Problemi su Reti

29 dicembre 2000

Finora sono stati formulati mediante il linguaggio di modellazione AMPL, problemi di programmazione lineare piuttosto generali. Esiste anche la possibilità di creare modelli per alcuni specici problemi. In questi casi, sono possibili entrambe le seguenti alternative:

(1) risolvere il problema come un qualsiasi problema di programmmazione lineare (PL);

(2) individuare nel problema una \struttura" particolare e risolverlo tenendo conto di questa ulteriore informazione.

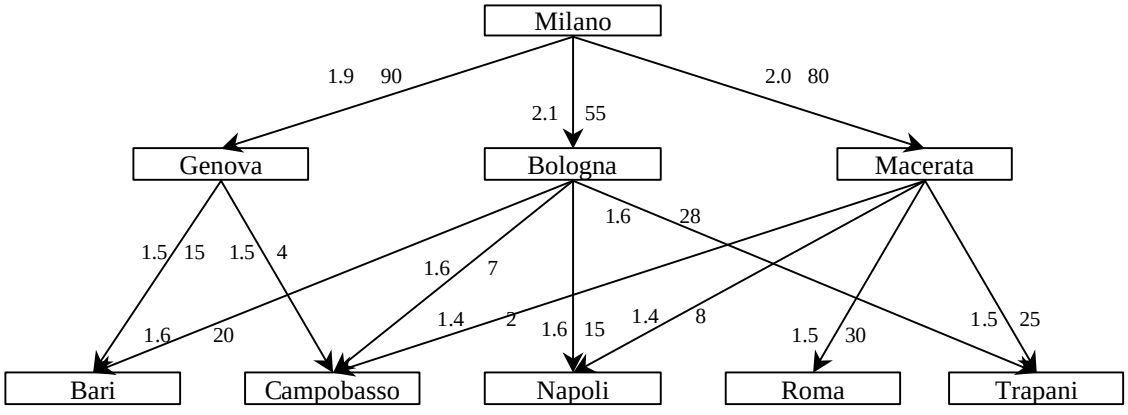
Vediamo un problema di trasporti come esempio di problema \su reti", che appartiene alla categoria (2).

1 Problema di trasporti

Si immagini per esempio (cfr. gura) di dover trasportare 80 tonnellate di una determinata merce, da uno stabilimento di produzione (Milano) ai negozi per la vendita (Bari, Campobasso, Napoli, Roma, Trapani), passando attraverso 3 magazzini di grossisti (Genova, Bologna, Macerata). La quantità di merce da trasportare da Milano ai negozi è così ripartita (in tonnellate):

	Bari	Campobasso	Napoli	Roma	Trapani
quantità	25	8	16	12	19

Non tutte le città sono collegate (cfr. gura), inoltre per ciascun \arco" (strada) che connette 2 città viene indicato: il costo unitario di trasporto ed il quantitativo massimo trasportabile.



Si chiede di minimizzare i costi totali di trasporto da Milano alle 5 citta ove sono ubicati i negozi. Una possibile formulazione (¯le .mod e .dat) potrebbe essere la seguente:

¯le network1.mod

```
set CITTA;              # comprende tutte le citta' nella figura.

set PERCORSI within {CITTA, CITTA};   # definiamo un sottoinsieme
                                      # delle coppie di citta'

param offerta_domanda {CITTA};   # la componente i-sima di questo
                                 # array di parametri e' positiva se
                                 # la citta' i-sima e' Milano, e'
                                 # nulla se la la citta' i-sima e'
                                 # Genova, Bologna o Macerata, e'
                                 # infine negativa per le rimanenti
                                 # componenti.

param costi {PERCORSI} >=0;    # indica il costo unitario associato
                               # a ciascuno dei possibili collega-
                               # menti tra citta'.

param merce_trasportabile {PERCORSI} >=0;   # indica il max quanti-
                                            # tativo di merce tra-
                                            # sportabile per ogni
                                            # collegamento.

var x {(i,j) in PERCORSI} >=0, <= merce_trasportabile[i,j];
                                  # e' il quantitativo di merce che
                                  # alla fine transita su ciascuno
                                  # dei collegamenti possibili.

minimize costi_complessivi: sum {(i,j) in PERCORSI} costi[i,j] * x[i,j];

s.t. equilibrio {i in CITTA}:
        sum {(k,i) in PERCORSI} x[k,i] + offerta_domanda[i] =
        sum {(i,j) in PERCORSI} x[i,j];
              # per ogni citta' vi e' un equilibrio tra i quantitativi
              # di merce "entrante" ed "uscente".
```

Si noti che i vincoli nel ¯le network1.mod rapppresentano relazioni di \equilibrio" per ciascuna citta , nel senso che la somma dei °ussi (frecce in ¯gura) di merci entranti e quelli uscenti devono equivalersi. Quindi la componente i-sima del vettore offerta_domanda sara

- positiva per citta da cui le merci escono esclusivamente (Milano);
- nulla per le citta con grossisti (citta di transito Genova, Bologna, Macerata);
- negativa per le rimanenti citta .

Il modello cosa come e stato sviluppato, ha lo svantaggio di risolvere il problema senza mostrare esplicitamente una di®erenziazione di ruoli tra le varie citta .

mentre per il file network.dat si ha l'espressione:

file network.dat

```
set CITTA := Milano Genova Bologna Macerata Bari Campobasso Napoli Roma Trapani;

set PERCORSI := Milano Genova  Milano Bologna  Milano Macerata
                Genova Bari  Genova Campobasso
                Bologna Bari  Bologna Campobasso Bologna Napoli
                Bologna Trapani
                Macerata Campobasso  Macerata Napoli  Macerata Roma
                Macerata Trapani  ;

param      offerta_domanda :=
  Milano          80
  Genova           0
  Bologna          0
  Macerata         0
  Bari            -2
  Campobasso       -8
  Napoli          -16
  Roma            -12
  Trapani         -19 ;

param:             costi       merce_trasportabile :=
    Milano Genova         1.9   90
    Milano Bologna        2.1   55
    Milano Macerata       2     80
    Genova Bari           1.5   15
    Genova Campobasso     1.5    4
    Bologna Bari          1.6   20
    Bologna Campobasso    1.6    7
    Bologna Napoli        1.6   15
    Bologna Trapani       1.6   28
    Macerata Campobasso   1.4    2
    Macerata Napoli       1.4    8
    Macerata Roma         1.5   30
    Macerata Trapani      1.5   25 ;
```

Viceversa è possibile evidenziare la struttura "grafica" del problema, identificando ciascuna città come **nodo** e ciascun collegamento tra coppie di città come **arco**. AMPL è in grado di mostrare tale struttura mediante l'introduzione esplicita dei costrutti **node** e **arc**, sostituendoli rispettivamente a quelli di **subject to** e **var**. In pratica a ciascun vincolo di equilibrio viene sostituita una dichiarazione **node** ed a ciascuna variabile si sostituisce un "arco". Quest'ultimo (cfr. figura) viene sostanzialmente identificato dai due nodi estremi da cui diparte ed arriva. Nel modello presentato allora, introducendo queste nuove dichiarazioni, il file .mod subisce una modifica mentre rimane inalterato il file .dat; il nuovo file .mod è dato dal seguente

file network2.mod

```
set CITTA;
set PERCORSI within {CITTA, CITTA};

param offerta_domanda {CITTA};
param costi {PERCORSI} >=0;
param merce_trasportabile {PERCORSI} >=0;

minimize costi_complessivi;

node NODO {i in CITTA}: net_out = offerta_domanda[i];

arc x {(i,j) in PERCORSI} >=0, <= merce_trasportabile[i,j],
      from NODO[i], to NODO[j], obj costi_complessivi costi[i,j];
```

che, relativamente alla sezione di dichiarazione insiemi e parametri, è identico al file network1.mod. Rimarchiamo inoltre il fatto che nel file network2.mod la dichiarazione delle variabili x è successiva tanto alla dichiarazione della funzione obiettivo quanto a quella dei vincoli; ciò si deve essenzialmente alla necessità di poter utilizzare in generale gli oggetti, solo dopo averli precedentemente definiti.
Per chiarire meglio le differenze con il file network1.mod, si osservi che nel file network2.mod, della funzione obiettivo è rimasto sostanzialmente il nome, identico a quello definito nel file network1.mod. La sua espressione è invece specificata due righe dopo. Nella istruzione successiva vengono dichiarati i nodi della rete; in essa possiamo distinguere le seguenti parti:

- inizia con la parola chiave node per indicare che si stanno introducendo nodi di una rete;
- viene inserito il nome del vettore di nodi NODO;
- si assegna alla parola chiave net_out la differenza tra la quantità di merce entrante e quella di merce uscente dal nodo: questo comunica ad AMPL se per ogni nodo questa differenza è nulla (nodo di transizione Genova, Bologna, Macerata), oppure è un nodo di origine o destinazione (Milano, Bari, Campobasso, Napoli, Roma, Trapani).

Infine nelle ultime due righe del file network2.mod vengono introdotte le variabili del problema, quindi queste istruzioni sostituiscono completamente la dichiarazione var nel file network1.mod; per esse possiamo dire quanto segue:

- cominciano con la parola chiave arc per indicare che la variabile x[i,j] è associata all'arco [i,j], essendo i,j due città dell'insieme CITTA;
- poi segue il vincolo bilatero >=0, <= merce_trasportabile[i,j] sulla variabile x[i,j] (tale vincolo bilatero è in generale opzionale);
- poi viene comunicato ad AMPL chi sono gli estremi dell'arco [i,j], introducendoli con le parole chiave from e to, separando le informazioni con una virgola;
- infine obj costi_complessivi costi[i,j] vuol dire che la funzione obiettivo costi_complessivi è fatta dalla somma dei prodotti degli archi [i,j] (non specificati ma presenti di default) per i costi[i,j]. L'obiettivo è quindi la somma di tutti i prodotti costi[i,j]*x[i,j], come nel caso precedente.

La variabile predefinita net_out serve a comunicare ad AMPL se nel nodo vi è o meno equilibrio; essa può essere equivalentemente sostituita con la variabile net_in, cambiando però di segno alla differenza successiva.