Programación Entera - Parte 2

Investigación Operativa

Universidad de San Andrés

En la clase anterior:

- Vimos problemas con variables si/no
- Ahora aparecen problemas con variables binarias y continuas juntas
- Ejemplo típico: decidir si se opera una planta y cuanto produce

Ejemplo: Planificación de producción

Una empresa opera con dos turnos: diurno y nocturno. El costo de poner en marcha la planta es:

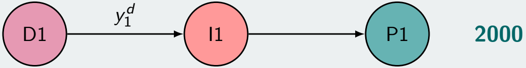
- Turno diurno: $8.000
- Turno nocturno: $4.500

Demanda para los próximos dos días:

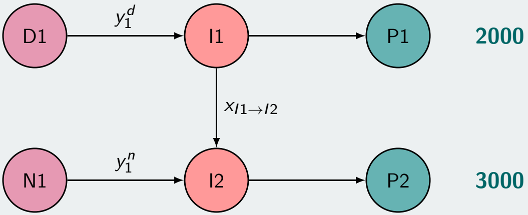
- Dia 1: 2000 unidades (diurno)
- Noche 1: 3000 unidades (nocturno)
- Dia 2: 2000 unidades (diurno)
- Noche 2: 3000 unidades (nocturno)

Costo de almacenaje: $1 por unidad por día. Se busca minimizar el costo total cumpliendo demanda.

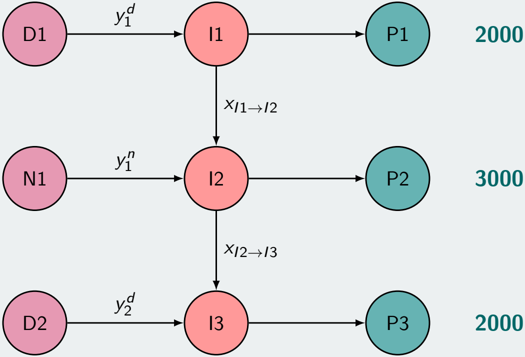
Visualización del problema: Turno D1



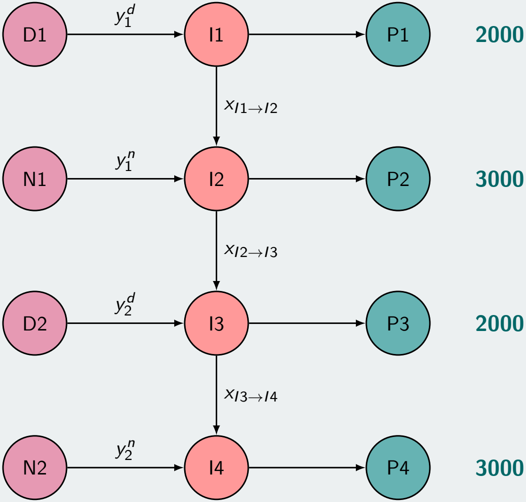
Visualización del problema: Turno N1



Visualización del problema: Turno D2



Visualización del problema: Turno N2



Variables y función objetivo

Variables:

- y_i^d: 1 si se produce en el turno diurno i, 0 si no
- y_i^n: 1 si se produce en el turno nocturno i, 0 si no
- $x_{Ii \to I(i+1)}$: cantidad almacenada entre periodos

Función objetivo:

$$\text{Min } Z = 8000(y_1^d + y_2^d) + 4500(y_1^n + y_2^n) + \sum_{i=1}^{3} x_{Ii \to I(i+1)}$$

Restricciones

- $y_i^d, y_i^n \in \{0, 1\}$
- $x_{Ii \to I(i+1)} \geq 0$
- Conservación de flujo en cada periodo

$$
\begin{aligned}
y_1^d - x_{I1 \to I2} &= x_{I1 \to P_1} \\
y_1^n + x_{I1 \to I2} - x_{I2 \to I3} &= x_{I2 \to P_2} \\
y_2^d + x_{I2 \to I3} - x_{I3 \to I4} &= x_{I3 \to P_3} \\
y_2^n + x_{I3 \to I4} &= x_{I4 \to P_4}
\end{aligned}
$$

Modelo en PICOS: Definicion de variables y datos

```python
import picos
import numpy as np

# Datos
costo_diurno = 8000
costo_nocturno = 4500
demanda = [2000, 3000, 2000, 3000]  # D1, N1, D2, N2
costo_almacenaje = 1

# Variables
P = picos.Problem()

y_d = picos.BinaryVariable('y_d', 2)  # diurno (dia 1 y dia 2)
y_n = picos.BinaryVariable('y_n', 2)  # nocturno (noche 1 y noche 2)
x_d = picos.RealVariable('x_d', 2, lower=0)
x_n = picos.RealVariable('x_n', 2, lower=0)
s = picos.RealVariable('s', 3, lower=0)
```

Modelo en PICOS: Objetivo y restricciones

```python
# Objetivo: costos fijos de produccion + almacenaje
P.set_objective('min',
    costo_diurno * picos.sum(y_d) +
    costo_nocturno * picos.sum(y_n) +
    costo_almacenaje * picos.sum(s)
)

tipo_M = 10000
for i in range(2):
    P.add_constraint(x_d[i] <= tipo_M * y_d[i])
    P.add_constraint(x_n[i] <= tipo_M * y_n[i])

# Balance de inventario
P.add_constraint(x_d[0] - demanda[0] == s[0])
P.add_constraint(x_n[0] + s[0] - demanda[1] == s[1])
P.add_constraint(x_d[1] + s[1] - demanda[2] == s[2])
P.add_constraint(x_n[1] + s[2] - demanda[3] == 0)
```

Modelo en PICOS: Resolucion y resultados

```
P.solve(solver='glpk')
print('Produccion diurna:', np.round(x_d.value, 2))
print('Produccion nocturna:', np.round(x_n.value, 2))
print('Turnos diurnos activados:', [int(round(y_d[i].value)) for i in range(2)])
print('Turnos nocturnos activados:', [int(round(y_n[i].value)) for i in range(2)])
print('Inventario final en cada periodo:', np.round(s.value, 2))
print('Costo total:', round(P.value, 2))
```

Salida

```
Produccion diurna:
[[2000.]
[   0.]]
Produccion nocturna:
[[5000.]
[3000.]]
Turnos diurnos activados:
[1, 0]
Turnos nocturnos activados:
[1, 1]
Inventario final en cada periodo:
[[   0.]
[2000.]
[   0.]]
Costo total: 19000.0
```

Ejemplo: Planificación eléctrica

Una empresa de energía debe satisfacer una demanda creciente durante 5 años. Hay 4 tipos de plantas con distintas capacidades y costos de construcción y operación. Se debe decidir en que año construir cada planta y cuando operarla para minimizar el costo total.

Datos del problema

Planta	Capacidad (millones kWh)	Costo construccion (M$)	Costo operativo anual (M$)
1	70	20	1.5
2	50	16	0.8
3	60	18	1.3
4	40	14	0.6

Demanda: 80, 100, 120, 140, 160 millones kWh por año.

Variables y modelo

Variables:

- x_{ij}: energía generada por planta i en año j
- α_{ij}: 1 si la planta i esta operativa en año j, 0 si no
- y_i: 1 si la planta i se construye

Función objetivo:

$$\text{Min } Z = \sum_{i=1}^{4} c_i y_i + \sum_{i=1}^{4} \sum_{j=1}^{5} o_i \alpha_{ij}$$

Restricciones

1. Demanda anual: $\sum_i x_{ij} \geq d_j$ para todo j
2. Capacidad de planta: $x_{ij} \leq cap_i \alpha_{ij}$
3. Operar solo si se construyo: $\alpha_{ij} \leq y_i$
4. Permanencia: una vez operativa, no se puede apagar: $\alpha_{i,j+1} \geq \alpha_{ij}$

Modelo en PICOS: Definicion de datos y variables

```python
import picos

demand = picos.Constant('demanda',  [80, 100, 120, 140, 160])
limits  = picos.Constant('cap',     [70, 50, 60, 40])
c       = picos.Constant('c',       [20, 16, 18, 14])
o       = picos.Constant('o',       [1.5, 0.8, 1.3, 0.6])

T, P = 5, 4  # anios, plantas
x = picos.RealVariable('x',  (P, T), lower=0)
alfa = picos.BinaryVariable('alfa', (P, T))
y = picos.BinaryVariable('y', P)

p = picos.Problem()
```

Modelo en PICOS: Restricciones

```python
# Demanda anual
for j in range(T):
    p.add_constraint(picos.sum(x[:, j]) >= demand[j])

# Capacidad de planta
for i in range(P):
    p.add_constraint(picos.sum(x[i, :]) <= 1e4*y[i])
    for j in range(T):
        p.add_constraint(x[i, j] <= 1e4*alfa[i, j])
        p.add_constraint(x[i, j] - limits[i] <= 1e4*(1-alfa[i, j]))

# Permanencia
for i in range(P):
    for j in range(T-1):
        p.add_constraint(alfa[i, j+1] >= alfa[i, j])

# Operacion implica construccion
for i in range(P):
    for j in range(T):
        p.add_constraint(alfa[i, j] <= y[i])
```

Modelo en PICOS: Objetivo y resolucion

```
# Objetivo
p.set_objective('min', c[0]*y[0] + c[1]*y[1] + c[2]*y[2] + c[3]*y[3] +
                     o[0]*picos.sum(alfa[0, :]) + o[1]*picos.sum(alfa[1, :]) +
                     o[2]*picos.sum(alfa[2, :]) + o[3]*picos.sum(alfa[3, :]))

# Resolver
p.options.verbosity = 1
p.solve(solver='glpk')

print(x)          # energia generada
print(p.value)    # costo total minimo
```

Salida esperada

```
[ 7.00e+01  5.00e+01  7.00e+01  7.00e+01
   7.00e+01]
[ 1.00e+01  5.00e+01  5.00e+01  5.00e+01
   5.00e+01]
[ 0.00e+00  0.00e+00  0.00e+00  0.00e+00
   0.00e+00]
[ 0.00e+00  0.00e+00  0.00e+00  2.00e+01
   4.00e+01]
62.7
```

Terminamos

¿Dudas?

¿Consultas?