Ivestigación de operaciones

September 16, 2021

1 Problema de asignación de recursos

Una fábrica produce 4 productos diferentes. El objetivo es determinar la cantidad de producción diaria que maximiza el beneficio para cada producto, teniendo en cuenta las siguientes condiciones.

- El precio por unidad de producto es de 20, 12, 40 y 25 soles para el primer, segundo, tercer y cuarto producto, respectivamente.
- Debido a las limitaciones de mano de obra, el número total de unidades producidas por día no puede superar las cincuenta.
- Para cada unidad del primer producto se consumen tres unidades de la materia prima A. Cada unidad del segundo producto requiere dos unidades de la materia prima A y una unidad de la materia prima B. Cada unidad del tercer producto necesita una unidad de A y dos unidades de B. Por último, cada unidad del cuarto producto requiere tres unidades de B.
- Debido a las limitaciones de transporte y almacenamiento, la fábrica puede consumir hasta cien unidades de la materia prima A y noventa unidades de B al día.

1.1 Planteamiento

Variables de decisiòn

$x_1 =:$ *Cantidad diaria producida del producto 1*

$x_2 =:$ *Cantidad diaria producida del producto 2*

$x_3 =:$ *Cantidad diaria producida del producto 3*

$x_4 =:$ *Cantidad diaria producida del producto 4*

Función objetivo:

Maximizar la cantidad diaria producida para obtener la mejor utilidad

(max) $Z = 20x_1 + 12x_2 + 40x_3 + 25x_4$

Restricciones:

$x_1 + x_2 + x_3 + x_4 \leq 50$ (maximo de unidades producidas por la mano de obra)

$3x_1 + 2x_2 + x_3 \leq 100$ (unidades consumidas por material A)

$x_2 + 2x_3 + 3x_4 \leq 90$ (unidades consumidas por material B)

$x_1 + x_2 + x_3 + x_4 \geq 0$

1.2 Resoluciòn

Para solucionar este proble he obtado por utilizar una libreria llamada PuLP que provee métodos que nos ayudan a resolver problemas de programación lineal.

[4]:
```
from pulp import  LpProblem , LpVariable, LpStatus , LpMaximize, lpSum

from utils  import solve_display_model
```

[12]:
```
# modelo

def solve_display_model(model: LpProblem):
    model.solve()
    print(f"status: {model.status}, {LpStatus[model.status]}")
    print(f"Función objetivo: {model.objective.value()}")

    print("variables:")
    for var in model.variables():
        print(f"{var.name}: {var.value()}")

    print("Restricciones: ")
    for name, constraint in model.constraints.items():
        print(f"{name}: {constraint.value()}")


def solve1():
    model = LpProblem(name="Resource_problem" , sense=LpMaximize)

    # variables de decisión

    x1 =  LpVariable("x1" , lowBound=0)
    x2 = LpVariable("x2" , lowBound=0)
    x3 = LpVariable("x3" , lowBound=0)
    x4 = LpVariable("x4" , lowBound=0)

    # funciòn objetivo

    obj_func = 20 * x1 + 12 * x2 +  40 *  x3 + 25 *  x4

    # restricciones

    man_power =( x1 +  x2 + x3 + x4 <= 50 , "man_power")
    a_material =  (3 * x1 + 2 * x2 + x3 <= 100 , "a_material")
    b_material =  (x2  + 2 * x3 +  3 * x4 <= 90  ,"b_material")

    model+=man_power
    model+=a_material
    model+=b_material
```

```
    model+=obj_func

    solve_display_model(model)

solve1()
```

```
status: 1, Optimal
Función objetivo: 1900.0
variables:
x1: 5.0
x2: 0.0
x3: 45.0
x4: 0.0
Restricciones:
man_power: 0.0
a_material: -40.0
b_material: 0.0
```

1.3 Interpretaciòn

Según el planteamiento del modelo, el objetivo fue encontrar la cantidad de productos que se deberia vender por dìa para encontrar la utilidad máxima y según el resultado se puede decir que:

- La utilidad máxima que podriamos obtener en un dìa es de 1900 soles
- No se recomieda fabricar ningún producto de x_2 ni x_4 dadas las restricciones
- El producto que genera más utilidad es el producto x_3 y para sacar el maximo provecho se deberia fabricas 45 unidades

Segùn el resultado de las holguras se puede decir que:

- La capacidad máxima de la mano de obra es de 50 unidades no sobra, ni falta.

$5 + 0 + 45 + 0 \leq 50$ $50 \leq 50$

- No se utiliza toda la materia prima A, y sobra 40 unidades

$3x_1 + 2x_2 + x_3 \leq 100$ $15 + 0 + 45 \leq 100$ $60 \leq 100$

- Y tambien vemos que se logra consumir el màximo consumo de la materia prima b

$x_2 + 2x_3 + 3x_4 \leq 90$ $0 + 2(45) + 3(0) \leq 90$ $90 \leq 90$

1.4 Solución con tora


LINEAR PROGRAMMING

TORA Optimization System, Windows®-version 1.00
Copyright © 2000-2002 Hamdy A. Taha. All Rights Reserved
Thursday, September 16, 2021 10:40

LINEAR PROGRAMMING OUTPUT SUMMARY

Title: resource_problem
Final Iteration No.: 3
Objective Value (Max) =1900.00

Variable	Value	Obj Coeff	Obj Val Contrib
x1: x1	5.00	20.00	100.00
x2: x2	0.00	12.00	0.00
x3: x3	45.00	40.00	1800.00
x4: x4	0.00	25.00	0.00

Constraint	RHS	Slack-/Surplus+
1 (<)	100.00	40.00-
2 (<)	90.00	0.00
3 (<)	50.00	0.00

Sensitivity Analysis

Variable	Current Obj Coeff	Min Obj Coeff	Max Obj Coeff	Reduced Cost
x1: x1	20.00	0.00	40.00	0.00
x2: x2	12.00	-infinity	30.00	18.00
x3: x3	40.00	23.33	infinity	0.00
x4: x4	25.00	-infinity	50.00	25.00

Constraint	Current RHS	Min RHS	Max RHS	Dual Price
1 (<)	100.00	60.00	infinity	0.00
2 (<)	90.00	50.00	100.00	10.00
3 (<)	50.00	45.00	63.33	20.00

Esta funciòn ayudara a poder reutilizar el modelo y poder probar con distintos coeficientes

[49]:
```
def solve_generic(a, b , c , d ):
    model = LpProblem(name="Resource_problem" , sense=LpMaximize)

    # variables de decisión

    x1 =  LpVariable("x1" , lowBound=0)
    x2 = LpVariable("x2" , lowBound=0)
    x3 = LpVariable("x3" , lowBound=0)
    x4 = LpVariable("x4" , lowBound=0)

    # ahora x1 tiene el coeficiente 40

    obj_func = a * x1 + b * x2 +  c *  x3 + d *  x4

    # restricciones
```

```
    man_power =( x1 +  x2 + x3 + x4 <= 50 , "man_power")
    a_material =  (3 * x1 + 2 * x2 + x3 <= 100 , "a_material")
    b_material =  (x2  + 2 * x3 +  3 * x4 <= 90  ,"b_material")

    model+=man_power
    model+=a_material
    model+=b_material
    model+=obj_func

    solve_display_model(model)
```

Esta es la soluciòn con el software tora el cual indica el mismo resultado que el que obtenimos con PuLP,pero el software tora nos da el análisis de la sensibilidad con el cual podemos deducir que:

1.4.1 variables

- x_1: El coeficiente es cero y la reducciòn de costo es cero pero el maximo coeficiente es 40 osea podramos interpretar que cuando el coeficiente de x_1 sea 40 podriamos encontrar otra solución optima.

[50]:
```
solve_generic(40 , 12 , 40 ,25)
```

```
status: 1, Optimal
Función objetivo: 2000.0
variables:
x1: 5.0
x2: 0.0
x3: 45.0
x4: 0.0
Restricciones:
man_power: 0.0
a_material: -40.0
b_material: 0.0
```

La soluciòn es la misma excepto por el resultado de la funciòn objetivo, ahora se tiene 2000 de beneficio , indicando que podriamos aumentar ese nùmero, ahora serìa cuestiòn de determinar que tanto es la posibilidad de poder elevar ese precio.

- x_2: El costo reducido ahora es 18 indicando que ser podrìa elevar en 18 al coeficiente actual para obtener el mismo resultado.

[51]:
```
solve_generic(20 , (12 + 18) , 40 ,25)
```

```
status: 1, Optimal
Función objetivo: 1900.0
variables:
x1: 5.0
x2: 0.0
x3: 45.0
```

```
x4: 0.0
Restricciones:
man_power: 0.0
a_material: -40.0
b_material: 0.0
```

1.4.2 Constraints

- **Materia prima A 1 (<):** Esta restriccion no es vinculante y siempre van a sobrar 40 unidades
- **Materia prima B 2 (<):** El `Dual price` nos indica que si se tuviera 1 unidad màs de la materia prima B se produciria 10 unidades de màs.
- **mano de obra 3 (<):** El `Dual price` nos indica que si se tuviera 1 trabajador màs se lograria producir 20 unidades màs pero debido a las otras restriccciones lo maximo que podriamos llegar a producir es 63