

Appunti delle Esercitazione di Ricerca Operativa AMPL Plus v1.6

a cura di G. Liuzzi*

a.a. 2000-2001

1 Modello di Produzione multi-impianto

Una industria manifatturiera dispone di due impianti per la produzione, rispettivamente impianto A e B. Ciascun impianto concorre alla fabbricazione di due tipi di uno stesso prodotto, *standard* e *deluxe*. Una singola unità di prodotto modello *standard* ha un prezzo di vendita di 10\$ mentre una unità di prodotto modello *deluxe*, più costoso, ha un prezzo di vendita di 15\$.

Ogni impianto (A e B) dispone di due processi per la fabbricazione dei suoi prodotti, levigatura e pulitura. Tuttavia, mentre l'impianto A può disporre complessivamente di 80 ore per la levigatura e 60 per la pulitura, l'impianto B dispone di 60 ore di levigatura e 75 di pulitura.

Nella tabella sottostante sono riportati, per ogni impianto, i tempi (in ore per unità di prodotto) necessari rispettivamente ai processi di levigatura e pulitura.

| | Impianto A | | Impianto B | |
|------------|------------|--------|------------|--------|
| | Standard | Deluxe | Standard | Deluxe |
| Levigatura | 4 | 2 | 5 | 3 |
| Pulitura | 2 | 5 | 5 | 6 |

In oltre, per fabbricare ogni unità di ogni prodotto sono necessari 4Kg di un materiale grezzo (p.es. ferro). L'industria può complessivamente (impianto A + B) contare su 120Kg di ferro a settimana.

Quello che il manager dell'industria ha pensato, inizialmente, di fare è stato di ripartire il ferro tra i due impianti nel seguente modo:

- 75Kg all'impianto A;
- 45Kg all'impianto B;

dando, cioè, un maggior quantitativo di materia prima all'impianto A perché dotato di macchine più veloci e quindi in grado, probabilmente, di far fruttare maggiormente le risorse assegnategli.

A questo punto possiamo considerare i due Impianti A e B separatamente l'uno dall'altro e formulare per essi due problemi di programmazione lineare per massimizzare i profitti settimanali.

Impianto A: È abbastanza semplice riconoscere in

- x_{st}^A ≡ unità di prodotto, nel modello standard, fabbricate settimanalmente;
- x_{dl}^A ≡ unità di prodotto, nel modello deluxe, fabbricate settimanalmente.

*liuzzi@dis.uniroma1.it

le variabili di decisione per il nostro problema. Con questa scelta per le variabili di decisione e trascurando il vincolo di interezza per ragioni di semplicità espositiva, possiamo scrivere il problema di PL associato all'impianto A nel seguente modo:

$$\begin{aligned}
 \max \quad & 10x_{st}^A + 15x_{dl}^A && \leftarrow \text{massimizziamo i profitti} \\
 & 4x_{st}^A + 4x_{dl}^A \leq 75 && \leftarrow \text{vincolo sul quantitativo di ferro} \\
 & 4x_{st}^A + 2x_{dl}^A \leq 80 && \leftarrow \text{vincolo sulle ore di levigatura} \\
 & 2x_{st}^A + 5x_{dl}^A \leq 60 && \leftarrow \text{vincolo sulle ore di pulitura} \\
 & x_{st}^A \geq 0, \quad x_{dl}^A \geq 0
 \end{aligned} \tag{1}$$

A questo punto, prima di passare all'impianto B, vediamo come fare per scrivere in AMPL Plus il precedente problema di PL. Per fare questo è sufficiente creare il seguente file di modello (un semplice file di testo ma con estensione `.mod`) p.es. `multiplantA.mod`.

```

multiplantA.mod
-----
var x_stA;
var x_dlA;

maximize profit: 10*x_stA + 15*x_dlA;

subject to      ferro: 4*x_stA + 4*x_dlA <= 75;
subject to      levigatura: 4*x_stA + 2*x_dlA <= 80;
s.t.            pulitura: 2*x_stA + 5*x_dlA <= 60;
s.t.            st_limit: x_stA >= 0;
s.t.            dl_limit: x_dlA >= 0;

```

Ora esaminiamo nel dettaglio il file `multiplantA.mod` di sopra. Notiamo che:

- ogni variabile è definita da una istruzione `var`;
- la funzione obiettivo viene definita da una istruzione che comincia con la parola chiave `maximize` (o `minimize` nel caso il problema fosse di minimizzazione) seguita da un nome per la funzione obiettivo (in questo caso `profit`);
- ogni vincolo è definito in una istruzione che comincia con la parola chiave `subject to` (ma è possibile anche l'abbreviazione `s.t.`) seguita da un nome simbolico per il vincolo stesso. In questo caso abbiamo usato i nomi `ferro`, `levigatura`, `pulitura`, `st_limit`, `dl_limit`.

A questo punto, per risolvere il problema in AMPL Plus è sufficiente aprire il file `multiplantA.mod`, quindi, dal menu Run della finestra di AMPL Plus, scegliere prima la voce `Build model` e poi `Solve problem`. Se il solutore scelto è in grado di risolvere il problema che abbiamo scritto, sarà possibile vedere i risultati dell'ottimizzazione scegliendo, sempre dal menu Run, il comando `Build results`.

Se il file di modello è stato scritto correttamente, all'ottimo l'impianto A ha un profitto di 225\$ ottenuto fabbricando 11,25 unità standard e 7,5 deluxe. In particolare si è utilizzato tutto il ferro e sono state sfruttate tutte le ore di pulitura a disposizione. Le macchine per la levigatura, invece, sono state sottoutilizzate essendo avanzate 20 ore di tempo macchina. Supponiamo ora di voler generalizzare il precedente problema nel modo seguente. Si hanno:

- un insieme P di prodotti, anziché due soli prodotti
- un insieme R di risorse a disposizione (anziché solo Kg di ferro, ore di levigatura e ore di pulitura)
- per ogni prodotto $i \in P$, c_i = prezzo unitario del prodotto i

- per ogni risorsa $j \in R$, b_j = quantità massima disponibile di risorsa j
- per ogni prodotto $i \in P$ e risorsa $j \in R$ a_{ij} = quantità di risorsa j necessaria alla produzione di una unità di prodotto i

Concordemente con quanto detto in precedenza, le variabili di decisione saranno per ogni prodotto $i \in P$

x_i = unità di prodotto i fabbricate settimanalmente.

Quindi, un problema che generalizzi il problema di produzione dell'impianto A si ottiene, anche qui trascurando il vincolo di interezza sulle variabili di decisione, scrivendo:

$$\begin{aligned} \max \quad & \sum_{i \in P} c_i x_i = c^\top x \\ \text{c.v.} \quad & \sum_{i \in P} a_{ij} x_i \leq b_j, \quad \text{per ogni } j \in R \\ & x_i \geq 0, \quad \text{per ogni } i \in P \end{aligned}$$

Vediamo ora in che modo tradurre il precedente modello *generale* di problema di produzione in AMPL. La procedura è esattamente quella di prima solo che ora avremo il file `multiplant.mod` così fatto:

```

multiplant.mod
-----
set PROD;
set RISORSE;

param b{RISORSE};
param a{PROD,RISORSE};
param c{PROD};

var x{j in PROD} >= 0;

maximize profit: sum{j in PROD} c[j]*x[j];

s.t. vincolo{i in RISORSE}: sum{j in PROD} a[j,i]*x[j] <= b[i];

```

Ora analizziamo le istruzioni del file `multiplant.mod`. Anzitutto notiamo le istruzioni

```

set PROD;
set RISORSE;

```

con la quale si dichiarano `PROD` e `RISORSE` come insiemi di un numero imprecisato di elementi (prodotti e risorse). Subito dopo abbiamo tre istruzioni che ci servono per definire altrettanti parametri del modello. La prima di queste

```

param b{RISORSE};

```

definisce un vettore di parametri con tante componenti quanti sono gli elementi dell'insieme `RISORSE`. L'istruzione successiva ovvero

```

param a{PROD,RISORSE};

```

definisce invece una matrice di parametri con tante righe quanti sono i prodotti e tante colonne quante sono le risorse necessarie alla produzione. È possibile accedere a ciascuna componente di un vettore o una matrice come si accede alle componenti di un array in C e

cioè specificando il nome del parametro, p.es. “b”, seguito dagli indici della componente che si vuole selezionare racchiusi tra parentesi quadre (b[1],b[2], ecc. oppure, nel caso della matrice “a”, a[1,2]).

La funzione obiettivo `profit`, definita dalla istruzione

```
maximize profit: sum{j in PROD} c[j]*x[j];
```

è la somma, estesa a tutti gli elementi dell'insieme *PROD* ovvero a tutti i prodotti, dei termini `c[i]*x[i]`. Infine è interessante notare l'ultima istruzione

```
s.t. vincolo{i in RISORSE}: sum{j in PROD}a[j,i]*x[j] <= b[i];
```

con la quale si definiscono tanti vincoli quanti sono gli elementi dell'insieme *RISORSE*. Nell'esempio precedente è ricorrente l'uso della espressione

```
{i in P}
```

ove *P* nel nostro caso è *PROD* o *RISORSE*. Questo costrutto è detto *espressione di indicizzazione*. Questo tipo di espressioni possono essere usate non solo per definire variabili e parametri o sommatorie ma anche per definire vincoli e comunque ogni qual volta nel modello matematico venga usato il quantificatore *per ogni* (\forall).

Ovviamente, è possibile particularizzare nuovamente questo modello generale di problema specificandone i parametri. In particolare è possibile riottenere il modello del file `multiplantA.mod` creando un file di dati `multiplantA.dat` (ancora una volta un file di testo ma con estensione `.dat`) così fatto:

```
multiplantA.dat
```

```
set    PROD := STANDARD, DELUXE;
set    RISORSE := FERRO, LEVIGATURA, PULITURA;

param:      b      :=
    FERRO    75
    LEVIGATURA 80
    PULITURA 60    ;

param      a(tr):  STANDARD    DELUXE :=
    FERRO      4            4
    LEVIGATURA 4            2
    PULITURA  2            5    ;

param:      c      :=
    STANDARD   10
    DELUXE     15    ;
```

Facciamo notare che in questo modo, avendo cioè a disposizione il file `multiplant.mod` contenente il modello separato dai dati del problema (contenuti invece nel file `multiplantA.dat`), possiamo risolvere differenti problemi di massimizzazione del profitto semplicemente cambiando i dati contenuti nel file con estensione `.dat`.

Sfruttando questo fatto è banalissimo risolvere in AMPL il problema relativo all'impianto B che esaminiamo qui di seguito.

Impianto B: Questa volta indichiamo le variabili di decisione come

- x_{st}^B \equiv unità di prodotto, nel modello standard, fabbricate settimanalmente;
- x_{dt}^B \equiv unità di prodotto, nel modello deluxe, fabbricate settimanalmente.

La formulazione matematica del problema di PL associato all'impianto B sarà analogamente al caso precedente:

$$\begin{aligned}
 \max \quad & 10x_{st}^B + 15x_{dl}^B && \leftarrow \text{massimizziamo i profitti} \\
 & 4x_{st}^B + 4x_{dl}^B \leq 45 && \leftarrow \text{vincolo sul quantitativo di ferro} \\
 & 5x_{st}^B + 3x_{dl}^B \leq 60 && \leftarrow \text{vincolo sulle ore di levigatura} \\
 & 5x_{st}^B + 6x_{dl}^B \leq 75 && \leftarrow \text{vincolo sulle ore di pulitura} \\
 & x_{st}^B \geq 0, \quad x_{dl}^B \geq 0
 \end{aligned} \tag{2}$$

in cui abbiamo modificato solo il nome delle variabili e i valori dei parametri sostituendoli con quelli relativi all'impianto B.

Siccome il modello generale è il medesimo, non ci sarà alcun bisogno, in AMPL, di riscrivere il file `.mod` potendo riusare il file di prima `multiplant.mod`. L'unica cosa da fare è invece quella di scrivere un nuovo file dei dati p.es. `multiplantB.dat`. Per completezza riportiamo qui sotto il contenuto del file `multiplantB.dat`

```

multiplantB.dat
-----
set    PROD := STANDARD, DELUXE;
set    RISORSE := FERRO, LEVIGATURA, PULITURA;

param:          b      :=
    FERRO       45
    LEVIGATURA  60
    PULITURA   75    ;

param          a(tr):  STANDARD    DELUXE    :=
    FERRO       4           4
    LEVIGATURA  5           3
    PULITURA   5           6    ;

param:          c      :=
    STANDARD    10
    DELUXE      15    ;

```

A questo punto, per risolvere il problema relativo all'impianto B in AMPL Plus è sufficiente aprire il file `multiplant.mod` ed eseguirne il build con il comando `Build model`. Poi sarà necessario aprire il file dei dati `multiplantB.dat` ed eseguirne il build con il comando `Build data`. Infine si dovrà risolvere il problema ancora una volta con il comando `Solve problem`. Se il solutore scelto è in grado di risolvere il problema che abbiamo scritto, sarà possibile vedere i risultati dell'ottimizzazione scegliendo, sempre dal menu Run, il comando `Build results`.

Se i file `.mod` e `.dat` sono stati scritti correttamente, all'ottimo l'impianto B ha un profitto di 168,75\$ ottenuto fabbricando 11,25 unità deluxe e nessuna unità standard. In particolare è stato utilizzato tutto il ferro ma, sia le macchine per la levigatura che quelle per la pulitura, sono state sottoutilizzate essendo avanzate rispettivamente 26,25 e 7,5 ore di tempo macchina.

Modello Unico: Anche se i due problemi sono stati risolti e le soluzioni ottime per ciascuno dei due calcolate, nel problema generale, quello cioè che si trova ad affrontare il manager dell'impresa, c'è ancora un elemento su cui è forse possibile intervenire con profitto. Fin ora infatti ci siamo, per così dire fidati delle indicazioni del manager accettando la sua

ripartizione del ferro tra i due impianti. Vediamo se è possibile fare di meglio. In particolare andiamo a considerare un modello unico per i due impianti A e B.

Ovviamente, pur considerando un modello unico per i due impianti, rimarranno invariati i vincoli relativi alle disponibilità di ore per la levigatura e la pulitura nei due impianti dato che ciascuno di essi possiede le proprie macchine per eseguire le due operazioni. Quello che cambia, se consideriamo un modello unico, è il vincolo sulla disponibilità di ferro. In questo caso infatti avremo che *complessivamente* (A+B) il fabbisogno di ferro non dovrà superare la quantità massima disponibile di 120Kg. In definitiva, quello che stiamo chiedendo al modello è di decidere automaticamente come ripartire il ferro tra i due impianti. Formulando questo nuovo problema ed usando le stesse variabili di decisione usate nei due casi separati, otteniamo il seguente problema di PL

$$\begin{array}{llll}
 \max & 10x_{st}^A + 15x_{dl}^A + & 10x_{st}^B + 15x_{dl}^B & \leftarrow \text{profitto di A + B} \\
 & 4x_{st}^A + 4x_{dl}^A + & 4x_{st}^B + 4x_{dl}^B & \leq 120 \leftarrow \text{vincolo sul ferro} \\
 & 4x_{st}^A + 2x_{dl}^A & & \leq 80 \leftarrow \text{vincolo sulle ore di levigatura} \\
 & 2x_{st}^A + 5x_{dl}^A & & \leq 60 \leftarrow \text{vincolo sulle ore di pulitura} \\
 & & 5x_{st}^B + 3x_{dl}^B & \leq 60 \leftarrow \text{vincolo sulle ore di levigatura} \\
 & & 5x_{st}^B + 6x_{dl}^B & \leq 75 \leftarrow \text{vincolo sulle ore di pulitura} \\
 & x_{st}^A, x_{dl}^A \geq 0 & x_{st}^B, x_{dl}^B \geq 0 &
 \end{array}$$

Un possibile file di modello AMPL per il problema precedente è il seguente

multiplantAB.mod

```

var x_stA >= 0;
var x_d1A >= 0;
var x_stB >= 0;
var x_d1B >= 0;

maximize profit: 10*x_stA + 15*x_d1A + 10*x_stB + 15*x_d1B;

s.t.      ferro: 4*x_stA + 4*x_d1A + 4*x_stB + 4*x_d1B <= 120;
s.t.      levigatura_A: 4*x_stA + 2*x_d1A <= 80;
s.t.      pulitura_A: 2*x_stA + 5*x_d1A <= 60;
s.t.      levigatura_B: 5*x_stB + 3*x_d1B <= 60;
s.t.      pulitura_B: 5*x_stB + 6*x_d1B <= 75;

```

Facendo risolvere il precedente modello ad AMPL otteniamo un profitto complessivo di 404,17\$ ottenuto frabbricando nell'impianto A: 9,17 prodotti standard e 8,33 prodotti deluxe e fabbricando, nell'impianto B, solo 12,5 prodotti deluxe. È importante notare i seguenti punti:

- il profitto totale in questo caso (404,17) è superiore alla somma dei profitti ottenuti considerando separatamente i due modelli;
- l'impianto A ora usa 70Kg di ferro (contro i 75Kg di prima) mentre l'impianto B ne usa 50Kg (contro i 45Kg di prima).

2 Problema di produzione multiperiodo

Una fabbrica produce due tipi di pneumatici A e B ed ha una gestione trimestrale della produzione. Per i prossimi tre mesi deve soddisfare il seguente ordine (espresso in numero di pneumatici richiesti ciascun mese)

| | tipo A | tipo B |
|-----------------|--------|--------|
| ottobre | 9000 | 8000 |
| novembre | 9000 | 8000 |
| dicembre | 9000 | 8000 |

Per la produzione di questi pneumatici la fabbrica dispone di due linee di produzione **L1** e **L2**. Per avere un pneumatico finito e pronto per essere venduto, è necessaria la lavorazione di materiale grezzo su solo una delle due macchine. Il numero di ore in cui le macchine sono disponibili ciascun mese sono riportate nella seguente tabella

| | L1 | L2 |
|-----------------|------|------|
| ottobre | 2000 | 3000 |
| novembre | 400 | 800 |
| dicembre | 200 | 1000 |

I tempi necessari per produrre questi pneumatici varia a seconda del tipo e della macchina usata. Tali tempi sono riportati nella seguente tabella (in ore)

| | L1 | L2 |
|---------------|------|------|
| tipo A | 0.10 | 0.12 |
| tipo B | 0.12 | 0.18 |

Il costo di ogni ora di lavorazione su una linea di produzione è uguale per entrambe le linee ed è pari a lire 6000. Il costo del materiale grezzo necessario per produrre ciascun pneumatico è di lire 2500 per il tipo A e di lire 4000 per il tipo B.

Nel primo e nel secondo mese del trimestre è possibile produrre più di quanto richiesto nello stesso mese; la produzione in eccesso deve essere immagazzinata per essere usata nel mese successivo. Ogni mese, il costo di tale immagazzinamento è di lire 500 per ciascun pneumatico immagazzinato. Si assuma che all'inizio del trimestre non ci sia nessun prodotto immagazzinato e analogamente alla fine del trimestre non rimanga nessun prodotto immagazzinato.

Costruire un modello lineare che permetta di pianificare la produzione trimestrale minimizzando il costo complessivo trascurando l'interezza dei prodotti.

Formulazione.

Si tratta di un problema di allocazione ottima di risorse nel quale si deve tenere presente la possibilità dell'immagazzinamento del prodotto in eccesso.

– *Variabili.* Si introducono le variabili x_{ij}^h che indicano la quantità di pneumatici di tipo $j = A, B$ prodotti dalla i -esima linea produzione ($i = 1, 2$) nell' h -esimo mese ($h = 1, 2, 3$ per ottobre, novembre e dicembre). Si indichino inoltre con y_i^h , la quantità di pneumatici di tipo $i = A, B$ da immagazzinare nell' h -esimo mese (con $h = 1, 2$ per ottobre e novembre).

– *Funzione obiettivo.* La funzione obiettivo da minimizzare è data dal costo complessivo di produzione. Poiché un'ora di lavorazione su una linea di produzione costa lire 6000, e poiché i tempi di lavorazione cambiano a seconda della linea di produzione utilizzata, per produrre ciascun pneumatico di tipo A si spende lire 600 se si utilizza la linea **L1** e lire 720 se si utilizza la linea **L2**. Analogamente, il costo di ciascun pneumatico del tipo B è di lire

720 se si utilizza la macchina 1, e di lire 1080 se si utilizza la macchina 2. Quindi tenendo conto del costo del materiale grezzo e dell'immagazzinamento, il costo complessivo sarà

$$\begin{aligned}
& 600(x_{11}^{ott} + x_{11}^{nov} + x_{11}^{dic}) + 720(x_{21}^{ott} + x_{21}^{nov} + x_{21}^{dic}) + \\
& + 720(x_{12}^{ott} + x_{12}^{nov} + x_{12}^{dic}) + 1080(x_{22}^{ott} + x_{22}^{nov} + x_{22}^{dic}) + \\
& + 2500(x_{11}^{ott} + x_{11}^{nov} + x_{11}^{dic} + x_{21}^{ott} + x_{21}^{nov} + x_{21}^{dic}) + \\
& + 4000(x_{12}^{ott} + x_{12}^{nov} + x_{12}^{dic} + x_{22}^{ott} + x_{22}^{nov} + x_{22}^{dic}) + \\
& + 500(y_1^{ott} + y_1^{nov} + y_2^{ott} + y_2^{nov}).
\end{aligned}$$

– *Vincoli.* I vincoli dovuti alla disponibilità limitata delle macchine sono

$$\begin{aligned}
0.10x_{11}^{ott} + 0.12x_{12}^{ott} & \leq 2000 \\
0.10x_{11}^{nov} + 0.12x_{12}^{nov} & \leq 400 \\
0.10x_{11}^{dic} + 0.12x_{12}^{dic} & \leq 200
\end{aligned}$$

$$\begin{aligned}
0.12x_{21}^{ott} + 0.18x_{22}^{ott} & \leq 3000 \\
0.12x_{21}^{nov} + 0.18x_{22}^{nov} & \leq 800 \\
0.12x_{21}^{dic} + 0.18x_{22}^{dic} & \leq 1000.
\end{aligned}$$

Si hanno inoltre i seguenti vincoli dovuti alla richiesta e all'immagazzinamento:

$$\begin{aligned}
x_{11}^{ott} + x_{21}^{ott} & = 9000 + y_1^{ott} \\
x_{11}^{nov} + x_{21}^{nov} + y_1^{ott} & = 9000 + y_1^{nov} \\
x_{11}^{dic} + x_{21}^{dic} + y_1^{nov} & = 9000
\end{aligned}$$

$$\begin{aligned}
x_{12}^{ott} + x_{22}^{ott} & = 8000 + y_2^{ott} \\
x_{12}^{nov} + x_{22}^{nov} + y_2^{ott} & = 8000 + y_2^{nov} \\
x_{12}^{dic} + x_{22}^{dic} + y_2^{nov} & = 8000.
\end{aligned}$$

Si hanno inoltre i vincoli di non negatività sulle variabili. Quindi il modello finale è:

$$\begin{aligned}
\min \quad & \left(600(x_{11}^{ott} + x_{11}^{nov} + x_{11}^{dic}) + 720(x_{21}^{ott} + x_{21}^{nov} + x_{21}^{dic}) + \right. \\
& + 720(x_{12}^{ott} + x_{12}^{nov} + x_{12}^{dic}) + 1080(x_{22}^{ott} + x_{22}^{nov} + x_{22}^{dic}) + \\
& + 2500(x_{11}^{ott} + x_{11}^{nov} + x_{11}^{dic} + x_{21}^{ott} + x_{21}^{nov} + x_{21}^{dic}) + \\
& + 4000(x_{12}^{ott} + x_{12}^{nov} + x_{12}^{dic} + x_{22}^{ott} + x_{22}^{nov} + x_{22}^{dic}) + \\
& \left. + 500(y_1^{ott} + y_1^{nov} + y_2^{ott} + y_2^{nov}) \right) \\
\\
& 0.10x_{11}^{ott} + 0.12x_{12}^{ott} \leq 2000 \\
& 0.10x_{11}^{nov} + 0.12x_{12}^{nov} \leq 400 \\
& 0.10x_{11}^{dic} + 0.12x_{12}^{dic} \leq 200 \\
& 0.12x_{21}^{ott} + 0.18x_{22}^{ott} \leq 3000 \\
& 0.12x_{21}^{nov} + 0.18x_{22}^{nov} \leq 800 \\
& 0.12x_{21}^{dic} + 0.18x_{22}^{dic} \leq 1000. \\
& x_{11}^{ott} + x_{21}^{ott} = 9000 + y_1^{ott} \\
& x_{11}^{nov} + x_{21}^{nov} + y_1^{ott} = 9000 + y_1^{nov} \\
& x_{11}^{dic} + x_{21}^{dic} + y_1^{nov} = 9000 \\
& x_{12}^{ott} + x_{22}^{ott} = 8000 + y_2^{ott} \\
& x_{12}^{nov} + x_{22}^{nov} + y_2^{ott} = 8000 + y_2^{nov} \\
& x_{12}^{dic} + x_{22}^{dic} + y_2^{nov} = 8000. \\
& x_{ij}^h \geq 0 \quad i = 1, 2, j = A, B, h = ott, nov, dic
\end{aligned}$$

Indichiamo con

- **PROD** l'insieme dei prodotti (quindi p.es. $\text{PROD} = \{\text{TipoA}, \text{TipoB}\}$);
- **RIS** l'insieme di risorse necessarie alla produzione; (p.es. lavorazione sulle linee L1 e L2)
- **T** il numero dei mesi di interesse (p.es. tre, novembre, dicembre e gennaio).

Ricordiamo che le variabili di decisione sono state scelte come

$x_{ij}^h \equiv$ quantità di pneumatici di tipo j in **PROD** prodotti usando la i -esima linea con i in **RIS** nel periodo h -esimo ($h = 1..T$)

$y_i^h \equiv$ quantità di prodotto i immagazzinato nel periodo h -esimo ($h = 1..T-1$)

Inoltre, sono note le seguenti quantità

- il costo $c_r[j, i]$ di risorsa j per produrre una unità di i ;
- il costo $c_m[i]$ di produzione di una unità di prodotto i ;
- il costo $c_i[i]$ di immagazzinamento di una unità di prodotto i ;
- la disponibilità $disp[j, h]$ di risorsa j nel periodo h ;
- la quantità $a[j, i]$ di risorsa j per produrre una unità di i ;
- la domanda $dom[i, h]$ di prodotto i nel periodo h ;

A questo punto vediamo come scrivere il problema appena descritto in AMPL Plus. Prima di passare alla scrittura vera e propria del file di modello (**.mod**) di AMPL, è interessante notare il fatto che il problema descritto come abbiamo fatto sopra può essere considerato "parametrico" nel senso che nessun dato numerico è stato specificato esplicitamente. Tuttavia,

per formulare matematicamente un problema di programmazione matematica, non abbiamo alcun bisogno di conoscere i dati numerici della particolare istanza del nostro problema (specificando quindi quali sono i prodotti, quali le risorse, quanti sono i mesi di interesse, ecc.).

AMPL Plus ci offre la possibilità di scrivere un modello di problema senza preoccuparci di specificarne i dati numerici. Una volta scritto il file del modello "parametrico", per risolvere il problema occorre assegnare ai parametri del problema i valori numerici specifici dell'istanza di problema che vogliamo risolvere.

Vediamo ora come scrivere in AMPL il modello del problema di produzione multiperiodo.

```

multiper.mod
-----

set PROD;
set RIS;

param T > 0;
param c_m{PROD};
param c_i{PROD};
param disp{RIS,1..T};
param a{RIS,PROD};
param c_r{RIS,PROD};
param dom{PROD,1..T};

var x{i in PROD,h in 1..T, j in RIS} >= 0;;
var y{i in PROD, h in 1..T-1} >= 0;

minimize costi:
    sum{i in PROD,j in RIS} (c_r[j,i]*(sum{h in 1..T} x[i,h,j]))
+sum{i in PROD}      c_m[i]*(sum{j in RIS, h in 1..T} x[i,h,j])
+sum{i in PROD}      c_i[i]*(sum{h in 1..T-1} y[i,h]);

s.t. vincolo_prod{j in RIS,h in 1..T}:
    sum{i in PROD} a[j,i]*x[i,h,j]<= disp[j,h];
s.t. balance1{i in PROD}:
    sum{j in RIS} x[i,1,j]= dom[i,1]+y[i,1];
s.t. balance2{i in PROD, h in 2..T-1}:
    sum{j in RIS} x[i,h,j]+y[i,h-1]= dom[i,h]+y[i,h];
s.t. balance3{i in PROD}:
    sum{j in RIS} x[i,T,j]+y[i,T-1]= dom[i,T];

```

Ora analizziamo le istruzioni del file `multiper.mod`. Anzitutto notiamo le istruzioni

```

set PROD;
set RIS;

```

con le quali si dichiarano `PROD` e `RISORSE` come insiemi ("set") di un numero imprecisato di elementi (prodotti e risorse). Subito dopo abbiamo sette istruzioni che ci servono per definire altrettanti parametri del modello. La prima di queste

```

param T > 0;

```

definisce un parametro `T` strettamente positivo che indicherà il numero di periodi rispetto a cui pianificare. Subito dopo troviamo le due istruzioni

```

param c_m{PROD};
param c_i{PROD};

```

che definiscono due vettori (c_m e c_i) di parametri con tante componenti quanti sono gli elementi dell'insieme PROD. Le istruzioni `param` successive ovvero

```

param disp{RIS,1..T};
param a{RIS,PROD};
param c_r{RIS,PROD};
param dom{PROD,1..T};

```

definiscono invece quattro matrici di parametri. Ciascuna matrice ha un numero di righe pari alla cardinalità del primo insieme nella sua definizione e numero di colonne pari alla cardinalità del secondo. È possibile accedere a ciascuna componente di un vettore o di una matrice come si accede alle componenti di un array in C e cioè specificando il nome del parametro, p.es. “a”, seguito dagli indici della componente che si vuole selezionare racchiusi tra parentesi quadre (p.es. $a[1,2]$, $a[2,3]$, ecc. oppure, nel caso del vettore “dom”, $dom[2]$).

La funzione obiettivo `costi`, definita dalla istruzione

```

minimize costi:
    sum{i in PROD,j in RIS} (c_r[j,i]*(sum{h in 1..T} x[i,h,j]))
+sum{i in PROD}      c_m[i]*(sum{j in RIS, h in 1..T} x[i,h,j])
+sum{i in PROD}      c_i[i]*(sum{h in 1..T-1} y[i,h]);

```

è una somma di costi relativi a:

- l'acquisto delle risorse necessarie alla produzione;
- il funzionamento delle catene produttive dei singoli prodotti;
- i costi di immagazzinamento dei prodotti nel magazzino merci.

Infine è interessante notare le istruzioni che definiscono i vincoli del problema

```

s.t. vincolo_prod{j in RIS,h in 1..T}:
    sum{i in PROD} a[j,i]*x[i,h,j]<= disp[j,h];
s.t. balance1{i in PROD}:
    sum{j in RIS} x[i,1,j]= dom[i,1]+y[i,1];
s.t. balance2{i in PROD, h in 2..T-1}:
    sum{j in RIS} x[i,h,j]+y[i,h-1]= dom[i,h]+y[i,h];
s.t. balance3{i in PROD}:
    sum{j in RIS} x[i,T,j]+y[i,T-1]= dom[i,T];

```

con le quali si definiscono array e matrici di vincoli utilizzando la stessa sintassi già vista per la dichiarazione di array e matrici di parametri. Nell'esempio precedente è ricorrente l'uso della espressione

```
{i in P}
```

ove P nel nostro caso può essere PROD, RIS oppure 1..T. Questo costrutto è detto *espressione di indicizzazione* ed è utilizzato non solo per definire variabili e parametri ma anche, come visto, per le sommatorie, per definire i vincoli del problema e comunque ogni qual volta nel modello matematico sia naturale l'uso del quantificatore *per ogni* (\forall).

Supponiamo ora di voler risolvere il seguente caso particolare di problema di produzione multiperiodo. Quello che facciamo per particularizzare il problema è specificarne tutti i parametri. In AMPL Plus questo si risolve nella creazione di un file dei dati (con estensione `.dat`) che, compilato insieme al file del modello, ci consente di risolvere l'istanza di problema considerato. Vediamo quindi come scrivere il file dei dati di AMPL in cui specificare il valore

dei parametri che compaiono nel file del modello. In prima approssimazione supponiamo che la domanda si mantenga costante nell'arco di tempo considerato (che é quanto di piú semplice possa fare l'industria per pianificare la propria produzione).

```

multipercost.dat
-----

set PROD:= TipoA, TipoB;
set RIS:= L1, L2;

param T=3;

param disp:      1      2      3 :=
    L1      2000    400    200
    L2      3000    800    1000 ;

param a:      TipoA    TipoB :=
    L1      0.10     0.12
    L2      0.12     0.18 ;

param c_r:      TipoA    TipoB :=
    L1      600      720
    L2      720      1080 ;

param: c_m :=
    TipoA    2500
    TipoB    4000;

param: c_i :=
    TipoA    500
    TipoB    500;

param  dom:      1      2      3 :=
    TipoA    9000    9000    9000
    TipoB    8000    8000    8000 ;

```

Facciamo notare che in questo modo, avendo cioè a disposizione il file `multiper.mod` contenente il modello separato dai dati del problema (contenuti invece nel file `multipercost.dat`), possiamo risolvere differenti problemi di pianificazione multiperiodo semplicemente cambiando i dati contenuti nel file con estensione `.dat`.

Supponiamo infatti di avere delle domande variabili nei mesi su cui si sta operando (piuttosto che costanti) e precisamente supponiamo di avere le domande riportate nella seguente tabella:

| | Novembre | Dicembre | Gennaio |
|-------|----------|----------|---------|
| TipoA | 16000 | 7000 | 4000 |
| TipoB | 14000 | 4000 | 6000 |

Lo studente provi a risolvere il nuovo problema di produzione multiperiodo modificando o, se necessario, riscrivendo il file del modello e/o quello dei dati in AMPL Plus.

3 Sintassi di AMPL

In questa sezione introdurremo, spiegandola il più possibile, la sintassi dei principali costrutti di AMPL. Tuttavia, non potendo e, soprattutto, non volendo questo essere un manuale di AMPL, utilizzeremo allo scopo di commentare le istruzioni più significative, alcuni semplici esempi di problemi di programmazione matematica lineare.

Cominciamo quindi descrivendo brevemente il problema del “*Lot Sizing*” (dimensionamento delle scorte). Consideriamo a questo scopo il caso di una industria che deve pianificare su un certo numero di mesi la propria produzione di un non meglio specificato bene di consumo, potendo contare mensilmente sulla disponibilità di un magazzino merci in cui stoccare mese per mese le quantità rimaste invendute. Il magazzino è vuoto all’inizio del primo mese e deve rimanere vuoto alla fine del periodo di interesse. Mese per mese è nota la quantità \mathbf{dom}_i di prodotti che il mercato è in grado di assorbire. Oltre a questo, come è ragionevole aspettarsi, l’industria sostiene dei costi \mathbf{cprod}_i e \mathbf{cstoc}_i (variabili con i mesi) per produrre i suoi articoli e per stoccare in magazzino i pezzi non venduti.

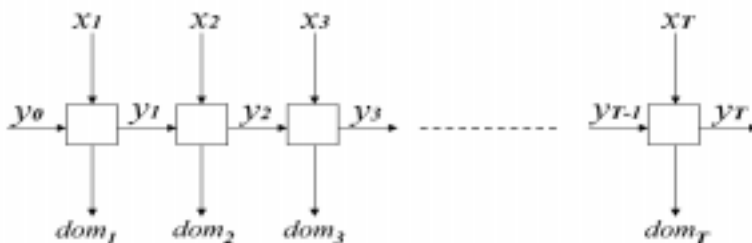


Figura 1

In figura 1, avendo ipotizzato un orizzonte temporale di T mesi, è stato schematizzato il problema e allo stesso tempo sono state indicate le variabili di decisione che servono per poter formulare matematicamente il problema del Lot Sizing. Supponendo di voler minimizzare i costi sull’intero orizzonte temporale di T mesi, abbiamo il seguente problema di PL

$$\begin{aligned}
 \min \quad & \sum_{i=1}^T \mathbf{cprod}_i x_i + \sum_{i=1}^{T-1} \mathbf{cstoc}_i y_i \\
 \text{s.t.} \quad & y_{i-1} - y_i = \mathbf{dom}_i - x_i \quad \forall i \in \{1, \dots, T\} \\
 & y_0 = 0, \quad y_T = 0
 \end{aligned} \tag{3}$$

Senza la necessità di ulteriori informazioni, quindi senza bisogno che siano specificati

- il numero T di mesi di interesse,
- le domande \mathbf{dom}_i del mercato nei vari mesi e
- i costi \mathbf{cprod}_i di produzione e quelli \mathbf{cstoc}_i di stoccaggio,

possiamo già scrivere in AMPL un modello del problema del Lot Sizing. Ovviamente però tale modello non potrà essere risolto fino a quando, specificandone tutti i dati (in un file separato), il problema generale non sarà stato *particolarizzato* ad una ben precisa istanza del problema di Lot Sizing (p.es. dicendo che i mesi sono 3 ecc.).

Il file AMPL contenente la descrizione del modello è il seguente

```

_____ lotsizing1.mod _____
param T integer > 1;

var x{1..T} >= 0;

```

```

var y{0..T} >= 0;

param dom{1..T};
param cprod{1..T};
param cstoc{1..T-1};

minimize cost: ( sum{i in 1..T}cprod[i]*x[i])
                + (sum{i in 1..T-1}cstoc[i]*y[i]) ;

s.t. balance{i in 1..T}: y[i-1] + x[i] = y[i] + dom[i];
s.t.      inizio:      y[0] = 0;
s.t.      fine:       y[T] = 0;

```

Supponiamo a questo punto che l'industria, per ragioni sanitarie, voglia svuotare completamente i suoi magazzini ogni cinque mesi oltre che alla fine del periodo di studio. Per tener conto di questo nuovo vincolo, occorre modificare il modello aggiungendo un nuovo tipo di vincoli, come riportato di sotto per il file `lotsizing2.mod`

```

lotsizing1.mod
.
.
s.t. balance{i in 1..T}: y[i-1] + x[i] = y[i] + dom[i];
s.t.      inizio:      y[0] = 0;
s.t.      fine:       y[T] = 0;
s.t. vincoloNew{i in 0..T by 5}: y[i] = 0;

```

in cui si è usato il costrutto

```
0..T by 5
```

per indicare l'insieme degli interi da 0 a T a cinque a cinque cioè l'insieme costituito dai numeri 0,5,10,15,....

Fin ora, abbiamo implicitamente rappresentato l'insieme dei mesi di interesse con la notazione

```
1..T
```

e questo è stato sufficiente per il semplice fatto che riuscivamo, in vari modi, a selezionare tutti i mesi "particolari" del nostro modello, quelli cioè che richiedevano l'inserimento di vincoli ad hoc.

Se però tra i vincoli del problema ci fosse stato il vincolo aggiuntivo che in un mese particolare (p. es. in **marzo**) il magazzino deve rimanere vuoto, avremmo dovuto riscrivere completamente il modello apportando al file `lotsizing1.mod` modifiche ben più profonde di quelle viste prima. In particolare, avremmo dovuto scrivere il nostro modello AMPL come riportato di seguito.

```
lotsizing2.mod
```

```

set MESI ordered;
param mesePart symbolic in MESI;

var x{MESI} >= 0;
var y{0..card{MESI}} >= 0;

```

```

param dom{MESI};
param cprod{MESI};
param cstoc{MESI diff {last(MESI)}};

minimize cost: ( sum{i in MESI}cprod[i]*x[i])
                + (sum{i in MESI diff {last(MESI)}}cstoc[i]*y[ord(i,MESI)]) ;

s.t. balance{i in MESI}:
      y[ord(i,MESI)-1] + x[i] = y[ord(i,MESI)] + dom[i];
s.t. inizio:                y[0] = 0;
s.t. fine:                  y[card(MESI)] = 0;
s.t. vuoto: y[ord(mesePart,MESI)] = 0;

```

È importante, nel commentare il file `lotsizing2.mod`, sottolineare il fatto che ancora non conosciamo

- ne da quali elementi (mesi) è composto l'insieme `MESI`,
- ne, tanto meno, quale è il mese particolare in cui il magazzino dovrà rimanere vuoto.

Tutti questi elementi saranno noti solamente quando ci verrà dato, o scriveremo noi, un file dei dati per il modello, particolarizzando quindi il problema generale ad un caso specifico di *lot sizing*.

Nel file `lotsizing.mod` notiamo anche la presenza di alcuni nuovi comandi quali ad esempio l'uso dell'istruzione

```
set MESI ordered;
```

che dichiara `MESI` come insieme ordinato di elementi non meglio specificati. Poi notiamo un'altra istruzione nuova e precisamente

```
param mesePart symbolic in MESI
```

che dichiara un parametro del modello `mesePart` che indica un elemento nell'insieme `MESI`. Torniamo a ripetere che, sia gli elementi dell'insieme `MESI` che il valore del parametro `mesePart`, dovranno essere specificati, separatamente, nel momento in cui verrà scritto il file dei dati del modello.

In secondo luogo, notiamo la dichiarazione dell'array di variabili `y`

```
var y{0..card{MESI}};
```

in cui si è usata l'istruzione `card(MESI)` che restituisce il numero di elementi dell'insieme `MESI`. Inoltre, avendo dichiarato `MESI` come insieme ordinato, su di esso è possibile fare tutta una serie di operazioni, alcune delle quali sono state usate nel file `lotsizing2.mod`, che sono

- `first(MESI)` e `last(MESI)` che ritornano il primo e rispettivamente l'ultimo elemento dell'insieme `MESI`,
- `prev(t,MESI)` e `next(t,MESI)` che restituiscono l'elemento dell'insieme `MESI` che precede e rispettivamente segue (nell'ordinamento) l'elemento `t`,
- `next(t,MESI,n)` e `prev(t,MESI,n)` che danno, rispettivamente, l'`n`-esimo elemento in `MESI` dopo e prima di `t`,
- `ord(t,MESI)` che restituisce la posizione dell'elemento `t` nell'insieme ordinato `MESI`.

Quindi se l'insieme `MESI` fosse il seguente

```
set MESI := dicembre novembre marzo febbraio luglio giugno agosto;
```

- `first(MESI)` darebbe dicembre,
- `last(MESI)` darebbe agosto,
- `next(marzo,MESI,2)` darebbe luglio,
- `prev(marzo,MESI,2)` darebbe dicembre,
- `ord(febbraio,MESI)` darebbe 4.

Continuando ad analizzare il file di modello, notiamo che, nella definizione del parametro `cstoc` e della funzione obiettivo, si è usato il costrutto

```
MESI diff {last(MESI)}
```

che restituisce l'insieme *differenza* tra `MESI` ed il suo ultimo elemento. Oltre alla differenza, dati due insiemi `A` e `B` è possibile fare in AMPL altre operazioni come

- `A union B` \equiv insieme degli elementi che stanno in `A` o `B`,
- `A inter B` \equiv insieme degli elementi che stanno in `A` e `B`,
- `A diff B` \equiv insieme degli elementi che stanno in `A` e non in `B`,
- `A symdiff B` \equiv insieme degli elementi che stanno in `A` o `B` ma non in entrambi.

Ora scriviamo un file di dati per il modello contenuto in `lotsizing2.mod`.

```

                                lotsizing2.dat
-----
param :MESI:          dom   cprod :=
    febbraio         300   100
        marzo         300   100
        aprile        400   100
        maggio        400   500
        giugno        500   300
        luglio        400   300
        agosto        400   400
    settembre        500   500
    ottobre          300   500;

param mesePart := maggio;

param:          cstoc :=
    febbraio    50
        marzo    50
        aprile   60
        maggio   70
        giugno   60
        luglio   80
        agosto   60
    settembre   60;

```

Nel file di dati precedente, ci sembra opportuno mettere in evidenza il modo in cui sono stati inizializzati l'insieme `MESI` e i vettori `dom` e `cprod`. Infatti, la loro inizializzazione è stata fatta contemporaneamente in una unica istruzione `param`, piuttosto che inizializzare `MESI` separatamente nel modo seguente.

```
set MESI :=    febbraio, marzo, aprile, maggio, giugno,
              luglio, agosto, settembre, ottobre;
```

Esercizio: *Come bisogna modificare il modello e/o i dati per tener conto del fatto che non si vuole produrre niente nell'ultimo mese? Il valore dell'ottimo si modifica?*

Come bisogna invece modificare il modello e/o i dati per tener conto del fatto che non si vuole produrre niente nel mese di giugno? Il valore dell'ottimo si modifica?

Ora introduciamo nel problema una ulteriore complicazione e cioè supponiamo che l'industria fabbrichi non un unico prodotto ma un certo insieme di prodotti. Indichiamo con **PROD** l'insieme dei prodotti. Supponiamo inoltre che l'industria sopporti costi di produzione che, oltre a dipendere dai mesi, dipendono anche dal prodotto considerato e che si vogliano soddisfare domande dipendenti, anche queste, sia dai mesi che dal prodotto. Per tenere conto di queste modifiche, dobbiamo aggiornare il file del modello in modo da ottenere il seguente file `lotsizing3.mod`.

```

lotsizing3.mod
-----
set MESI ordered;
set PROD;

param mesePart symbolic in MESI;

var x{PROD,MESI} >= 0;
var y{PROD,0..card(MESI)} >= 0;

param dom{PROD,MESI};
param cprod{PROD,MESI};
param cstoc{MESI diff {last(MESI)}};

minimize cost:
  (sum{i in MESI}sum{j in PROD}cprod[j,i]*x[j,i])
  +(sum{i in MESI diff {last(MESI)}}
    cstoc[i]*sum{j in PROD}y[j,ord(i,MESI)]) ;

s.t. balance{j in PROD,i in MESI}:
      y[j,ord(i,MESI)-1] - y[j,ord(i,MESI)] = dom[j,i] - x[j,i];
s.t. inizio{j in PROD}:                y[j,0] = 0;
s.t. fine{j in PROD}:                  y[j,card(MESI)] = 0;
s.t. vuoto{j in PROD}: y[j,ord(mesePart,MESI)] = 0;

```

Un possibile file dei dati per il precedente modello dovrà per tanto prevedere gli assegnamenti, oltre che per l'insieme **PROD**, anche per le matrici di parametri **cprod** e **dom**, come nel seguente file `lotsizing3.dat`.

```

lotsizing3.dat
-----
set MESI := febbraio, marzo, aprile, maggio, giugno,
          luglio, agosto, settembre, ottobre;
set PROD := normale, lusso;

param mesePart := maggio;

param dom(tr): normale      lusso      :=
  febbraio    300          200
  marzo       300          200
  aprile      400          300
  maggio      400          300
  giugno      500          400

```

```

    luglio      400      300
    agosto      400      300
settembre      500      400
    ottobre     300      200;

param cprod(tr): normale      lusso      :=
    febbraio    100      200
    marzo       100      200
    aprile      100      300
    maggio      100      200
    giugno      200      300
    luglio      400      300
    agosto      400      300
settembre      500      400
    ottobre     600      500;

param:      cstoc :=
    febbraio  50
    marzo     50
    aprile    60
    maggio    70
    giugno    60
    luglio    80
    agosto    60
settembre    60;

```

Anche qui ci sembra opportuno discutere brevemente il modo in cui in un file di dati vengono assegnati valori ad un vettore o una matrice. La sintassi di AMPL prevede, per assegnare valori ad un vettore (p.es. `cstoc`), la seguente istruzione:

```

param:  cstoc := febbraio 50 marzo 50 aprile 60 maggio 70
        giugno 60 luglio 80 agosto 60 settembre 60;

```

che però, opportunamente indentata restituisce la ben più leggibile forma

```

param:  cstoc :=
    febbraio  50
    marzo     50
    aprile    60
    maggio    70
    giugno    60
    luglio    80
    agosto    60
    settembre 60;

```

Per una matrice, invece, il discorso è leggermente più complicato. Infatti l'istruzione standard per assegnare valori p.es. a `cprod` sarebbe la seguente:

```

param:  cprod := normale febbraio 100 normale marzo 100 normale aprile 100
        normale maggio 100 .... normale ottobre 600
        lusso febbraio 200 lusso marzo 200 lusso aprile 300
        lusso maggio 200 .... lusso ottobre 500;

```

che prevede di specificare tutte le componenti mediante indicazione del primo indice, secondo indice e valore. Questo metodo presenta degli ovvi svantaggi, non ultimo quello di dover ripetere molte volte gli stessi indici. Per questo motivo è prevista anche la più concisa notazione seguente

```

param cprod(tr): normale      lusso      :=
    febbraio      100        200
    marzo         100        200
    aprile        100        300
    maggio        100        200
    giugno        200        300
    luglio        400        300
    agosto        400        300
    settembre     500        400
    ottobre       600        500;

```

(tr) serve per indicare che la matrice verrà data in forma trasposta. In pratica, è come se la matrice venisse inserita per colonne (o righe nel caso della trasposta). Per finire, descriviamo brevemente il caso in cui si debbano assegnare dei valori ad un parametro con tre o più dimensioni. Per non appesantire troppo la scrittura, supponiamo di avere un parametro pippo a tre dimensioni cioè:

```

set DIM1;
set DIM2;
set DIM3;
param pippo{DIM1,DIM2,DIM3};

```

e supponiamo che i tre insiemi siano definiti nel file dei dati come

```

set DIM1 := 1 2 3;
set DIM2 := A B C;
set DIM3 := A1 B2 C3;

```

Oltre al metodo tradizionale di assegnare valori al parametro, ovvero, come visto nel caso di matrici a due dimensioni

```

param: pippo := 1 A A1 7   1 A B2 8   1 A C3 9
                1 B A1 6   1 B B2 7   1 B C3 8
                ...       ...       ...
                3 C A1 1   3 C B2 2   3 C C3 3;

```

anche qui è prevista una notazione più concisa. In pratica quello che si fa è “*affettare*” il parametro in matrici di dimensione due e quindi assegnare valori in maniera molto simile a quanto visto prima, ovvero

```

param pippo:=
    [1,*,*]:  A1  B2  C3 :=
              A   10  20  30
              B   15  25  35
              C   40  50  60

    [2,*,*]:  A1  B2  C3 :=
              A   17  23  29
              B   10  20  30
              C   10  29  35

```

```
[3,*,*]:  A1  B2  C3 :=
          A   1   2   3
          B   7   5   9
          C  10  10  10 ;
```

Esercizio: Modificare il file `lotsizing3.mod` per tener conto del fatto che:

- il magazzino ha una capacità limitata pari a `cap`;
- il magazzino ha una capacità limitata e dipendente dal mese;
- il magazzino ha una capacità limitata e dipendente dal prodotto;
- il magazzino ha una capacità limitata e dipendente sia dal prodotto che dal mese.

Esercizio: il problema della dieta.

Si consideri il problema di voler scegliere in che quantità assumere certi cibi in maniera tale da soddisfare, per alcuni nutrienti (vitamine) base, dei requisiti minimi e non superando dei limiti massimi, minimizzando il costo complessivo della dieta. Nella tabella che segue si riportano, per ciascun alimento, il costo in euro e il contenuto percentuale dei nutrienti di interesse.

| alimento | costo | A | C | B1 | B2 |
|-----------------|-------|----|----|----|----|
| manzo | 3.19 | 60 | 20 | 10 | 15 |
| pollo | 2.59 | 8 | 0 | 20 | 20 |
| pesce | 2.29 | 8 | 10 | 15 | 10 |
| prosciutto | 2.89 | 40 | 40 | 35 | 10 |
| pasta al sugo | 1.89 | 15 | 35 | 15 | 15 |
| polpettone | 1.99 | 70 | 30 | 15 | 15 |
| pasta in bianco | 1.99 | 25 | 50 | 25 | 15 |
| tacchino | 2.49 | 60 | 20 | 15 | 10 |

La dieta deve essere tale da soddisfare i requisiti massimi e minimi (in percentuale) riportati in tabella:

| max | | min | |
|-----|----|-----|----|
| A | B2 | C | B1 |
| 15 | 20 | 25 | 20 |

Scrivere in AMPL un file di modello ed un file di dati che rappresentino correttamente il problema della dieta. Nello svolgere l'esercizio, lo studente cerchi di rendere il modello quanto più possibile generale e indipendente dai dati del problema.

4 Richiami sull'assegnazione di valori ai Parametri

Ci sembra opportuno richiamare brevemente il modo in cui in un file di dati (con estensione `.dat`) vengono assegnati valori ad un vettore di parametri ad una, due o più dimensioni. Procediamo per gradi e, per cominciare, supponiamo che nel file del modello siano presenti le istruzioni

```
set DIM;  
param vett{DIM};
```

che dichiarano `DIM` come insieme e `vett` come vettore di parametri ad una dimensione indicizzato sugli elementi di `DIM`. Per assegnare valori al parametro `vett`, nel file dei dati, la sintassi di AMPL prevede la seguente istruzione:

```
set DIM := pippo pluto paperino topolino paperoga paperone  
          clarabella qui quo qua  
param: vett := pippo 50 pluto 50 paperino 60 topolino 70  
          paperoga 60 paperone 80 clarabella 60 qui 60  
          quo 75 qua 85;
```

ove, per maggiore chiarezza, abbiamo riportato la definizione dell'insieme `DIM`. In pratica, quindi, per assegnare valori ad un parametro dobbiamo specificare le coppie indice–valore (nel caso monodimensionale). Le precedenti istruzioni, opportunamente indentate, restituiscono la ben più leggibile forma seguente

```
param:      vett :=  
    pippo   50  
    pluto   50  
    paperino 60  
    topolino 70  
    paperoga 60  
    paperone 80  
    clarabella 60  
    qui     60  
    quo     75  
    qua     85;
```

Se volessimo assegnare valori a due parametri definiti sullo stesso insieme di indici, oltre al ripetere due volte l'istruzione vista prima, possiamo, in maniera più compatta, scrivere

```
param:      vett1 vett2 :=  
    pippo   50  20  
    pluto   50  20  
    paperino 60  10  
    topolino 70  40  
    paperoga 60  33  
    paperone 80  36  
    clarabella 60  78  
    qui     60  95  
    quo     75  12  
    qua     85  54;
```

in cui, in pratica, bisogna specificare delle triple di valori: indice, valore da assegnare a `vett1`, valore da assegnare a `vett2`.

Per una matrice (parametro a due dimensioni), invece, il discorso è leggermente più complicato. Infatti l'istruzione standard per assegnare valori p.es. a `mat` definito come

```

set DIM1;
set DIM2;
param mat{DIM1,DIM2};

```

sarebbe la seguente:

```

set DIM1 := A B C
set DIM2 := pippo pluto paperino topolino paperoga paperone
           clarabella archimede
param: mat := A pippo 100 A pluto 100 A paperino 100
           A topolino 100 ... A archimene 600
           B pippo 200 ... B archimene 400
           C pippo 200 ... C archimede 500;

```

che prevede di specificare tutte le componenti mediante indicazione del primo indice, secondo indice e valore. Questo metodo presenta degli ovvi svantaggi, non ultimo quello di dover ripetere molte volte gli stessi indici. Per questo motivo è prevista anche la più concisa notazione seguente

```

param cprod(tr):   A      B      C      :=
      pippo      100    200    200
      pluto      100    200    100
      paperino   100    300    100
      topolino   100    200    100
      paperoga   200    300    300
      paperone   400    300    500
      clarabella 400    300    600
      archimede  600    400    500 ;

```

(**tr**) serve per indicare che la matrice verrà data in forma trasposta (come in questo caso). In pratica, è come se la matrice venisse inserita per colonne (o righe nel caso della trasposta). Per finire, descriviamo brevemente il caso in cui si debbano assegnare dei valori ad un parametro con tre o più dimensioni. Per non appesantire troppo la scrittura, supponiamo di avere un parametro **etabeta** a tre dimensioni cioè:

```

set DIM1;
set DIM2;
set DIM3;
param etabeta{DIM1,DIM2,DIM3};

```

e supponiamo che i tre insiemi siano definiti nel file dei dati come

```

set DIM1 := 1 2 3;
set DIM2 := A B C;
set DIM3 := A1 B2 C3;

```

Oltre al metodo tradizionale di assegnare valori al parametro, ovvero, come visto nel caso di matrici a due dimensioni

```

param: etabeta := 1 A A1 7   1 A B2 8   1 A C3 9
                1 B A1 6   1 B B2 7   1 B C3 8
                ...      ...      ...
                3 C A1 1   3 C B2 2   3 C C3 3;

```

anche qui è prevista una notazione più concisa. In pratica quello che si fa è “*affettare*” il parametro in matrici di dimensione due e quindi assegnare valori in maniera molto simile a quanto visto prima, ovvero

```

param etabeta :=
  [1,*,*]:  A1  B2  C3 :=
            A   7  8  9
            B   6  7  8
            C  40 50 60

  [2,*,*]:  A1  B2  C3 :=
            A  17 23 29
            B  10 20 30
            C  10 29 35

  [3,*,*]:  A1  B2  C3 :=
            A   1  2  3
            B   7  5  9
            C   1  2  3 ;

```

Notiamo che nella seconda modalità di assegnamento di valori al parametro **etabeta**, non dobbiamo mettere il simbolo $\boxed{:=}$ dopo la parola chiave **param**.

5 Esercizio

Una industria metalmeccanica produce tre tipi di acciaio (normale, inox, temperato) utilizzando quattro macchine (M1, M2, M3, M4). Le macchine sono attive per otto ore al giorno. La tabella che segue riporta, per ciascuna macchina, il tempo necessario (in ore) per lavorare una tonnellata di acciaio.

| | M1 | M2 | M3 | M4 |
|-----------|-----|-----|------|------|
| normale | 0.1 | 0.2 | 0.05 | 0.05 |
| inox | 0.5 | 0.6 | 0.4 | 0.5 |
| temperato | 0.3 | 0.5 | 0.2 | 0.3 |

Supponendo infine che tutto l'acciaio prodotto viene assorbito dal mercato fruttando all'azienda i seguenti profitti (in milioni di lire per tonnellata)

| | normale | inox | temperato |
|-----------------------|---------|------|-----------|
| profitto (mln £/ton.) | 6 | 25 | 20 |

Si scriva in AMPL un problema di PL che massimizzi i profitti giornalieri dell'industria. Come prima cosa dobbiamo ovviamente individuare le *variabili di decisione* del problema. Introduciamo le seguenti variabili

$x_{mp} \equiv$ tonnellate di acciaio di tipo p lavorate sulla macchina m , ove p varia in {normale, inox, temperato} e m in {M1, M2, M3, M4}.

Indicando con

$tprod_{mp}$ il tempo necessario per lavorare sulla macchina m una tonnellata di acciaio di tipo p ,
 $tlim$ il tempo limite di utilizzo delle macchine (pari ad 8 ore),
 $prezzo_p$ il prezzo di una tonnellata di acciaio di tipo p

possiamo scrivere la seguente formulazione matematica del problema in esame:

$$\max \sum_p \text{prezzo}_p \sum_m x_{mp}$$

$$\sum_p \text{tprod}_{m,p} x_{mp} \leq \text{tlim} \quad \text{per ogni } m$$

Vediamo subito come scrivere in AMPL, *separando il modello dai dati*, il problema appena visto. Il file AMPL contenente la descrizione del modello è il seguente

```

acciaio.mod
-----
set PROD;
set MACCHINE;

param tprod{MACCHINE,PROD};
param tlim;
param prezzo{PROD};

var x{MACCHINE,PROD} >= 0;

maximize profit: sum{p in PROD}prezzo[p]*
                sum{m in MACCHINE}x[m,p];

s.t. tempo{m in MACCHINE}:
     sum{p in PROD}tprod[m,p]*x[m,p] <= tlim;

```

Ora scriviamo il file di dati per il modello contenuto in acciaio.mod.

```

acciaio.dat
-----

set    PROD := normale inox temperato;
set    MACCHINE := M1 M2 M3 M4;

param:    prezzo :=
    normale 6
    inox 25
    temperato 20 ;

param    tlim := 8;

param    tprod :    normale    inox    temperato :=
    M1 0.1    0.5    0.3
    M2 0.2    0.6    0.5
    M3 0.05   0.4    0.2
    M4 0.05   0.5    0.3 ;

```

Il valore della funzione obiettivo all'ottimo è in questo caso: 2786.667

Per evitare i lunghi tempi di avvio delle macchine, supponiamo che l'industria decida di farle lavorare 24 ore su 24 dividendo la giornata (di 24 ore) in tre turni lavorativi: 8-16, 16-24, 24-8. La tabella seguente riporta, per ogni turno di lavoro e per ogni macchina il tempo necessario (in ore) per lavorare una tonnellata di acciaio.

| | 8-16 | | | | 16-24 | | | | 24-8 | | | |
|-----------|------|-----|------|------|-------|------|------|------|------|------|------|------|
| | M1 | M2 | M3 | M4 | M1 | M2 | M3 | M4 | M1 | M2 | M3 | M4 |
| normale | 0.1 | 0.2 | 0.05 | 0.05 | 0.2 | 0.3 | 0.09 | 0.09 | 0.3 | 0.4 | 0.12 | 0.14 |
| inox | 0.5 | 0.6 | 0.4 | 0.5 | 0.55 | 0.65 | 0.45 | 0.5 | 0.55 | 0.65 | 0.45 | 0.5 |
| temperato | 0.3 | 0.5 | 0.2 | 0.3 | 0.41 | 0.6 | 0.27 | 0.38 | 0.41 | 0.6 | 0.27 | 0.38 |

Tuttavia, per questioni sindacali, ciascuna macchina può lavorare per un tempo limitato e dipendente dal turno di lavoro, come riportato nella tabella che segue.

| | 8-16 | 16-24 | 24-8 |
|----|------|-------|------|
| M1 | 7 | 6 | 7 |
| M2 | 7 | 7 | 5 |
| M3 | 7 | 7 | 4 |
| M4 | 7 | 5 | 4 |

Esercizio: Come bisogna modificare i file del modello e dei dati per tenere in considerazione le nuove specifiche del problema?

Questa volta le *variabili di decisione* del problema, come pure i parametri **tprod** e **tlim**, devono tenere in considerazione i turni di lavoro, quindi:

| |
|--|
| $x_{mp}^t \equiv$ tonnellate di acciaio di tipo p lavorate sulla macchina m nel turno t , ove p varia in {normale, inox, temperato}, m in {M1, M2, M3, M4} e t in {8-16, 16-24, 24-8}. |
|--|

| |
|--|
| $tprod_{mp}^t$ il tempo necessario per lavorare sulla macchina m una tonnellata di acciaio di tipo p nel turno t , |
| $tlim_m^t$ il tempo limite di utilizzo della macchina m nel turno t |

Quindi, volendo scrivere in AMPL il problema così modificato, dovremmo scrivere i due file seguenti

```

_____ acciaio1.mod _____

set TURNI;
set PROD;
set MACCHINE;

param tprod{MACCHINE,TURNI,PROD};
param tlim{MACCHINE,TURNI};
param prezzo{PROD};

var x{MACCHINE,TURNI,PROD} >= 0;

maximize profit: sum{p in PROD}prezzo[p]*
                sum{m in MACCHINE,t in TURNI}x[m,t,p];

s.t. tempo{m in MACCHINE,t in TURNI}:
    sum{p in PROD}tprod[m,t,p]*x[m,t,p] <= tlim[m,t];

_____
_____ acciaio1.dat _____

set TURNI := '8-16' '16-24' '24-8';
set PROD := normale inox temperato;
set MACCHINE := M1 M2 M3 M4;

param: prezzo :=

```

```

normale 6
inox 25
temperato 20 ;

param tlim: '8-16' '16-24' '24-8' :=
M1 7 6 7
M2 7 7 5
M3 7 7 4
M4 7 5 4 ;

param tprod :=
[*,'8-16',*]: normale inox temperato :=
M1 0.1 0.5 0.3
M2 0.2 0.6 0.5
M3 0.05 0.4 0.2
M4 0.05 0.5 0.3
[*,'16-24',*]: normale inox temperato :=
M1 0.2 0.55 0.41
M2 0.3 0.65 0.6
M3 0.09 0.45 0.27
M4 0.09 0.5 0.38
[*,'24-8',*]: normale inox temperato :=
M1 0.3 0.55 0.41
M2 0.4 0.65 0.6
M3 0.12 0.45 0.27
M4 0.14 0.5 0.27 ;

```

Nell'ultimo file (`acciaio1.dat`) per assegnare valori al parametro a tre dimensioni (`tprod`) abbiamo usato la sintassi vista brevemente nella prima sezione.

Il valore della funzione obiettivo all'ottimo è in questo caso: 4978.463

Esercizio: *Supponiamo che per ragioni di mercato, siano assegnate delle quantità minime e massime per ciascun prodotto secondo la seguente tabella.*

| | <i>min</i> | <i>max</i> |
|------------------|------------|------------|
| <i>normale</i> | 200 | 300 |
| <i>inox</i> | — | 10 |
| <i>temperato</i> | — | — |

Come bisogna modificare i file del modello e/o dei dati per tener conto di queste specifiche? Il valore della funzione obiettivo all'ottimo si modifica? (Aiuto: la costante numerica ∞ si indica, in AMPL, con la parola chiave `Infinity`)

Un possibile modo è quello di aggiungere nel file del modello le seguenti istruzioni

```

param ulim{PROD};
param llim{PROD};
.
.
s.t. u_lim{p in PROD}: sum{t in TURNI, m in MACCHINE}x[m,t,p] <= ulim[p];
s.t. l_lim{p in PROD}: sum{t in TURNI, m in MACCHINE}x[m,t,p] >= llim[p];

```

e nel file dei dati

```

param:      llim :=
           normale 200
           inox    -Infinity
           temperato -Infinity ;

param:      ulim :=
           normale 300
           inox    10
           temperato +Infinity ;

```

Il valore della funzione obiettivo all'ottimo deve essere: 4903.035

Esercizio: *Supponiamo ora che per ragioni di manutenzione, la macchina M2 non possa funzionare nel turno 16-24. Come bisogna modificare i file del modello e dei dati per tenere in considerazione le nuove specifiche del problema? Il valore della funzione obiettivo all'ottimo si modifica?*

Le istruzioni da aggiungere sono: nel file acciaio1.mod

```

param mpart symbolic in MACCHINE;
param tpart symbolic in TURNI;
.
.
.
s.t. vinc{p in PROD}: x[mpart,tpart,p] = 0;

```

e nel file acciaio1.dat le definizioni relative ai due nuovi parametri `mpart` e `tpart`

```

param mpart := M2;
param tpart := '16-24';

```

Occorre sottolineare l'importanza delle definizioni dei parametri `mpart` e `tpart` nel file dei dati. Infatti, rappresentando degli elementi degli insiemi `MACCHINE` e `TURNI` rispettivamente, nell'assegnamento bisogna fare attenzione ad assegnare loro dei valori contenuti negli insiemi `MACCHINE` e rispettivamente `TURNI`, questo per non generare errori (di *semantica*).

Il valore della funzione obiettivo all'ottimo deve essere: 4664.317

6 Programmazione Intera (a cura di G. Fasano e G. Liuzzi)

Cominciamo questa sezione con la descrizione di un problema noto con il nome di *Segregated Storage Problem*. È necessario immagazzinare quattro merci (M1, M2, M3, M4) all'interno di sette silos (S1, S2, S3, S4, S5, S6, S7). In ogni silo può trovare posto solo un tipo di merce. Di seguito si riporta una tabella che riassume:

- le quantità (in ton.) di merce i -esima (q_i) da immagazzinare,
- il costo di immagazzinamento (c_{ij}), per ogni tonnellata di merce i -esima, nel silo j -esimo in milioni di lire,
- il costo fisso (f_j), in milioni di lire, associato all'utilizzo del silo j -esimo,
- la capacità (s_j), in tonnellate, del j -esimo silo;

| | S1 | S2 | S3 | S4 | S5 | S6 | S7 | m_i |
|-------------|----|----|----|----|----|-----|-----|-------|
| M1 | 1 | 2 | 2 | 3 | 4 | 5 | 5 | 75 |
| M2 | 2 | 3 | 3 | 3 | 1 | 5 | 5 | 50 |
| M3 | 4 | 4 | 3 | 2 | 1 | 5 | 5 | 25 |
| M4 | 1 | 1 | 2 | 2 | 3 | 5 | 5 | 80 |
| Costi fissi | 2 | 4 | 6 | 8 | 10 | 12 | 14 | |
| Capacità | 25 | 25 | 40 | 60 | 80 | 100 | 100 | |

Costruire un modello di programmazione lineare mista per la minimizzazione dei costi associati all'immagazzinamento delle merci.

Il nostro obiettivo è quello di trascrivere in AMPL il problema di immagazzinamento appena visto. Come prima cosa dobbiamo ovviamente individuare le *variabili di decisione* del problema. Una possibile scelta per le variabili di decisione è la seguente

Variabili di decisione:

$$\begin{aligned}
 x_{ij} &\equiv \text{tonnellate di merce } M_i \text{ immagazzinate nel silo } S_j \\
 y_{ij} &= \begin{cases} 1 & \text{se si immagazzina la merce } M_i \text{ nel silo } S_j \\ 0 & \text{altrimenti} \end{cases} \quad \left(\begin{array}{l} i = 1, \dots, 4 \\ j = 1, \dots, 7 \end{array} \right)
 \end{aligned}$$

Con questa scelta per le variabili di decisione possiamo scrivere la seguente funzione obiettivo (da minimizzare)

Funzione obiettivo:

$$\sum_{i=1}^4 \sum_{j=1}^7 c_{ij} x_{ij} + \sum_{j=1}^7 f_j \sum_{i=1}^4 y_{ij}$$

Vincoli del problema:

$$\sum_{i=1}^4 y_{ij} \leq 1 \quad \text{Vincoli sul numero di merci in ciascun silo (per ogni } j = 1, \dots, 7)$$

$$\sum_{j=1}^7 x_{ij} = q_i \quad \text{Vincoli sulle quantità da immagazzinare (per ogni } i = 1, \dots, 4)$$

$$\sum_{i=1}^4 x_{ij} \leq s_j \quad \text{Vincoli sulle capacità dei silos (per ogni } j = 1, \dots, 7)$$

$$x_{ij} \leq y_{ij} M \quad \text{Vincoli tecnici (per ogni } i = 1, \dots, 4 \text{ e } j = 1, \dots, 7, M \gg 1)$$

$$x_{ij} \geq 0 \quad \text{Vincoli di non negatività (per ogni } i = 1, \dots, 4 \text{ e } j = 1, \dots, 7)$$

$$y_{ij} \in \{0, 1\} \quad \text{Vincoli di interezza (per ogni } i = 1, \dots, 4 \text{ e } j = 1, \dots, 7)$$

Facciamo esplicitamente notare che il vincolo

$$x_{ij} \leq y_{ij} M \quad (\text{per ogni } i = 1, \dots, 4 \text{ e } j = 1, \dots, 7, M \gg 1)$$

serve ad esprimere la relazione logica

$$y_{ij} = 0 \implies x_{ij} = 0$$

ovvero, equivalentemente,

$$x_{ij} > 0 \implies y_{ij} = 1$$

che era stata implicitamente fatta contestualmente alla scelta delle variabili di decisione. Senza l'aggiunta di questo vincolo infatti le variabili x_{ij} ed y_{ij} sarebbero completamente scorrelate le une dalle altre.

Passiamo ora alla implementazione in AMPL del problema dell'immagazzinamento. Di seguito riportiamo il contenuto dei file del modello `storageStud.mod` e dei dati `storageStud.dat`

storageStud.mod

```

set MERCI;
set SILOS;

param costo_merce_silos{MERCI,SILOS} >= 0;
param costo_fisso {SILOS} >= 0;
param capacita {SILOS} >= 0;
param merce_totale {MERCI} >= 0;
param big_M := 10^5;

var x{MERCI,SILOS} >= 0;

var y{MERCI,SILOS} >= 0, binary;

minimize costi_complessivi:
    sum {i in MERCI, j in SILOS} costo_merce_silos[i,j]*x[i,j] +
    sum {j in SILOS} costo_fisso[j]*sum {i in MERCI} y[i,j];

#####
#s.t. magazzino {i in MERCI}:
#
#           sum {j in SILOS} x[i,j] = merce_totale[i];#
#####

s.t. merci_in_silos {j in SILOS}:
    sum {i in MERCI} y[i,j] <= 1;

#####
#s.t. vincolo_tecnico {i in MERCI, j in SILOS}:
#
#           x[i,j] <= big_M*y[i,j];
#
#####

```

storageStud.dat

```

set MERCI := M1 M2 M3 M4;
set SILOS := S1 S2 S3 S4 S5 S6 S7;

#####
#param costo_merce_silos:
#
#           S1    S2    S3    S4    S5    S6    S7:=
#
#           M1  1    2    2    3    4    5    5
#           M2  2    3    3    3    1    5    5
#           M3  4    4    3    2    1    5    5
#
#####

```

```

#      M4 1      1      2      2      3      5      5 ;      #
#####

param:      costo_fisso  capacita:=
  S1        2           25
  S2        4           25
  S3        6           40
  S4        8           60
  S5       10           80
  S6       12          100
  S7       14          100 ;

param      merce_totale:=
  M1        75
  M2        50
  M3        25
  M4        80 ;

```

Esercizio: Modificare il file del modello e/o quello dei dati per tener conto del fatto che è presente un ottavo silo (S8) con le seguenti caratteristiche:

```

      costo di immagazz. per ton. di
merce1      3
merce2      3
merce3      3
merce4      3

costo fisso = 3
capacita' = 100

```

Esercizio: Modificare il file del modello e/o quello dei dati per tener conto del fatto che se nel silo S1 si immagazzina la merce M1, allora in S2 non si può immagazzinare M4.

Esercizio: Modificare il file del modello e/o quello dei dati per tener conto della seguente ulteriore specifica: La quantità di merce immagazzinata nel silo S3 sia non inferiore alla quantità media di merce immagazzinata in tutti i restanti silos. (Aiuto: la funzione AMPL che restituisce la cardinalità di un insieme è `card(INS)`)

Ricordiamo che, in un file di modello `.mod`, per indicare che un parametro rappresenta un generico elemento di un insieme, si usa la notazione

```
param <nome_parametro> symbolic in <nome_insieme>
```

Ricordiamo inoltre che, dato un insieme GENINS, per indicare l'insieme degli elementi di GENINS aventi una certa proprietà, si usa la notazione

```
{i in GENINS: <proprietà'>}
```

Soluzione Esercizio 2:

Un possibile modo di risolvere questo esercizio, consiste nell'aggiungere al file del modello le seguenti righe di codice:

```
.  
.br/>var z binary;  
.br/>s.t. vincolo_ese2_1:x['M1','S1'] <= z*big_M;  
s.t. vincolo_ese2_2:x['M4','S2'] <= (1-z)*big_M;  
.br/>.
```

Infatti, quello che vogliamo è riuscire ad esprimere il seguente vincolo logico

$$x_{11} > 0 \implies x_{42} = 0$$

Per fare questo, introduciamo una nuova variabile binaria z tale che:

$$\begin{aligned} x_{11} > 0 &\implies z = 1 \\ 1 - z = 0 &\implies x_{42} = 0. \end{aligned}$$

Le precedenti due implicazioni logiche sono infine esprimibili mediante aggiunta al modello dei vincoli

$$\begin{aligned} x_{11} &\leq zM \\ x_{42} &\leq (1 - z)M \end{aligned}$$

Soluzione Esercizio 3: Un possibile modo di risolvere questo esercizio, consiste nell'aggiungere al file `storageStud.mod` le seguenti righe di codice:

```
.br/>.br/>param siloPart symbolic in SILOS;  
.br/>s.t. vincolo_ese3:sum{i in MERCI}x[i,siloPart] >=  
    (sum{j in SILOS: j != siloPart}sum{i in MERCI}x[i,j])/(card(SILOS)-1);  
.br/>.
```

ed al file `storageStud.dat` la seguente riga:

```
.br/>.br/>param siloPart := S3;  
.br/>.
```

Riferimenti bibliografici

- [1] R. Fourer, D.M. Gay, and B.W. Kernighan, *AMPL a modeling language for mathematical programming*, boyd & fraser publishing company, Massachusetts, 1993