 Grille et jetons (guer14-5)

—- Run LPL Code , HTML Document –

Problem: The model is from [2], Chapter 14.5.

Listing 68: The Complete Model implemented in LPL [5]

```
model guer14_5 "Grille et jetons";
set n,m := [1..4];
parameter   P := 10;
binary variable x{n,m} "(n,m) est occupee par un jeton";
integer NL{n} [0..2] "nombre de jetons en ligne par 2";
integer NC{n} [0..2] "nombre de jetons en colonne par 2";
constraint
  Total: sum{n,m} x = P;
  Lig{n}: sum{m} x = 2*NL[n];
  Col1{m}: sum{n} x = 2*NC[m];
maximize any: x[1];
Writep(x);
end
```

69 Partage de tonneaux entre des neveux (guer14-6)

—- Run LPL Code , HTML Document –

Problem: The model is from [2], Chapter 14.6.

Listing 69: The Complete Model implemented in LPL [5]

```
model guer14_6 "Partage de tonneaux entre des neveux";
set
  n := [1..5]  "Nombre de neveux";
  T := [Plein '3/4' Moite '1/4' Vide]
    "Nombre de type de tonneaux differents (en contenu)";
parameter
  r{T} := [1.00, 0.75, 0.50, 0.25, 0.00]
    "remplissage des tonneaux, plein, 3/4, 1/2, 1/4 et vide";
  c{T} := [10000, 1000, 100, 10, 1]
    "Coefficients pour la contrainte tous differents";
variable p{n} "Valeur du produit scalaire c(_).x(i,_)";
integer variable x{n,T} [1..5]
    "Nombre de tonneaux de type T donnes au neveu n";
constraint
  Neveux{n}    : sum{T} x = 9;
  Tonneaux{T} : sum{n} x = 9;
  Equitable{n}: sum{T} r*x = 4.5;
  TousDiff{n} : sum{T} c*x = p;
  Ordre{n|n>1}: p - p[n-1] >= 1;
minimize any: x[1];
Writep(p,x);
end
```

70 Le probleme des n reines (guer14-7)

—- Run LPL Code , HTML Document –

Problem: The model is from [2], Chapter 14.7..

Listing 70: The Complete Model implemented in LPL [5]

```
model guer14_7 "Le probleme des n reines";
set n,m := [1..100];
binary variable c{n,m}  "une case de l'echiquier vaut 1 si une reine
    est sur cette case, 0 sinon.";
constraint
  Ligne{n}      : sum{m} c <= 1;
  Colonne{m}    : sum{n} c <= 1;
  DiagDL{m}     : sum{n|n>=m} c[n-m+1,n] <= 1;
  DiagDC{n|n>1}: sum{m|m>=n} c[m,m-n+1] <= 1;
  DiagGL{m}     : sum{n|n<=m} c[n,m-n+1] <= 1;
  DiagGC{n|n>1}: sum{m|m>=n} c[m,#n-m+n] <= 1;
maximize NbReines: sum{n,m} c "le nombre de reines";
Writep(NbReines,c);
end
```

References

[1] Artelys. *Artelys Kalis User's Guide Documentation*. Artelys, 2020, Release 13.0.0.

[2] Sevaux M. Guéret Ch., Prins Ch. *Programmation linéaire, 65 problèmes d'optimisation modélisés et résolus avec Visual Xpress*. Eyrolles, 2000.

[3] MatMod. Homepage for Learning Mathematical Modeling : https://matmod.ch.

[4] Heipcke S. *Applications of optimization with Xpress*. dash optimization, 2000,
(Look for "Applications of optimization with Xpress-MP" in Google
A translation of [2].

[5] Hürlimann T. Reference Manual for the LPL Modeling Language, most recent version. https://matmod.ch/lpl/doc/manual.pdf.