KNAPSACK 01 (KP-01)

- n oggetti
- p_i: profitto oggetto i-esimo
- w_i: peso oggetto i-esimo
- c: capacità dello zaino
- massimizzare il profitto degli oggetti inseriti nello zaino rispettando il vincolo di capacità

Modello PLI di KP-01

Modello matematico del KP01:

x_i = 1 se oggetto i selezionato,

0 altrimenti

$$\max \Sigma_{i=1,n}\ p_i\, x_i$$

$$\Sigma_{i=1,n}\ w_i\, x_i \ <= c$$

$$x_i \in \{0,1\} \qquad\qquad i = 1,\ldots, n$$

Esempio di KP-01

Numero oggetti:

$n = 10$

Profitti:

$p_i = (5,7,4,2,2,1,8,6,4,9)$

Pesi:

$w_i = (8,9,3,2,3,1,4,7,5,10)$

Capacità:

$c = 14$

Soluzione MPL di KP-01(1)

```
TITLE
KP01;
INDEX
oggetti := 1..10;
DATA
Profitti[oggetti]:=(5,7,4,2,2,1,8,6,4,9);
Pesi[oggetti]:=(8,9,3,2,3,1,4,7,5,10);
capacita := 14;
```

Soluzione MPL di KP-01(2)

```
VARIABLES
x[oggetti];
MODEL
MAX z = SUM(oggetti: x*profitti);
SUBJECT TO
SUM(oggetti: x * pesi)  <=  capacita;
BINARY x
END
```

MULTIPLE KNAPSACK 01 (MKP-01)

- n oggetti
- p_i: profitto oggetto i-esimo
- w_i: peso oggetto i-esimo
- m: numero di zaini a disposizione
- c: capacità zaini
- un oggetto può essere inserito in un solo zaino
- massimizzare il profitto degli oggetti inseriti negli zaini

Modello PLI di MKP-01

Modello matematico del MKP-01:

x_{ij} = 1 se oggetto i inserito nello zaino j,

0 altrimenti

$$\max \Sigma_{i=1,n} p_i (\Sigma_{j=1,m} x_{ij})$$

$$\Sigma_{i=1,n} w_i x_{ij} <= c \qquad j = 1, \ldots, m$$

$$\Sigma_{j=1,m} x_{ij} <= 1 \qquad i = 1, \ldots, n$$

$$x_{ij} \in \{0,1\} \quad i = 1,\ldots, n \quad j = 1, \ldots, m$$

Esempio di MKP-01

Numero oggetti:

$n = 10$

Profitti:

$p_i = (5,7,4,2,2,1,8,6,4,9)$

Pesi:

$w_i = (8,9,3,2,3,1,4,7,5,10)$

Capacità:

$c = 14$

Numero zaini disponibili:

$m = 2$

Soluzione MPL di MKP-01(1)

```
TITLE
MKP01;
INDEX
oggetti := 1..10;
bin := 1..2;
DATA
Profitti[oggetti]:=(5,7,4,2,2,1,8,6,4,9);
Pesi[oggetti]:=(8,9,3,2,3,1,4,7,5,10);
capacita := 14;
```

Soluzione MPL di MKP-01(2)

```
VARIABLES
x[oggetti, bin];
MODEL
MAX z = SUM(oggetti, bin : x*profitti);
SUBJECT TO
cap[bin]: SUM(oggetti: x[bin] * pesi)  <=
  capacita;
v[oggetti]: SUM(bin: x[oggetti] ) <= 1;
BINARY x
END
```

LOCALIZZAZIONE DI IMPIANTI (FL)

- n possibili impianti
- Ogni impianto j è caratterizzato da:

  F_j = costo iniziale

  V_j = costo per unità di prodotto

  M_j = massima capacità produttiva (per il periodo in esame)
- Minimizzare il costo complessivo necessario per produrre, nel periodo in esame, R unità di prodotto.

Modello PLI di FL

Variabili decisionali:

Y_j = 1 se impianto j utilizzato, 0 altrimenti

X_j = unità di prodotto realizzate dall'impianto j nel periodo in esame

Modello PLI di FL (2)

Modello matematico:

$$\min \Sigma_{j=1,n} (F_j Y_j + V_j X_j)$$

$$\Sigma_{j=1,n} X_j >= R$$

$$X_j <= M_j Y_j \qquad j = 1, \ldots, n$$

$$X_j >= 0 \qquad j = 1, \ldots, n$$

$$Y_j \in \{0,1\} \qquad j = 1, \ldots, n$$

Esempio di FL

Numero impianti:

$n = 6$

Capacità produttiva max impianto:

$M_j = (100,90,75,30,45,70)$

Costo fisso apertura impianto:

$F_j = (256,200,180,90,110,135)$

Costo variabile produttivo:

$V_j = (7,9,11,18,16,10)$

Richiesta prodotto:

$R = 200$

Soluzione MPL di FL (1)

```
INDEX
impianti:=(1,2,3,4,5,6);
DATA
Max_prod[impianti]:=(100,90,75,30,45,70);
Fix_cost[impianti]:=(256,200,180,90,110,135);
Var_cost[impianti]:= (7,9,11,18,16,10);
Req:= 200;
```

Soluzione MPL di FL (2)

```
DECISION VARIABLES
prod[impianti];   !produzione impianto
costr[impianti];  !impianto costruito o no

MACRO
Prod_cost:= SUM(impianti: prod * Var_cost);
Init_cost:= SUM(impianti: costr *Fix_cost);
```

Soluzione MPL di FL (3)

```
MODEL
MIN Costo=Init_cost + Prod_cost;
SUBJECT TO
TOTP: SUM(impianti: prod) >= Req;
CMAX[impianti]: prod - Max_prod * costr <= 0;
BINARY
costr[impianti];
END
```