problema di trasporto 2

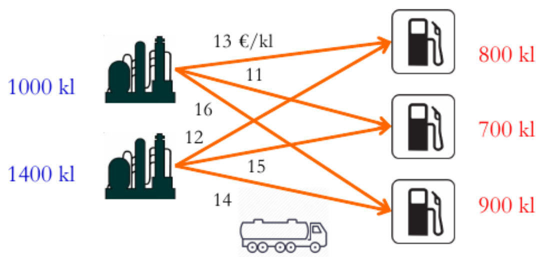
July 10, 2021

1 problema di trasporto 2

Due depositi (A e B) di carburante riforniscono tre distributori (1,2 e 3).

Ogni deposito ha una disponibilità limitata di carburante e ogni distributore ha una richiesta nota.

I costi di trasporto tra depositi e distributori dipende dalla distanza e dalla quantità di carburante



		distributori			
	disponibilità (kl)	Milano (1)	Roma (2)	Napoli (3)	
deposito A	1000	13	11	16	costi di trasporto (€/kl)
deposito B	1400	12	15	14	
	richiesta (kl)	800	700	900	

Quanto carburante inviare da ogni deposito a ogni distributore in modo da soddisfare tutte le richieste al costo minimo?

1.1 formulazione generale

Una rete logistica (single-commodity) è formata da n sorgenti $S_1,\dots,S_n$ e m destinazioni $T_1,\dots,T_m$.

Dati la disponibilità di prodotto di ogni sorgente, la richiesta di prodotto di ogni destinazione e il costo unitario di trasporto per ogni coppia sorgente-destinazione, quanto prodotto trasportare da ogni sorgente a ogni destinazione in modo da rispettare le disponibilità delle sorgenti e soddisfare le richieste delle destinazioni al costo minimo?

1.2 modello parametrico

Parametri
- n numero di sorgenti (ad es. depositi, impianti, ...)
- m numero di destinazioni (ad es. punti vendita, utenti, ...).

- c_{ij} costo unitario di trasporto dalla sorgente S_i alla destinazione T_j
- d_i disponibilità della sorgente S_i
- r_j richiesta della destinazione T_j

variabili decisionali
- $x_{ij} \in \mathbb{R}$: unità di prodotto trasportate dalla sorgente S_i alla destinazione T_j

funzione obiettivo:
minimizzazione del costo totale

$$z = \min \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij}$$

vincoli:
- La quantità totale prelevata da ogni sorgente non può superare la disponibilità.

$$\sum_{j=1}^{m} x_{ij} \le d_i \qquad \forall i = 1, \dots, n$$

- La quantità totale consegnata a una destinazione deve essere almeno pari alla richiesta.

$$\sum_{i=1}^{n} x_{ij} \ge r_j \qquad \forall j = 1, \dots, m$$

- Le quantità trasportate sono numeri non negativi.

$$x_{ij} \ge 0 \qquad \forall i = 1, \dots, n \quad \forall j = 1, \dots, m$$

[5]:
```python
from pyomo.environ import *
from pyomo.opt import SolverStatus, TerminationCondition

model = ConcreteModel()
model.name = "problema di miscelazione"

i = [ 'deposito A', 'deposito B' ]
d = [ 1000, 1400 ]

j = [ 'milano', 'roma', 'napoli' ]
r = [ 800, 700, 900 ]

c = [[13,   11,  16],
     [12,   15,  14]]

model.i = Set(initialize=i)
model.j = Set(initialize=j)

c_dict = {}
for i, mi in enumerate(model.i):
    for j, mj in enumerate(model.j):
        c_dict[mi, mj] = c[i][j]

d = {mi: d[i] for i, mi in enumerate(model.i)}
r = {mj: r[j] for j, mj in enumerate(model.j)}

model.d = Param(model.i, initialize=d)
model.r = Param(model.j, initialize=r)

model.c = Param(model.i, model.j, initialize=c_dict)

model.x = Var(model.i, model.j, domain=NonNegativeReals, initialize=0)

obj_expr = sum(sum(model.c[i, j]*model.x[i, j] for j in model.j) for i in model.i)

model.cost = Objective(expr = obj_expr, sense=minimize)

model.constraints = ConstraintList()

for i in model.i:
    model.constraints.add(expr = sum(model.x[i, j] for j in model.j) <= model.d[i])

for j in model.j:
    model.constraints.add(expr = sum(model.x[i, j] for i in model.i) >= model.r[j])

results = SolverFactory('glpk').solve(model)

model.display()
```

```
Model problema di miscelazione

  Variables:
    x : Size=6, Index=x_index
        Key                       : Lower : Value : Upper : Fixed : Domain
        ('deposito A', 'milano') :     0 : 300.0 :  None : False : NonNegativeReals
        ('deposito A', 'napoli') :     0 :   0.0 :  None : False : NonNegativeReals
          ('deposito A', 'roma') :     0 : 700.0 :  None : False : NonNegativeReals
        ('deposito B', 'milano') :     0 : 500.0 :  None : False : NonNegativeReals
        ('deposito B', 'napoli') :     0 : 900.0 :  None : False : NonNegativeReals
          ('deposito B', 'roma') :     0 :   0.0 :  None : False : NonNegativeReals

  Objectives:
    cost : Size=1, Index=None, Active=True
        Key  : Active : Value
        None :   True : 30200.0

  Constraints:
    constraints : Size=5
        Key : Lower : Body   : Upper
          1 :  None : 1000.0 : 1000.0
          2 :  None : 1400.0 : 1400.0
          3 : 800.0 :  800.0 :   None
          4 : 700.0 :  700.0 :   None
          5 : 900.0 :  900.0 :   None
```

[ ]: