AMPL as support for Practical Linear Programming

1 Initiation in AMPL

See AMPL(A Mathematical Programming Language)
`https://en.wikipedia.org/wiki/AMPL` ou
`https://www.artelys.com/fr/composants-numeriques/ampl`.

2 What's AMPL

From AMPL website (`http://www.ampl.com/`) :
AMPL is a comprehensive and powerful algebraic modeling language for linear and nonlinear optimization problems, in discrete or continuous variables. Developed at Bell Laboratories, AMPL lets you use *common notation* and *familiar concepts* to formulate optimization models and examine solutions, while *the computer manages* communication with an appropriate solver. AMPL's flexibility and convenience render it ideal for rapid prototyping and model development, while its speed and control options make it an especially efficient choice for repeated production runs. It can also used as a programming language.

3 AMPL basics

One simple way to use AMPL is to create some text files, containing the details of the optimization problem we want to solve, and pass them to AMPL for obtaining the solution. Usually, 3 text files are needed :

1. the model file (it usually has the `.mod` extension) : it contains the definition of the parameters, the variables, the objective function, the constraints. Parameters and variables can have different types (binary, integer, etc) ; vectors and matrices of variables or parameters can be defined.
2. the data file (it usually has the `.dat` extension) : it contains all information for the definition of an instance of the problem (same model but with different data). In this file, numerical values can be associated to the parameters defined in the model file.
3. the run file (it usually has the `.run` extension) : it contains additional information, such as the files AMPL needs to collect to solve the problem (the name of the model file and the name of the data file) and the name of the solver it needs to use.

4 A simple example

Let us consider a simple optimization problem. We will write the 3 text files described in the previous section, and we will use them in order to solve this problem with AMPL and CPLEX.

We consider a diet problem : we want to minimize the cost of the food we eat while some nutritional constraints (calories, sugar and fat levels, etc) are satisfied. This is the list of food :

1. chocolate candies,
2. chocolate ice cream,
3. coca cola,
4. cheesecake.

Suppose we have a vector x, where x_i represents the quantity of food i. x represents the set of decision variables of the following optimization problem :

$$\min_x 50x_1 + 20x_2 + 30x_3 + 80x_4 \qquad (\text{each } x_i \geq 0 \text{ is weighted with the cost})$$

subject to

$$\begin{array}{rcrcrcrcrr}
400x_1 & + & 200x_2 & + & 150x_3 & + & 500x_4 & \geq & 500 & \text{calories} \\
3x_1 & + & 2x_2 & & & & & \geq & 6 & \text{chocolate} \\
2x_1 & + & 2x_2 & + & 4x_3 & + & 4x_4 & \geq & 10 & \text{sugar} \\
2x_1 & + & 4x_2 & + & x_3 & + & 5x_4 & \geq & 8 & \text{fat}
\end{array}$$

The file `diet.mod`

```
# parameters and variables
param N, integer, >0;  # quantity of food
param c {1..N}, >0;    # vector, the cost of each food
param a {1..N,1..N+1}; # matrix associated to nutritional constraints
var x {1..N}, >= 0;    # the decision variables

# the optimization problem objective
minimize totalcost : sum{i in 1..N} c[i]*x[i];

# the constraints
subject to nutrition {i in 1..N} : sum{j in 1..N} a[i,j]*x[j] >= a[i,N+1];
```

The file `diet.dat`

```
param N := 4;
param : c :=
1 50
2 20
3 30
4 80
;
param : a :=
1 1 400
1 2 200
1 3 150
1 4 500
1 5 500
2 1   3
2 2   2
2 3   0
2 4   0
2 5   6
3 1   2
```

```
3 2   2
3 3   4
3 4   4
3 5  10
4 1   2
4 2   4
4 3   1
4 4   5
4 5   8
;
```

The file `diet.run`

```
model diet.mod;
data  diet.dat;
option solver cplexamp;

solve;

display totalcost;
display x;
```

Execution with AMPL

```
$ ampl diet.run
CPLEX 11.0.0: optimal solution; objective 90
2 dual simplex iterations (0 in phase I)
totalcost = 90

x [*] :=
1  0
2  3
3  1
4  0
;
```

5 FAQ

It is crucial to separate the model from the data, as well to identify parameters and variables.

1. Any parameter is explicitly written in .dat file.
2. Any variable is part of the solution found by the solver.
3. Once the model has been written in the .mod file, you proceed with .dat file where you enter the values of the parameters respecting the type declared in the model.

6 Try AMPL Online

Go to page : `http://ampl.com/try-ampl/start/`

1. Upload some of the files .mod and .dat and click submit.
2. Chose a solver from the list of solvers (by default it is : Minos)
3. Click on Send for launch the solver. Display the solution.

Exercises

Admissible cells : Given a grid rectangular grid $T^{m\times n}$ of boxes called "cells". Some of these cells are « admissible», the others are «non-admissible». Also are given $m+n$ non-negative integers $r_1,\ldots,r_m,c_1,\ldots,c_n$. The goal is to fill in the admissible cells with integers such as :

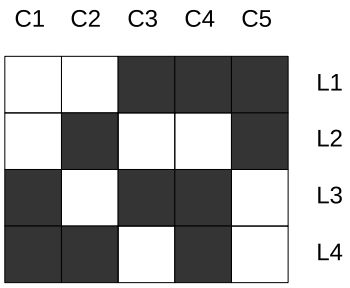
— The sum of the numbers allocated on the admissible cells on row i should be less or equal than r_i ;

— The sum of the numbers allocated on the admissible cells on column j should be less or equal than c_j ;

— The total sum of all these numbers should be maximum.

Example Let $m=4, n=5, r_1=9, r_2=10, r_3=15, r_4=2, c_1=7, c_2=5, c_3=9, c_4=4, c_5=8$, and the set of admissible cells is $A=\{(1,1),(1,2),(2,1),(2,3),(2,4),(3,2),(3,5),(4,3),(4,5)\}$ where the first index corresponds to the row, second index–to the column number.

Instance



Tasks :

1. Give the Linear Programming formulation of the problem.
2. Write an AMPL model (file `.mod`) that uses matrices.
3. Write the `.dat` file containing the previous data.
4. Solve the AMPL model.
5. Write an AMPL model (file `.mod`) that uses set.
6. Write the `.dat` file containing the previous data.
7. Solve the AMPL model.

Hints for solving « Admissible cells »

AMPL language accepts both matrices and sets.

Declaring binary matrix $A^{m\times n}$ associated to the admissible cells :
In your .mod file :

```
param m, integer, >= 0;
param n, integer, >= 0;
param A{1..m, 1..n}, binary;
```

In your .dat file : [1]

```
param m := 5;
param n :=3;
param A :=
1 1 1
1 2 1
1 3 0
2 1 0
2 2 1
2 3 1
3 1 1
3 2 0
3 3 1
4 1 0
4 2 1
4 3 1
5 1 1
5 2 1
5 3 0
;
```

How to use sets for the same purpose ?

In your .mod file :

```
set A_Set within {1..m, 1..n};
```

In your .dat file :

```
set  A_Set :=
1 1
1 2
2 2
2 3
3 1
3 3
4 2
4 3
5 1
5 2
;
```

1. Given data are illustration and do not correspond to the above instance.

Optimal parking : On Dantzig Street cars can be parked on both sides of the street. Mr. Edmonds, who lives at number 1, is organizing a party for around 30 people, who will arrive in 15 cars. The length of the i-th car is λ_i, expressed in meters as follows :

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
λ_i	4	4,5	3	4,1	2,4	4,2	3,7	3,5	3,2	4,5	2,3	3,3	3,8	4,6	3

In order to avoid bothering the neighbors, Mr. Edmonds would like to arrange the parking on both sides of the street so that the length of the street occupied by his friends' cars should be minimum.

1. Give a mathematical programming formulation and solve the problem with AMPL.
2. Add the following constraints.
   (a) The length of the cars parked on the left side should be less than 20 meters.
   (b) On exactly one of the street sides the cars should not occupy more than 20m ?
   (c) Cars longer than 4 m should be parked on left side.
   (d) If the length of the left side is bigger than 10 meters, the length of the right side should be less than 13 meters.

Network Design 1 : The graph below represents a military telecom network. It contains eleven sites (nodes) connected by bidirectional links for data transfer. The reliability specifications require that sites number 10 and 11 continue to communicate even in case of destruction of whatever three other sites. Check if the network meets this requirement.

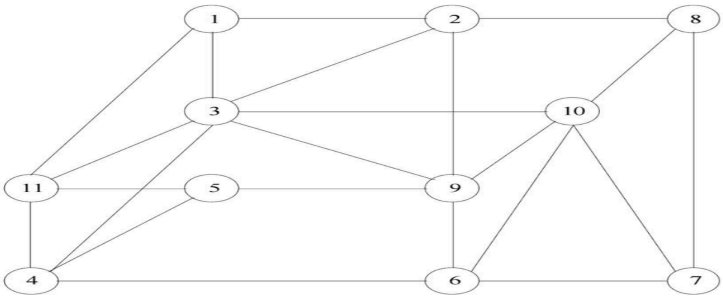


FIGURE 1 – A military telecom network.

Network Design 2 : Orange is the unique owner and handler of the telecom network in the figure below.

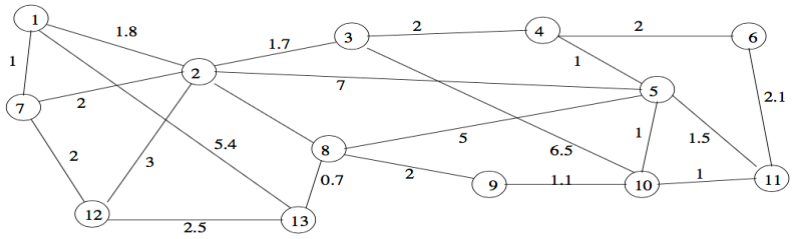


FIGURE 2 – Orange telecom network.

The costs on the links are proportional to the distances $d(i, j)$ between the nodes, expressed in units of 10km. Because of anti-trust regulations, Orange must delegate to SFR and Bouygtel two subnetworks each having at least two nodes (with Orange handling the third part). Orange therefore needs to design a backbone network to connect the three subnetworks. Transforming an existing link into a backbone link costs $c = 25$ euros/km.

1. Formulate a mathematical program to minimize the cost of implementing a backbone connecting the three subnetworks, and solve it with AMPL.
2. How does the solution change if Orange decides to partition its network in 4 subnetworks instead of 3 ?