ROAR III

Ricerca Operativa Applicazioni Reali

Alessandro Gobbi Alice Raffaele Gabriella Colajanni Eugenia Taranto

IIS Antonietti, Iseo (BS)

26 novembre 2022

Introduzione

Chi siamo e i nostri contatti

Alessandro Gobbi (UniBS)
alessandro.gobbi@unibs.it

Alice Raffaele (UniVR)
alice.raffaele@univr.it

Gabriella Colajanni (UniCT)
gabriella.colajanni@unict.it

Eugenia Taranto (UniCT)
eugenia.taranto@unict.it

Correzione dei compiti – Secret Santa

Secret Santa

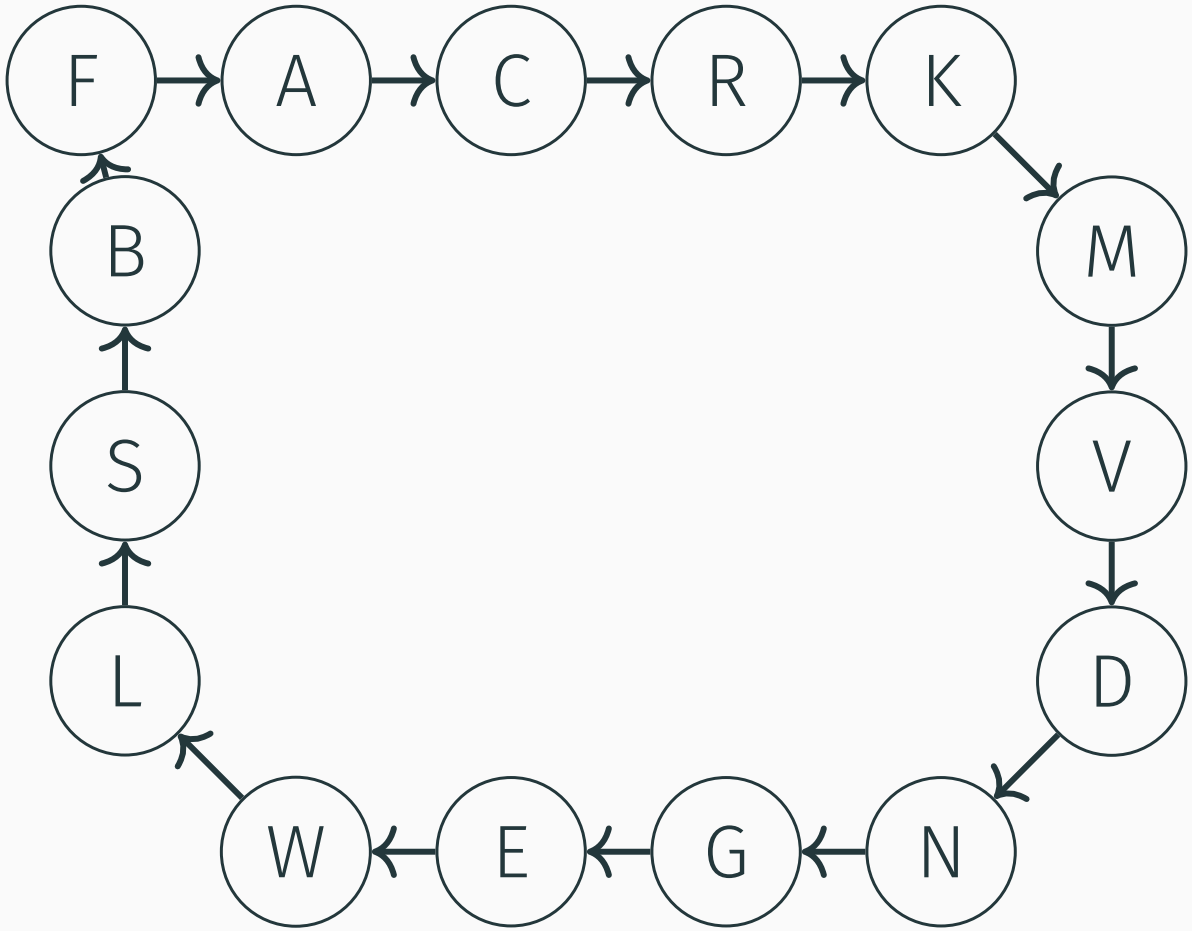
Per Natale, il gruppo di lettura della Biblioteca di Chiari organizza un *Secret Santa* per scambiarsi dei doni. In particolare, ogni partecipante che aderisce all'iniziativa è assegnato a un altro a cui dovrà regalare un libro e, a sua volta, riceverà un libro in dono da un'altra persona ancora.

L'elenco delle persone aderenti è il seguente: Alessandra, Bruna, Carlo, Daniela, Elisa, Fabio, Germana, Katia, Luca, Mariangela, Nicola, Roberta, Simone, Vilma e William. Avendo proposto l'iniziativa anche l'anno precedente, la coordinatrice del gruppo di lettura:

- preferirebbe non riassegnare gli stessi destinatari alle stesse persone;
- vuole evitare che due persone siano assegnate l'una all'altra;
- vuole che sia Fabio a regalare un libro ad Alessandra.

Gli assegnamenti già fatti in passato sono i seguenti: Alessandra ha già regalato un libro a Elisa, Bruna a Nicola, Carlo a Simone, Daniela a Katia, Elisa a Fabio, Fabio a Germana, Katia ad Alessandra, Luca a William.

1. A quale problema classico della Ricerca Operativa che abbiamo già incontrato può essere ridotto questo problema?



Al problema del Commesso Viaggiatore
(Travelling Salesman Problem)

2. Implementare e risolvere il problema con Python+PuLP in due modi:
    2.1 prima, scrivendo tutti i dati dell'istanza direttamente nello script (come avete sempre fatto finora);
    2.2 poi, provando a leggere i dati dell'istanza da un file di testo (che vi forniamo) e adattando di conseguenza lo script.

Correzione compiti – Secret Santa v1

```
from pulp import *

PARTECIPANTI = ['Alessandra', 'Bruna', 'Carlo', 'Daniela', 'Elisa', 'Fabio', 'Germana',
                'Katia', 'Luca', 'Mariangela', 'Nicola', 'Roberta', 'Simone', 'Vilma', 'William']

NON_PREFERITI = [('Alessandra', 'Elisa'), ('Bruna', 'Nicola'), ('Carlo', 'Simone'), ('Daniela', 'Katia'),
           ('Elisa', 'Fabio'), ('Fabio', 'Germana'), ('Katia', 'Alessandra'), ('Luca', 'William')]
costo_base = 1
penalità_non_preferiti = 1000

# Definizione del modello
modello = LpProblem('SecretSanta', LpMinimize)
# Variabili
x = LpVariable.dicts('x', (PARTECIPANTI, PARTECIPANTI), lowBound=0, upBound=1, cat=LpBinary)
# Vincoli
for i in PARTECIPANTI:
    modello += lpSum(x[i][j] for j in PARTECIPANTI if j != i) == 1 # ognuno regala un libro a qualcun altro
    modello += lpSum(x[k][i] for k in PARTECIPANTI if k != i) == 1 # ognuno riceve un libro da qualcun altro

for i in PARTECIPANTI:
    for j in PARTECIPANTI:
        if i != j:
            modello += x[i][j] + x[j][i] <= 1 # due persone non possono essere assegnate l'uno all'altra
modello += x['Fabio']['Alessandra'] == 1 # Fabio regala un libro ad Alessandra

# Funzione obiettivo
modello += lpSum(x[i][j] for i in PARTECIPANTI for j in PARTECIPANTI)*costo_base
        + lpSum(x[i][j] for (i,j) in NON_PREFERITI)*penalità_non_preferiti

status = modello.solve(PULP_CBC_CMD(msg=0))
if status == 1:
    print("Funzione obiettivo:", round(value(modello.objective)))
    print("\nAssegnamenti:")
    for i in PARTECIPANTI:
        for j in PARTECIPANTI:
           if x[i][j].varValue > 0:
               print("{} --> {}".format(i,j))
```

Correzione compiti – Secret Santa v2

```
from pulp import *

def leggi_input(nome_file):
    PARTECIPANTI = []
    NON_PREFERITI = []
    FISSATI = []
    costo_base = 0
    penalità_non_preferiti = 0
    with open(nome_file, 'r') as file:
       while True:
           riga = file.readline()
           if not riga:
               break
           if "PARTECIPANTI" in riga:
               riga_pezzi = riga.split()
               PARTECIPANTI = riga_pezzi[2:]
           if "NUM_NON_PREFERITI" in riga:
               riga_pezzi = riga.split()
               num_non_preferiti = int(riga_pezzi[2])
               for i in range(num_non_preferiti):
                   riga = file.readline()
                   riga_pezzi = riga.split()
                   coppia = (riga_pezzi[0], riga_pezzi[1])
                   NON_PREFERITI += [coppia]
           if "NUM_FISSATI" in riga:
               riga_pezzi = riga.split()
               num_fissati = int(riga_pezzi[2])
               for i in range(num_fissati):
                   riga = file.readline()
                   riga_pezzi = riga.split()
                   coppia = (riga_pezzi[0], riga_pezzi[1])
                   FISSATI += [coppia]
           if "COSTO_BASE" in riga:
               riga_pezzi = riga.split()
               costo_base = int(riga_pezzi[2])
           if "PENALITA_NON_PREFERITI" in riga:
               riga_pezzi = riga.split()
               penalità_non_preferiti = int(riga_pezzi[2])
    return PARTECIPANTI, NON_PREFERITI, FISSATI, costo_base, penalità_non_preferiti

PARTECIPANTI, NON_PREFERITI, FISSATI, costo_base, penalità_non_preferiti = leggi_input('secret_santa_v2.txt')
```

```
# Definizione del modello
modello = LpProblem('SecretSanta', LpMinimize)

# Variabili
x = LpVariable.dicts('x', (PARTECIPANTI, PARTECIPANTI), lowBound=0, upBound=1, cat=LpBinary)

# Vincoli
for i in PARTECIPANTI:
    modello += lpSum(x[i][j] for j in PARTECIPANTI if j != i) == 1 # ognuno regala un libro a qualcun altro
    modello += lpSum(x[k][i] for k in PARTECIPANTI if k != i) == 1 # ognuno riceve un libro da qualcun altro

for i in PARTECIPANTI:
    for j in PARTECIPANTI:
        if i != j:
            modello += x[i][j] + x[j][i] <= 1 # due persone non possono essere assegnate l'uno all'altra

for (i,j) in FISSATI:
    modello += x[i][j] == 1

# Funzione obiettivo
modello += lpSum(x[i][j] for i in PARTECIPANTI for j in PARTECIPANTI)*costo_base
       + lpSum(x[i][j] for (i,j) in NON_PREFERITI)*penalità_non_preferiti

status = modello.solve(PULP_CBC_CMD(msg=0))
if status == 1:
    print("Funzione obiettivo:", round(value(modello.objective)))
    print("\nAssegnamenti:")
    for i in PARTECIPANTI:
        for j in PARTECIPANTI:
           if x[i][j].varValue > 0:
               print("{} --> {}".format(i,j))
```

Correzione compiti – Soluzione ottima

```
In [2]: runfile('/Users/aliceraffaele/Desktop/Work/ROAR-III/secret_santa/secret_santa_v2.py',
wdir='/Users/aliceraffaele/Desktop/Work/ROAR-III/secret_santa')
Funzione obiettivo: 15

Assegnamenti:
Alessandra --> Carlo
Bruna --> Fabio
Carlo --> Roberta
Daniela --> Nicola
Elisa --> William
Fabio --> Alessandra
Germana --> Elisa
Katia --> Mariangela
Luca --> Simone
Mariangela --> Vilma
Nicola --> Germana
Roberta --> Katia
Simone --> Bruna
Vilma --> Daniela
William --> Luca
```

Parte 1:
Progetto finale

Vedasi slides *ROAR-Anno3-ProgettoFinale*

Parte 2:
Seminario aziendale di Stefano Bortolomiol (Optit s.r.l.)

Applicazioni multisettoriali della ricerca operativa: un'esperienza aziendale

OPTIT
optimal solutions

Optit è un'azienda con sedi a Bologna e Cesena che offre software e servizi di consulenza incentrati sulla ricerca operativa. In questo seminario **Stefano Bortolomiol** racconta come, combinando conoscenze matematiche, informatiche, ingegneristiche e consulenziali, Optit supporti aziende pubbliche e private nella risoluzione di problemi complessi nel settore energetico, nei servizi ambientali e nella logistica.

Conclusione

Dal codice al problema...

Abbiamo trovato il file *`Problema_1.py`* che contiene l'implementazione in Python+Pulp di un modello matematico.

Purtroppo il codice non è nemmeno commentato e quindi non abbiamo alcun indizio sulla formulazione matematica del problema.

Potreste aiutarci, partendo dal codice, a scrivere sia la formulazione matematica sia un plausibile testo che si adatti bene al problema?

Dal codice al problema...

```
from pulp import *

model = LpProblem("Problema?", LpMaximize)

indici = [1,2,3]
Parametri={1:150, 2: 320, 3:60}
Valori_1 = {1: 9500, 2: 123}
valori_2 = {
    1: {1: 30, 2: 120, 3: 12},
    2: {1: 0.7, 2: 1, 3: 0.5}}
valori_3 = {1: 0, 2: -1, 3:5}

var_x = LpVariable.dicts("x", indici, 0, None, LpInteger)

model += lpSum(var_x[i] * Parametri[i] for i in indici)

for j in valori_2.keys():
    model += lpSum(var_x[i]*valori_2[j][i] for i in indici) <= Valori_1[j]

model += lpSum(valori_3[k]*var_x[k] for k in indici) >= 0

status = model.solve(PULP_CBC_CMD(msg = 0))
if status == 1 :
    print("Soluzione ottima:")
    for v in model.variables():
            print(v.name, " = ", v.varValue)

    print("Il valore della funzione obiettivo è {}\n\n".format(round(value(model.objective),2)))
else:
    print("Problema non ammissibile")
```

Compiti per sabato 03 dicembre 2022

1. Lavoro di gruppo *`Dal codice al problema...`*
2. Sul progetto finale

Consegnare alla Prof.ssa Picchi i compiti **entro giovedì 01 dicembre**.

1) Lavoro di gruppo `Dal codice al problema...`

1. Leggere bene lo script *`Problema_1.py`* e comprendere ogni riga del codice.
2. Scrivere la formulazione del modello matematico descritto dal codice.
3. Ideare un plausibile testo del problema, coerente con il modello.

2) Sul progetto finale

Leggere molto attentamente la descrizione del progetto finale *`Filtrec`* per trovare almeno due analogie e almeno una differenza con il problema *`VRP Challenge`* di ROAR II.