Universidad Tecnológica de Pereira

PROBLEMA DEL TRANSPORTE

PRESENTADO POR:

ALEJANDRO VILLEGAS RAMIREZ

AL PROFESOR:

ANDRES FELIPE MARTINEZ CORREA

PEREIRA

JULIO DE 2021

UNIVERSIDAD TECNOLOGICA DE PEREIRA

Universidad Tecnológica de Pereira

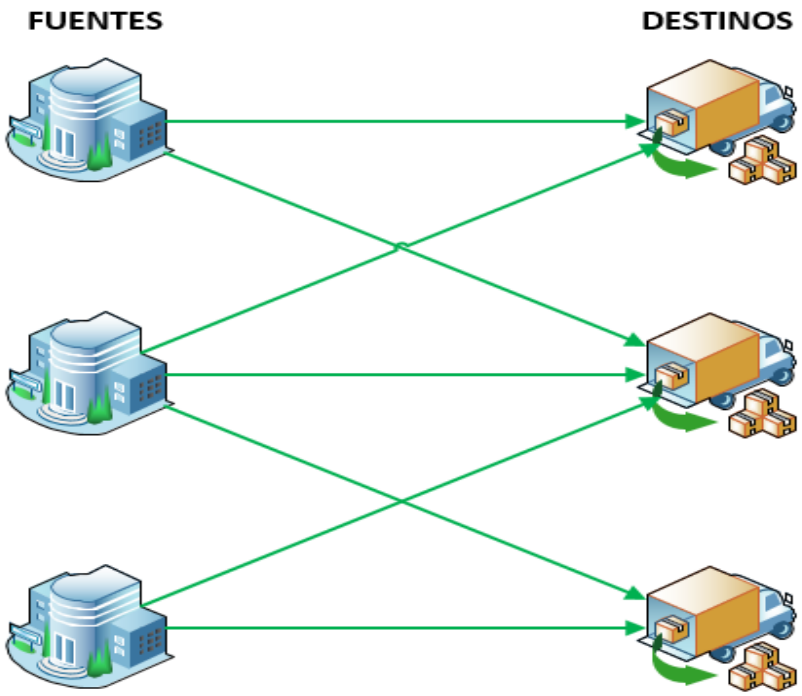
El problema del transporte.

El problema del transporte o distribución es un problema de redes especial en programación lineal que se funda en la necesidad de llevar unidades de un punto específico llamado fuente u origen hacia otro punto específico llamado destino. Los principales objetivos de un modelo de transporte son la satisfacción de todos los requerimientos establecidos por los destinos, y claro está, la minimización de los costos relacionados con el plan determinado por las rutas escogidas.

El contexto en el que se aplica el modelo de transporte es amplio y puede generar soluciones atinentes al área de operaciones, inventario y asignación de elementos.

El procedimiento de resolución de un modelo de transporte se puede llevar a cabo mediante programación lineal común, sin embargo, su estructura permite la creación de múltiples alternativas de solución tales como la estructura de asignación o los métodos heurísticos más populares como Vogel, Esquina Noroeste o Mínimos Costos.

Los problemas de transporte o distribución son uno de los más aplicados en la economía actual, dejando como es de prever múltiples casos de éxito a escala global que estimulan la aprehensión de estos.



Fuete original de la información: https://www.ingenieriaindustrialonline.com/investigacion-de-operaciones/problema-del-transporte-o-distribucion/

Universidad Tecnológica de Pereira

Explicación del funcionamiento del código.

Para entender mejor el funcionamiento del código, hay que poner en contexto al lector sobre el problema que vamos a tratar. En este ejemplo que vamos a presentar a continuación se tiene un problema de programación lineal básico, que se puede solucionar por el método del transporte, donde tenemos 3 plantas que distribuyen a 3 sitios. Para nuestro caso la oferta es igual a la demanda, entonces este problema esta balanceado.

Setting up a Transportation problem

- To set up a transportation problem let consider the following example

Example
A concrete company transports concrete from three plants, 1, 2 and 3, to three construction sites, A, B and C.

The requirements of the sites (tons/week):

Construction Sites	Demand (Requirement)
A	200
B	200
C	300

Supply capacity (tons/week):

Plant	Supply (Capacity)
1	300
2	300
3	100

Transportation cost for 1 ton of each concrete

From \ To	A	B	C
1	4	3	8
2	7	5	9
3	4	5	5

From \ To	A	B	C	Supply (Capacity)
1	4	3	8	300
2	7	5	9	300
3	4	5	5	100
Demand (Requirement)	200	200	300	

Universidad Tecnológica de Pereira

Siguiendo el modelo de solución de un problema lineal, como en todos estos, tenemos que hallar las restricciones, y la función objetivo, en nuestro caso:

Restricciones:

$$
\begin{aligned}
x_{1A} + x_{1B} + x_{1C} &= 300 \\
x_{2A} + x_{2B} + x_{2C} &= 300 \\
x_{3A} + x_{3B} + x_{3C} &= 100 \\
\\
x_{1A} + x_{2A} + x_{3A} &= 200 \\
x_{1B} + x_{2B} + x_{3B} &= 200 \\
x_{1C} + x_{2C} + x_{3C} &= 300
\end{aligned}
$$

Función objetivo:

$$\text{Min } Z = 4x_{1A} + 3x_{1B} + 8x_{1C} + 7x_{2A} + 5x_{2B} + 9x_{2C} + 4x_{3A} + 5x_{3B} + 5x_{3C}$$

Cuando ya tenemos el modelo matemático de esta forma, se puede trabajar el problema con Python.

Universidad Tecnológica de Pereira

Para esta solución vamos a utilizar la librería Pulp, y de esta vamos a llamar funciones que nos ayudan a solucionar el problema

```
#importamos punciones de la librería pulp
from pulp import LpProblem, LpVariable, LpStatus, LpMinimize, GLPK
, value
```

Las tres plantas representan las demandas y las ofertas que usan en nuestro problema, ademas creamos variables para crear variables de decisión n y m dependen de la demanda y la oferta del ejercicio

```
#Datos para optimizar el problema
m = 3 # Puntos de oferta
m = 3 # Puntos de demanda
a = range(1, M + 1)
a1 = range(M)
b = range(1, N+1)
b1 = range(N)
```

Utilizamos las variables que teníamos antes de las variables de decisión para crear una lista indexada para as variables de decisión

```
#lista de variables para x
xindx = [(a[i] , b[j]) for j in b1 for i in a1]
```

En esta parte en la función que llamamos modelo que contendrá los datos y el solver y ademas la función de minimizar

```
#creacion del modelo que contendrá los datos y la var de minimizar
modelo = LpProblem("problema del transporte", LpMinimize)


En este paso vamos a llamar a las variables de decisión
#creacion de las variables de decisión
x = LpVariable.dicts("X", xindx, 0, None)
```

Ahora escribimos la función objetivo para nuestro problema en este caso:

```
#funcion objetivo del problema
modelo += 4.0 * x[1,1] + 3.0 * x[1,2] + 8.0 * x[1,3] \
+ 7.0 * x[2,1] + 5.0 * x[2,2] + 9.0 * x[2,3] \
+ 4.0 * x[3,1] + 5.0 * x[3,2] + 5.0 * x[3,3],"costo de transporte"
```

Ahora escribimos las dos restricciones para oferta y para demanda

```
#restricciones de oferta
modelo += x[1,1] + x[1,2] + x[1,3] <= 300.0, "oferta parte 1"
modelo += x[2,1] + x[2,2] + x[2,3] <= 300.0, "oferta parte 2"
modelo += x[3,1] + x[3,2] + x[3,3] <= 100.0, "oferta parte 3"

#restricciones de demanda
modelo += x[1,1] + x[2,1] + x[3,1] >= 200.0, "demanda parte 1"
modelo += x[1,2] + x[2,2] + x[3,2] >= 200.0, "demanda parte 2"
modelo += x[1,3] + x[2,3] + x[3,3] >= 300.0, "demanda parte 3"
```

En esta parte llamamos el solver que nos brinda la librería Pulp y resolvemos el modelo utilizando GLPK solver

```
#usamos el problema utilizando el pulp solver
modelo.solve(GLPK())
```

Después de esto el problema esta resuelto, imprimimos si el estado de la solución es óptimo o no

```
#imprimo el estado de la solucion
print("estado:", LpStatus[modelo.status])
```

Universidad Tecnológica de Pereira

Para una mejor vista de lo que hizo el solver le pedimos que imprima todo lo que hizo para que lo podamos ver mejor, en este caso imprimimos primero laos valores de las variables de decisión

```
#imprimimos cada una de las variables con que se resolvió el solve
#valor optimo
for v in modelo.varibales():
    print(v.name, "=", v.varValue)

#imprimimos el valor optimizado de la funcion objetivo
print("funcion objetivo", value(modelo.objective))
```

Para finalizar tenemos que cuando ejecutemos el código la funcion objetivo dice que el costo mínimo es 3900, lo cual es correcto haciendo el ejercicio con otro método de solucion de ejercicios de programación lineal.

```
In [1]: runfile('E:/Python doc/Optimization/Soln_LP/TP Soln.py', wdir='E:/Python doc/
Optimization/Soln_LP')
Status: Optimal
X_(1,_1) = 200.0
X_(1,_2) = 100.0
X_(1,_3) = 0.0
X_(2,_1) = 0.0
X_(2,_2) = 100.0
X_(2,_3) = 200.0
X_(3,_1) = 0.0
X_(3,_2) = 0.0
X_(3,_3) = 100.0
Objective Function 3900.0

In [2]:
```