Esercizio 1: Rete logistica

Il problema richiede di determinare per ogni coppia di nodi il numero di *containers* da trasportare ogni giorno per riequilibrare la rete. Esistono quindi tante variabili quante le coppie di nodi; esse possono assumere solo valori non-negativi. Poiché tutti i dati sono interi, grazie alla particolare struttura combinatoria del problema (problema di trasporto) non è necessario imporre vincoli di integralità; le variabili si possono supporre continue. Anche dal punto di vista modellistico si può comunque supporre che eventuali valori frazionari sarebbero comunque accettabili, dal momento che si tratta di frequenze e non di numeri di *containers*.

I vincoli del problema impongono anzitutto il bilanciamento ad ogni nodo della rete: ogni giorno la differenza tra il numero di *containers* entranti ed uscenti deve essere uguale ed opposta in segno alla differenza tra *containers* in eccesso e quelli in difetto.

Inoltre è necessario imporre i vincoli sulla capacità delle tratte. Per facilitare l'analisi parametrica sulla capacità, si può introdurre una variabile ausiliaria, che rappresenta appunto la capacità, il cui valore viene poi fissato da un vincolo di uguaglianza su cui si esegue l'analisi parametrica.

La funzione obiettivo richiede la minimizzazione dei costi complessivi, che sono dati dalla somma di tutti i termini relativi ad ogni singola tratta (coppia di nodi). Per ogni tratta il costo è dato dal prodotto tra il numero di container da trasportare (variabile), la distanza tra i due nodi (data) ed il coefficiente dato (pari a 0.8). Per evitare di dover moltiplicare per 0.8 tutte le distanze una per una, dato che Lindo richiede di esplicitare tutti i coefficienti, si possono seguire due strade: la prima è quella di scrivere il modello in Lingo e poi di esportarlo in Lindo; la seconda è quella di introdurre una variabile ausiliaria z, porre z pari alla somma dei prodotti suddetti trascurando il coefficiente di proporzionalità e poi ottimizzare una funzione obiettivo pari a $0.8\ z$. Nei *files* di soluzione è stata seguita la prima delle due strade.

Si ottiene quindi un modello di programmazione lineare. Il modello Lingo è nel file RETELOG.LG4. Il modello Lindo equivalente è nel file RETELOG.LTX. La soluzione ottima è nel file RETELOG1.SOL.

La soluzione ottima è unica, poiché non esistono variabili fuori base con costo ridotto nullo.

Volendo aumentare la capacità di una tratta è ovviamente conveniente scegliere quella per cui la diminuzione del costo della rete logistica per ogni unità di capacità aggiunta risulta massimo. L'informazione si desume immediatamente dai prezzi-ombra delle risorse corrispondenti ai vincoli di capacità. Quasi tutti i vincoli hanno uno *slack* di risorsa (e quindi prezzo-ombra nullo) tranne due, i cui prezzi-ombra sono pari a 128 Euro/container e a 32 Euro/container. La tratta su cui è più efficiente intervenire, è quindi la tratta 1-3. Essa è anche l'unica su cui l'investimento in capacità aggiuntiva ha senso, poiché nell'altro caso il prezzo-ombra (32 Euro/container) è inferiore al costo delle unità di capacità (100 Euro/container) e quindi l'investimento non è giustificato.

Dall'analisi parametrica sul vincolo di capacità della tratta 1-3 si desume che il prezzo-ombra è pari a 128 Euro/container per le prime 5 unità di capacità aggiunte (informazione ricavabile anche dall'analisi di sensitività sui termini noti) e di 112 Euro/container per le successive 5 unità. Dopo tale aumento il prezzo-ombra si azzera e ulteriori aumenti di capacità non portano ad alcuna variazione nel valore ottimo del problema. La scelta migliore è quindi quella di aumentare di 10 container/giorno la capacità della tratta, ottenendo una diminuzione del costo complessivo da 56160 Euro/giorno a 54960 Euro/giorno cui vanno aggiunti i costi per aumentare la capacità, pari a 1000 Euro/giorno. Pertanto il risparmio complessivo è di 200 Euro/giorno.

Per rispondere all'ultima domanda basta eseguire l'analisi parametrica sul vincolo che impone il valore della capacità. Il risultato è riportato nel file RETELOG2.SOL. Il problema resta ammissibile fino ad un valore di capacità pari a 12.8 container/giorno cui corrisponde un costo di 158762 Euro/giorno.

Esercizio 2: Sfere

Senza perdita di generalità è comodo assumere che la sfera che contiene le altre sia centrata nell'origine. Le variabili del problema sono quindi le posizioni dei centri delle sfere di raggio unitario ed il raggio della sfera che le deve contenere. Si hanno quindi $3n$ variabili continue e libere ed una variabile continua non-negativa (il raggio della sfera grande).

La funzione obiettivo richiede semplicemente di minimizzare il valore di quest'ultima variabile.

I vincoli del problema impongono la non-sovrapposizione tra le sfere piccole ed il contenimento di ciascuna nella sfera grande. Nel primo caso si impone che la distanza Euclidea tra i centri di ogni coppia di sfere sia maggiore o uguale al doppio del loro raggio; nel secondo caso di impone che la distanza del centro di ogni sfera piccola dall'origine (centro della sfera grande) sia minore o uguale alla differenza tra il raggio della sfera grande ed il raggio della sfera piccola.

Il risultante modello di programmazione non-lineare è contenuto nel file Lingo SFERE.LG4 e la soluzione (minimo locale) è contenuta nel file SFERE.LGR.

Il problema non è convesso, quindi non si ha garanzia sull'ottimalità globale della soluzione trovata.

Esercizio 3: Scheduling di macchine parallele

Il problema chiede di determinare l'assegnamento dei *jobs* alle macchine (non richiede di determinare in quale ordine i *jobs* vengono eseguiti su ogni macchina). Quindi è necessario introdurre tante variabili binarie di assegnamento quante le coppie (job,macchina). La funzione obiettivo è di tipo minimax, poiché si chiede di minimizzare il massimo tempo di utilizzo delle varie macchine. Si introduce quindi una variabile ausiliaria di cui si richiede la minimizzazione e si impone che tale variabile assuma un valore maggior o uguale a ciascuno dei tempi di lavorazione complessivi su ogni macchina. I tempi di lavorazione complessivi su ogni macchina sono dati dal prodotto scalare del vettore dei *processing times* e del vettore delle variabili di assegnamento relative alla macchina stessa.

I vincoli di assegnamento sono uno per ogni *job* ed impongono che il numero di macchine cui il *job* viene assegnato sia pari ad 1.

I vincoli di incompatibilità si esprimono come "somma di due variabili binarie minore o uguale ad 1". Le due variabili binarie si riferiscono agli assegnamenti dei due *jobs* considerati alla stessa macchina. Per rendere il vincolo generale, in modo da poterlo scrivere per tutte le macchine e le coppie di *jobs*, si può esprimere il termine noto come $2 - t$, dove t è un dato che vale 1 se i *jobs* sono incompatibili e 0 se sono compatibili (come nel file dei dati).

I vincoli sul massimo e sul minimo numero di *jobs* si esprimono facilmente come limiti inferiori e superiori alla soma delle variabili di assegnamento relative alla stessa macchina, per ogni macchina. Ovviamente il numero massimo di *jobs* per macchina è pari al numero massimo di *set-up* consentiti più uno, dato che esiste un'operazione di *set-up* per ogni coppia di *jobs* consecutivi.

Il modello risultante è di programmazione lineare 0-1. Il modello Lingo è nel fle PARSCHED.LG4 e la soluzione ottima è nel file PARSCHED.LGR. E' garantito che la soluzione è ottima, non è garantito che sia unica.