CIND-221: Taller 1

Felipe Osorio

f.osoriosalgado@uandresbello.edu

Facultad de Ingeniería, UNAB

AMPL (A Mathematical Programming Language):

AMPL es un poderoso lenguaje de modelado que permite expresar problemas de optimización complejos y de gran escala.

Una característica notable de AMPL es su capacidad para soportar diversos solvers de optimización, tales como: Gurobi, CPLEX, COPT, NVIDIA cuOpt, MINOS, Knitro, LGO y muchos otros.

La sintaxis de AMPL es similar a la notación matemática, lo que hace muy apropiado para resolver problemas de optimización de forma muy precisa.

- Sitio web de AMPL: `https://ampl.com/`
- The (Original) AMPL Book: `https://ampl.com/resources/books/ampl-book/`

AMPL: A Mathematical Programming Language

La estructura de un modelo en AMPL se divide en las siguientes partes:

- Datos, parámetros y variables.
- Función objetivo.
- Restricciones para el problema.

La información se suele incluir en los siguientes archivos:

- Modelo (`.mod`): describe parámetros, variables, función objetivo y restricciones.
- Datos (`.dat`): valores para los parámetros y datos del modelo.

Adicionalmente, se puede incluir un Script (`.run`) que permite ejecutar el modelo y desplegar los resultados del proceso de optimización.

Usando AMPL

Ejemplo (Problema de Programación Lineal):

Considere una panadería donde se produce 2 tipos de pan: hogazas pan integral y de pan corriente. Se dispone de 300 kg de harina y 180 horas de uso de horno.

El pan integral utiliza 1.5 kg de harina, 2 hrs de horno y proporciona una ganancia de 5 UM por pieza. Por otro lado, el pan corriente usa 2.0 kg de harina, una hora de horneado y entrega una ganancia de 4 UM.

Determine la cantidad de hogazas que se deben hornear para maximizar su ganancia.

Usando AMPL

Archivo **panaderia.mod**

```
# variables
var x >= 0; # pan integral
var y >= 0; # pan corriente

# función objetivo
maximize z: 5 * x + 4 * y;

# restricciones
subject to harina: 1.5 * x + 2 * y <= 300;
subject to horno: 2 * x + y <= 180;

```

Usando AMPL

Ejecutando la optimización en la consola de AMPL:

```
ampl: model panaderia.mod; # define modelo
ampl: option solver cplex; # selecciona 'solver'
ampl: solve;
CPLEX 22.1.2: optimal solution; objective 648
2 simplex iterations
ampl: display x,y,z;
x = 24
y = 132
z = 648

```

Ejemplo (Problema de asignación):

Un departamento necesitar asignar un grupo de personas en un grupo de oficinas, y que cada persona debe ser asignada a una oficina, tal que, toda oficina debe ser ocupada sólo por una persona.

Suponga que se conoce el costo de asignación de cada persona a cada oficina, el que es dado por la siguiente matriz

	C118	C138	C140	C246	C250	C251	D237	D239	D241	M233	M239
Coullard	6	9	8	7	11	10	4	5	3	2	1
Daskin	11	8	7	6	9	10	1	5	4	2	3
Hazen	9	10	11	1	5	6	2	7	8	3	4
Hopp	11	9	8	10	6	5	1	7	4	2	3
Iravani	3	2	8	9	10	11	1	5	4	6	7
Linetsky	11	9	10	5	3	4	6	7	8	1	2
Mehrotra	6	11	10	9	8	7	1	2	5	4	3
Nelson	11	5	4	6	7	8	1	9	10	2	3
Smilowitz	11	9	10	8	6	5	7	3	4	1	2
Tamhane	5	6	9	8	4	3	7	10	11	2	1
White	11	9	8	4	6	5	3	10	7	2	1

Usando AMPL

Archivo **oficina.mod**

```
1  set ORIG;    # orígenes
2  set DEST;    # destinos
3
4  param C {ORIG,DEST} >= 0;  # costos de envío por unidad
5  var X {ORIG,DEST} >= 0;    # unidades a enviar
6
7  minimize Total_Cost:       # función objetivo
8     sum {i in ORIG, j in DEST} C[i,j] * X[i,j];
9
10 subject to Oficinas {j in DEST}:
11    sum {i in ORIG} X[i,j] = 1;
12
13 subject to Personas {i in ORIG}:
14    sum {j in DEST} X[i,j] <= 1;
15
```

Usando AMPL

Archivo **oficina.dat**

```
set ORIG := Coullard Daskin Hazen Hopp Iravani Linetsky
            Mehrotra Nelson Smilowitz Tamhane White ;

set DEST := C118 C138 C140 C246 C250 C251 D237 D239 D241 M233 M239;

param C:
           C118 C138 C140 C246 C250 C251 D237 D239 D241 M233 M239 :=
 Coullard    6    9    8    7   11   10    4    5    3    2    1
 Daskin     11    8    7    6    9   10    1    5    4    2    3
 Hazen       9   10   11    1    5    6    2    7    8    3    4
 Hopp       11    9    8   10    6    5    1    7    4    2    3
 Iravani     3    2    8    9   10   11    1    5    4    6    7
 Linetsky   11    9   10    5    3    4    6    7    8    1    2
 Mehrotra    6   11   10    9    8    7    1    2    5    4    3
 Nelson     11    5    4    6    7    8    1    9   10    2    3
 Smilowitz  11    9   10    8    6    5    7    3    4    1    2
 Tamhane     5    6    9    8    4    3    7   10   11    2    1
 White      11    9    8    4    6    5    3   10    7    2    1 ;

```

Usando AMPL

Ejecutando la optimización en la consola de AMPL:

```
ampl: reset; # eliminar caché
ampl: model oficinas.mod;
ampl: data oficinas.dat;
ampl: option solver cplex; # selecciona 'solver'
ampl: solve;
CPLEX 22.1.2: optimal solution; objective 28
10 simplex iterations

```

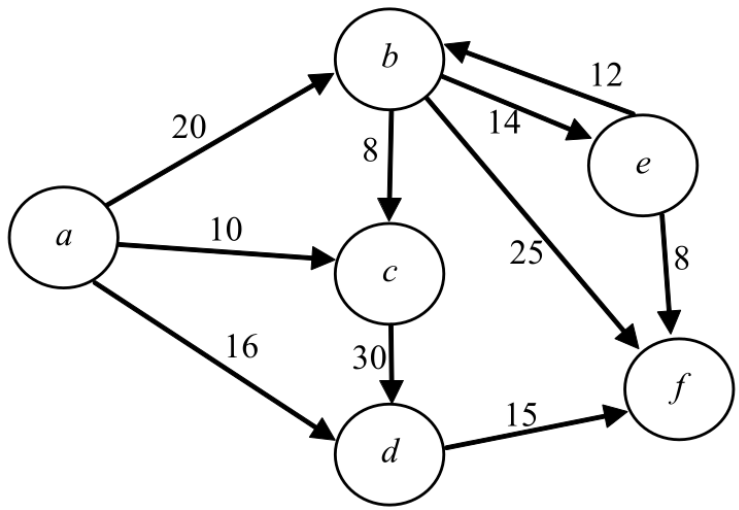
Usando AMPL

Salida:

```
ampl: display X;
X [*,*]
:          C118 C138 C140 C246 C250 C251 D237 D239 D241 M233 M239
     :=
Coullard     0    0    0    0    0    0    0    0    1    0    0
Daskin       0    0    0    0    0    0    1    0    0    0    0
Hazen        0    0    0    1    0    0    0    0    0    0    0
Hopp         0    0    0    0    0    1    0    0    0    0    0
Iravani      0    1    0    0    0    0    0    0    0    0    0
Linetsky     0    0    0    0    1    0    0    0    0    0    0
Mehrotra     0    0    0    0    0    0    0    1    0    0    0
Nelson       0    0    1    0    0    0    0    0    0    0    0
Smilowitz    0    0    0    0    0    0    0    0    0    1    0
Tamhane      1    0    0    0    0    0    0    0    0    0    0
White        0    0    0    0    0    0    0    0    0    0    1
;

```

Ejemplo (Problema de la ruta más corta):



Se desea viajar desde el nodo a al nodo f.

Usando AMPL

Archivo **rutas.mod**

```
set INTER; # intersecciones (nodos)

param entr symbolic in INTER;            # Entranda a la red
param exit symbolic in INTER, <> entr;   # Salida desde la red

set ROADS within (INTER diff {exit}) cross (INTER diff {entr});

param time {ROADS} >=0;          # tiempos (costos) de viajes
var Use {(i,j) in ROADS} >=0;    # 1 ssi (i,j) en la ruta más corta

minimize Total_Time: sum{(i,j) in ROADS} time[i,j] * Use[i,j];

subject to Start: sum {(entr,j) in ROADS} Use[entr,j] = 1;

subject to Balance {k in INTER diff{entr,exit}}:
  sum {(i,k) in ROADS} Use[i,k] = sum {(k,j) in ROADS} Use[k,j];

```

Usando AMPL

Archivo **rutas.dat**

```
set INTER := a b c d e f;

param entr := a;
param exit := f;

param: ROADS: time:=
a b 20,
a c 10,
a d 16,
b c 8,
b e 14
b f 25,
c d 30,
d f 15,
e b 12,
e f 8;

```

Usando AMPL

Ejecutando la optimización en la consola de AMPL:

```
ampl: reset;
ampl: model rutas.mod;
ampl: data rutas.dat;
ampl: option solver cplex;
ampl: solve;
CPLEX 22.1.2: optimal solution; objective 31
0 simplex iterations

```

Usando AMPL

Salida:

```
ampl: display Total_Time;
Total_Time = 31

ampl: display Use;
Use :=
a b   0
a c   0
a d   1
b c   0
b e   0
b f   0
c d   0
d f   1
e b   0
e f   0
;

```