Ricerca Operativa

Problemi di ottimizzazione su reti di flusso e cammini minimi

L. De Giovanni

AVVERTENZA: le note presentate di seguito non hanno alcuna pretesa di completezza, né hanno lo scopo di sostituirsi alle spiegazioni del docente. Il loro scopo è quello di fissare alcuni concetti presentati in classe. Le note contengono un numero limitato di esempi ed esercizi svolti. Questi rappresentano una parte fondamentale nella comprensione della materia e sono presentati in classe.

Contents

1 Problemi di flusso di costo minimo

Molti problemi di ottimizzazione combinatoria possono essere modellati ricorrendo ai grafi. Consideriamo il seguente esempio.

Esempio 1 (Un problema di distribuzione di energia) *Una società di produzione di energia elettrica dispone di diverse centrali di produzione e distribuzione, collegate tra loro con cavi reofori. Ogni centrale i può:*

- *produrre p_i kW di energia elettrica ($p_i = 0$ se la centrale non produce energia);*
- *distribuire energia elettrica su una sottorete di utenti la cui domanda complessiva è di d_i kW ($d_i = 0$ se la centrale non serve utenti);*
- *smistare l'energia da e verso altre centrali.*

I cavi che collegano una centrale i ad una centrale j hanno una capacità massima di u_{ij} kW e costano c_{ij} euro per ogni kW trasportato. La società vuole determinare il piano di distribuzione dell'energia elettrica di costo minimo, sotto l'ipotesi che l'energia complessivamente prodotta sia pari alla domanda totale delle sottoreti di utenti.

Per la modellazione matematica del problema consideriamo un grafo orientato costruito come segue. Sia $G = (N, A)$ un grafo i cui nodi corrispondono alle centrali e i cui archi corrispondono ai collegamenti tra le centrali. Per ogni nodo $v \in N$ definiamo il parametro $b_v = d_v - p_v$ che rappresenta la differenza tra la domanda che la centrale deve soddisfare e l'offerta di energia che è in grado di generare. Si noti che:

- $b_v > 0$ se la centrale deve soddisfare una domanda superiore alla capacità produttiva (la centrale deve far arrivare energia da altre centrali);
- $b_v < 0$ se nella centrale c'è un eccesso di offerta (la produzione in eccesso deve essere convogliata verso altre centrali);
- $b_v = 0$ se la domanda e l'offerta di energia si equivalgono o (come caso particolare) se la centrale svolge semplici funzioni di smistamento ($p_v = d_v = 0$).

In generale possiamo dire che b_v è la *richiesta* del nodo $v \in N$ (una richiesta negativa rappresenta un'offerta messa a disposizione della rete) e distinguere:

- nodi domanda (richiesta $b_i > 0$);
- nodi offerta (richiesta $b_i < 0$);
- nodi di transito (richiesta $b_i = 0$).

Le variabili decisionali sono definite sugli archi del grafo e sono relative alla quantità x_{ij} da far fluire sull'arco $(i,j) \in A$. Il modello è il seguente:

$$\begin{aligned} \min \quad & \sum_{(i,j)\in A} c_{ij}x_{ij} \\ s.t. \quad & \sum_{(i,v)\in A} x_{iv} - \sum_{(v,j)\in A} x_{vj} = b_v \quad \forall v \in N \\ & x_{ij} \leq u_{ij} \quad \forall (i,j) \in A \\ & x_{ij} \in \mathbb{R}_+ \end{aligned}$$

La funzione obiettivo minimizza il costo complessivo di distribuzione mentre i vincoli definiscono le x_{ij} come *flusso ammissibile* sulla rete:

- il primo insieme di vincoli, uno per ogni nodo, è detto *vincolo di bilanciamento dei nodi* e assicura che la differenza tra tutto il flusso (di energia elettrica) entrante in un nodo v (archi (i,v)) e tutto il flusso uscente dal nodo stesso (archi (v,j)) sia esattamente pari alla richiesta del nodo stesso (positiva per i nodi domanda, negativa per i nodi offerta e nulla per i nodi di transito);
- il secondo insieme di vincoli, uno per ogni arco, è il vincolo di *capacità degli archi* e limita la quantità che può fluire in ogni arco.

A questo punto disponiamo di un modello di programmazione lineare per il problema e pertanto, per la sua soluzione, possiamo utilizzare, ad esempio, il metodo del simplesso.

Il modello presentato per questo problema di distribuzione di energia elettrica può essere facilmente generalizzato a diversi altri problemi di distribuzione su reti: reti di distribuzione di beni materiali (il flusso è relativo al trasporto dei beni e i nodi rappresentano punti di produzione, consumo e smistamento), reti di telecomunicazioni (il flusso rappresenta la banda da riservare su ogni collegamento), reti di trasporto (il flusso rappresenta i veicoli sulla rete) etc. In generale, il problema prende il nome di *problema del flusso su reti di costo minimo* e può essere utilizzato per modellare e risolvere svariati problemi di ottimizzazione combinatoria (anche non direttamente modellabili su una rete di flusso).

2 Flusso di costo minimo e matrici totalmente unimodulari

Consideriamo ora l'esempio in Figura 1 (i numeri sugli archi rappresentano un flusso ammissibile).

Se trascuriamo i vincoli di capacità degli archi, il modello di flusso di costo minimo presenta i seguenti vincoli, scritti per esteso:

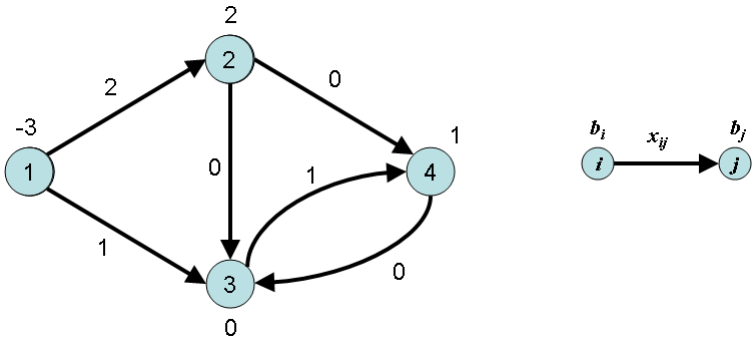

Figure 1: Esempio di flusso su rete.

$$
\begin{array}{rrrrrrrrrrrrll}
- & x_{12} & - & x_{13} & & & & & & & & = & -3 & \text{bilanciamento del nodo 1} \\
+ & x_{12} & & & - & x_{23} & - & x_{24} & & & & = & 2 & \text{bilanciamento del nodo 2} \\
 & & + & x_{13} & + & x_{23} & & & - & x_{34} & + & x_{43} & = & 0 & \text{bilanciamento del nodo 3} \\
 & & & & & & + & x_{24} & + & x_{34} & - & x_{43} & = & 1 & \text{bilanciamento del nodo 4}
\end{array}
$$

e la corrispondente matrice E dei vincoli è la seguente

x_{12}	x_{13}	x_{23}	x_{24}	x_{34}	x_{43}
-1	-1	0	0	0	0
$+1$	0	-1	-1	0	0
0	$+1$	$+1$	0	-1	$+1$
0	0	0	$+1$	$+1$	-1

Abbiamo una colonna per ogni variabile e, quindi, per ogni arco $(i, j) \in A$ e una riga per ogni nodo $v \in N$. Per ogni colonna, ci sono esattamente due numeri diversi da 0 e, in particolare un $+1$ e un -1. In effetti, il flusso x_{ij} su ogni arco (i, j) contribuisce al flusso di soli due nodi: viene aggiunto al bilanciamento del flusso del nodo in cui entra ($+1$ nella riga j) e viene sottratto dal bilanciamento del flusso del nodo da cui esce (-1 sul nodo i).

> Quindi, in generale, la matrice corrispondente ai vincoli di bilanciamento di flusso ha, per ogni colonna, un $+1$, un -1 e tutti gli altri elementi a 0.

Essendo la somma delle righe di E pari al vettore nullo, il rango della matrice E non è massimo. Si può in effetti dimostrare che:

$$\rho(E) = |N| - 1$$

Di conseguenza, affinché il sistema di equazioni $Ex = b$ relativo al bilanciamento di flusso abbia soluzione, è necessario che $\rho(E) = \rho(E|b) = |N| - 1$ e cioè che anche la somma di tutti i b_i sia 0:

$$\sum_{i \in N} b_i = 0$$

Tale condizione corrisponde a una *rete di flusso bilanciata*. Sotto tale condizione possiamo eliminare uno qualsiasi dei vincoli del sistema $Ex = b$, in quanto ridondante.

Se proviamo a calcolare il determinante di sottomatrici quadrate (di dimensioni 1×1, 2×2, 3×3 e 4×4), otteniamo come risultati sempre $+1$, -1 oppure 0. Il risultato è generalizzabile, grazie alla semplicità della matrice dei vincoli di bilanciamento. Si può infatti dimostrare la seguente proprietà.

Proprietà 1 *Sia E la matrice corrispondente ai vincoli di bilanciamento di flusso e D una qualsiasi sottomatrice quadrata (di qualsiasi dimensione). Allora* $\det(D) \in \{-1, 0, +1\}$.

Matrici con tale proprietà si dicono *matrici totalmente unimodulari (TUM)*. Quindi, la matrice relativa ai vincoli di bilanciamento di flusso è TUM.

Le matrici TUM hanno particolare rilevanza nell'ambito della programmazione lineare. Ricordiamo che una base B della matrice E è una sottomatrice quadrata di E di rango massimo ($|N|-1$). Comunque sia scelta una base B della matrice E, avremo che $det(B) \in \{-1, +1\}$, essendo lo 0 escluso dal fatto che la matrice B ha rango massimo (ed è quindi non singolare e invertibile). Si ricorda che

$$B^{-1} = \frac{1}{det(B)} B^{*T}$$

dove B^* è la matrice dei cofattori o complementi algebrici di B:

$$B^* = \begin{pmatrix} \text{cof}(B, B_{1,1}) & \dots & \text{cof}(B, B_{1,j}) \\ \vdots & \ddots & \vdots \\ \text{cof}(B, B_{i,1}) & \dots & \text{cof}(B, B_{i,j}) \end{pmatrix}^T$$

dove $B_{i,j}$ è l'elemento di B in posizione i, j. Il cofattore in posizione i, j è definito come:

$$\text{cof}(B, B_{i,j}) = (-1)^{i+j} \cdot \det(\text{minor}(B, i, j))$$

dove $minor(B, i, j)$ rappresenta il minore di B che si ottiene cancellando la riga $i-$esima e la colonna $j-$esima. Ora, E è TUM, e quindi:

- $B_{i,j} \in \{-1, 0, +1\}$, visto che ogni elemento è una particolare sottomatrice quadrata di E di ordine 1 il cui determinante (e quindi l'elemento stesso) è -1, 0 o $+1$;
- $\det(\text{minor}(B, i, j)) \in \{-1, 0, +1\}$ visto che $\text{minor}(B, i, j)$ è una sottomatrice di E.

Di conseguenza, ogni elemento di B^{-1} è -1, 0 o $+1$. Consideriamo ora una soluzione di base del sistema $Ex = b$, che rappresenta i vincoli di bilanciamento di flusso. Questa sarà ottenuta come:

$$x = \begin{bmatrix} x_B \\ x_E \end{bmatrix} = \begin{bmatrix} B^{-1}\tilde{b} \\ 0 \end{bmatrix}$$

dove B è una sottomatrice quadrata non singolare di E di dimensione $|N| - 1$[1] e $\tilde{b}$ è un sottovettore di b corrispondente all'eliminazione di un vincolo ridondante. In particolare, se $b \in \mathbb{Z}_+$, tutte le soluzioni di base avranno coordinate intere, essendo le variabili in base ottenute dal prodotto di una matrice intera (in particolare di -1, 0 o $+1$) per un vettore intero, e le variabili fuori base pari a 0 (numero intero).

Il risultato è generalizzabile ad un qualsiasi problema di programmazione lineare.

Proprietà 2 *Sia dato un problema di programmazione lineare in forma standard* $\min\{c^T x : Ax = b, x \geq 0\}$. *Se A è TUM e $b \in Z_+^m$, allora tutte le soluzione di base hanno coordinate intere.*

Un'importante conseguenza è che, se risolviamo il problema con l'algoritmo del simplesso, *che si applica solo quando le variabili sono definite reali*, ma che esplora solo soluzioni di base, otterremo una soluzione ottima di base *a coordinate intere*.

> Sia dato il problema di programmazione lineare ***intera*** $\min\{c^T x : Ax = b, x \in \mathbb{Z}_+^n\}$. Se A è TUM, il problema può essere risolto con l'algoritmo del simplesso.

Tornando al problema del flusso di costo minimo, possiamo anche dimostrare che la proprietà di totale unimodularità della matrice dei vincoli è conservata, anche se si aggiungono i vincoli di capacità degli archi. Supponiamo adesso che i beni che circolano sulla rete di flusso non siano divisibili (frigoriferi, passeggeri etc.). Il modello del flusso di costo minimo sarebbe:

$$\begin{aligned}
\min \quad & \sum_{(i,j)\in A} c_{ij} x_{ij} \\
s.t. \quad & \sum_{(i,v)\in A} x_{iv} - \sum_{(v,j)\in A} x_{vj} = b_v \quad \forall v \in N \\
& x_{ij} \leq u_{ij} \quad \forall (i,j) \in A \\
& x_{ij} \in \mathbb{Z}_+
\end{aligned}$$

in cui si utilizzano variabili intere. Ma, per le osservazioni sopra riportate, e sotto l'ipotesi (ragionevole) che i fattori di bilanciamento b_i siano interi e l'ipotesi di utilizzare l'algoritmo del simplesso per la soluzione del problema, possiamo in modo equivalente utilizzare lo stesso modello con variabili reali.

[1] Per ottenere una base B bisogna sempre escludere una riga di E, corrispondente ad un vincolo ridondante sotto l'ipotesi di bilanciamento della rete di flusso

3 Il problema del cammino minimo

Uno dei problemi classici sui grafi è il problema del cammino minimo, così definito. Sia dato un grafo $G = (N, A)$ con un costo c_{ij}, $\forall (i,j) \in A$, un nodo origine $s \in N$ e un nodo destinazione $d \in N$. Si vuole trovare il cammino P da s a d il cui costo (somma dei $c_{ij} : (i,j) \in P$) sia minimo. Introduciamo delle variabili decisionali binarie in corrispondenza degli archi del grafo:

$$x_{ij} = \begin{cases} 1, & \text{l'arco (i,j) è sul cammino minimo;} \\ 0, & \text{altrimenti.} \end{cases}$$

Un possibile modello è il seguente:

$$\begin{aligned} \min \quad & \sum_{(i,j)\in A} c_{ij} x_{ij} \\ \text{s.t.} \quad & \\ & \sum_{(i,v)\in A} x_{iv} - \sum_{(v,j)\in A} x_{vj} = \begin{cases} -1, & v = s; \\ +1, & v = d; \\ 0, & v \in N \setminus \{s,d\}. \end{cases} \\ & x_{ij} \in \{0,1\} \quad \forall\, (i,j) \in A \end{aligned}$$

Le variabili x_{ij} selezionano un insieme di archi a costo minimo. I vincoli stabiliscono che si può selezionare un solo arco uscente dall'origine; si può selezionare un solo arco entrante nella destinazione; per gli altri nodi, il numero di archi entranti selezionati deve essere uguale al numero di archi uscenti. Di fatto, le x_{ij} rappresentano un flusso unitario da s a d e il problema è stato formalizzato come un caso particolare del problema di flusso di costo minimo: si vuole far partire un'unità di flusso dall'origine s ($b_s = -1$) e farla arrivare alla destinazione d ($b_s = +1$) facendole compiere un percorso di costo minimo attraverso gli altri nodi della rete ($b_i = 0,\ \forall i \in N \setminus \{s,d\}$). Osserviamo che, anche eliminando il vincolo di dominio $x_{ij} \in \{0,1\}$, una soluzione ottima non avrà mai flussi superiori a 1, altrimenti si pagherebbe di più in funzione obiettivo. Pertanto, si potrebbe scrivere $x_{ij} \in \mathbb{Z}_+$. Inoltre, per le osservazioni sulla totale unimodularità della matrice dei vincoli, possiamo risolvere il problema del cammino minimo risolvendo il seguente modello di programmazione lineare:

$$\begin{aligned} \min \quad & \sum_{(i,j)\in A} c_{ij} x_{ij} \\ \text{s.t.} \quad & \\ & \sum_{(i,v)\in A} x_{iv} - \sum_{(v,j)\in A} x_{vj} = \begin{cases} -1, & v = s; \\ +1, & v = d; \\ 0, & v \in N \setminus \{s,d\}. \end{cases} \\ & x_{ij} \in \mathbb{R}_+ \quad \forall\, (i,j) \in A \end{aligned}$$

Visto che le variabili sono dichiarate come variabili reali e positive, possiamo applicare l'algoritmo del simplesso per risolvere il problema del cammino minimo.

4 Algoritmo label correcting per il problema del cammino minimo

La possibilità di usare un modello di programmazione lineare (non intera) per il problema del cammino minimo non ha solo la conseguenza di poter applicare l'algoritmo del simplesso, ma ha conseguenze più importanti. A questo punto, infatti, è possibile applicare al problema del cammino minimo la teoria della dualità in programmazione lineare (che non sarebbe applicabile se le variabili fossero intere) e derivare delle proprietà che possono essere sfruttate per la messa a punto di algoritmi di soluzione più efficienti. Il simplesso, infatti, ha complessità "esponenziale", mentre, vedremo, il problema del cammino minimo può essere risolto in tempo polinomiale.

Per applicare la teoria della dualità, scriviamo innanzitutto il duale del problema del cammino di costo minimo. Introduciamo una variabile π_i per ogni vincolo del primale e, quindi, per ogni nodo $i \in N$. Il problema duale avrà quindi la seguente funzione obiettivo:

$$\max \pi_d - \pi_s$$

e un vincolo per ogni variabile x_{ij}, cioè per ogni arco $(i, j) \in A$. La scrittura del vincolo duale relativo alla variabile x_{ij} considera la colonna della variabile stessa che, come abbiamo visto, presenta solo due elementi diversi da 0: $+1$ in corrispondenza del vincolo del nodo j (variabile duale π_j) e -1 in corrispondenza del vincolo del nodo i (variabile duale π_i). Il vincolo duale sarà quindi del tipo:

$$\pi_j - \pi_i \leq c_{ij}$$

Complessivamente, il duale del problema del cammino minimo è il seguente:

$$\begin{aligned} \max \quad & \pi_d - \pi_s \\ \text{s.t.} \quad & \pi_j - \pi_i \leq c_{ij} \quad \forall\, (i,j) \in A \\ & \pi_v \in \mathbb{R} \quad \forall\, v \in N \end{aligned}$$

Notiamo che i vincoli del problema duale sono particolarmente semplici. Si potrebbe quindi pensare di implementare un algoritmo che stabilisca dei valori ammissibili per le variabili π. Il valore π_i è anche detto *etichetta* (o *label*) del nodo i. Se, in corrispondenza di etichette ammissibili, riusciamo a costruire una soluzione del problema primale (variabili x) che sia ammissibile primale e in scarti complementari con π, allora avremmo ottenuto

una soluzione ottima per il problema del cammino minimo[2]. Un algoritmo molto semplice che permette di ottenere delle etichette ammissibili è il seguente, detto *label correcting* perché procede *correggendo* iterativamente le etichette che non soddisfano il vincolo duale:

Algoritmo *label correcting* generico

π_s := 0; set $p(s) = \wedge$;
for each $v \in N - s$ { set $\pi_v(v)$:=$+\infty$, set $p(v) = \wedge$; }
while ($\exists\,(i,j) \in A : \pi_j > \pi_i + c_{ij}$) do {
    set π_j := $\pi_i + c_{ij}$
    set $p(j)$:= i;
}

I primi passi sono di inizializzazione. Si fa notare che se aggiungiamo una stessa costante (sia positiva che negativa) a *tutte* le etichette, non cambia niente in termini di valore della funzione obiettivo duale e di soddisfazione dei vincoli duali (la stessa costante viene sempre prima sommata e poi sottratta). La soluzione duale è invariante rispetto all'aggiunta di una costante e quindi, possiamo fissare una variabile duale ad un valore arbitrario. Scegliamo di fissare $\pi_s = 0$. Si noti che tale osservazione corrisponde all'esistenza di un vincolo primale ridondante, e che fissare $\pi_s = 0$ corrisponde ad eliminare il vincolo del bilanciamento del nodo origine (la variabile π_s non compare più nel duale). Per gli altri nodi, scegliamo un valore iniziale molto elevato, con la convenzione che

$$+\infty \pm cost. = +\infty$$

Per il resto, l'algoritmo è un semplice ciclo che, ad ogni iterazione, controlla se un *qualsiasi* vincolo duale non è rispettato e, nella stessa iterazione, fa in modo che lo stesso vincolo sia rispettato all'uguaglianza, aggiornando opportunamente l'etichetta della testa dell'arco corrispondente. L'algoritmo, inoltre, mantiene un vettore di puntatori p, uno per ogni nodo. Ogni volta che l'etichetta di un nodo j viene aggiornata, il puntatore punta al nodo i che è coda dell'arco che ha causato l'aggiornamento. Prima di dimostrare la correttezza dell'algoritmo, vediamo un esempio di applicazione.

Esempio 2 *Si applichi l'algoritmo* label correcting generico *al grafo in Figura 2, per calcolare un insieme di etichette ammissibili duali con nodo origine $s = 1$ e nodo destintazione $d = 6$.*

Inizializzazioni: $\pi_1 = 0$; $\pi_2 = \pi_3 = \pi_4 = \pi_5 = \pi_5 = +\infty$;
Iterazione 1: arco $(1, 2)$: $\pi_2 = 7$; $p(2) = 1$.
Iterazione 2: arco $(2, 4)$: $\pi_4 = 11$; $p(4) = 2$.

[2]In realtà, questa tecnica è utilizzata molto spesso, per derivare degli algoritmi risolutivi di problemi di ottimizzazione combinatoria che siano modellabili come problemi di programmazione lineare.

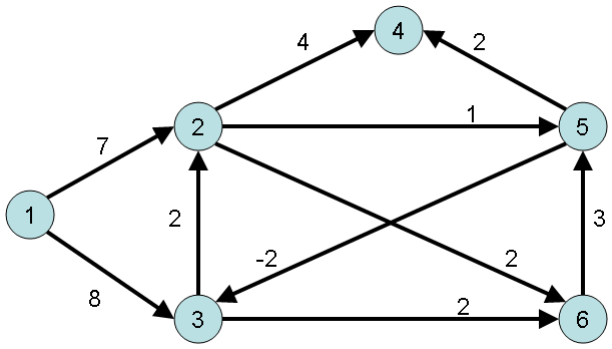

Figure 2: Rete di flusso dell'Esempio 2.

Iterazione 3: arco $(1,3)$: $\pi_3 = 8$; $p(3) = 1$.
Iterazione 4: arco $(3,6)$: $\pi_6 = 10$; $p(6) = 3$.
Iterazione 5: arco $(6,5)$: $\pi_5 = 13$; $p(5) = 6$.
Iterazione 6: arco $(2,5)$: $\pi_5 = 8$; $p(5) = 2$.
Iterazione 7: arco $(5,3)$: $\pi_3 = 6$; $p(3) = 5$.
Iterazione 8: arco $(5,4)$: $\pi_4 = 10$; $p(4) = 5$.
Iterazione 9: arco $(3,6)$: $\pi_6 = 8$; $p(6) = 3$.

A questo punto, tutti gli archi soddisfano i vincoli duali e l'algoritmo termima.

Si fa notare che, seguendo a ritroso i puntatori memorizzati nelle variabili $p(i)$, a partire dal nodo d, si ottiene un cammino da s a d:

$$6 \leftarrow 3 \leftarrow 5 \leftarrow 2 \leftarrow 1$$

e che il costo di questo cammino è proprio π_6: $7 + 1 - 2 + 2 = 8$.

4.1 Proprietà dell'algoritmo

La dimostrazione di correttezza dell'algoritmo usa le condizioni di complementarietà primale duale. Per la loro applicazione, dobbiamo dimostrare il seguente Lemma.

Lemma 1 *Ad ogni iterazione, considerati i valori correnti delle etichette π, per ogni nodo $j \in N : \pi_j < +\infty$ si ha:*

- *esiste un cammino P dal nodo origine s a j;*
- *$p(j)$ è il predecessore di j in tale cammino P;*
- *il costo del cammino P è pari a π_j.*

Dimostrazione: Dimostriamo l'asserto per induzione[3]. Dimostriamo che la proposizione è vera all'iterazione $k = 1$. Alla fine dell'iterazione 1, esistono al più due nodi $j : \pi_j < +\infty$. Il primo nodo è l'origine s. Per tale nodo è immediato verificare che: esiste un cammino da s (formato dal solo nodo s); il predecessore di s in questo cammino è NULL, come stabilito da $p(s)$; il costo di questo cammino è 0, come riportato da π_s. Sia $w \neq s$ il secondo nodo tale che $\pi_w < +\infty$. Se $\pi_w < +\infty, w \neq s$ alla fine della prima iterazione, allora π_w è stato aggiornato, cioè $\exists\ i : \pi_w > \pi_i + c_{iw}$ e, di conseguenza, $\pi_i < +\infty$. Ma all'inizio della prima iterazione solo $\pi_s < +\infty$, cioè $i = s$. Di conseguenza, abbiamo posto $\pi_w = c_{sw}$ e $p(w) = s$. Anche per w è quindi immediato che: esiste un cammino $s \to w$ da s a w; il predecessore di w in questo cammino è $s = p(w)$; il costo del cammino è $c_{sw} = \pi_w$. È quindi vero che, all'iterazione 1, le tre condizioni sono vere per ogni nodo $j : \pi_j < +\infty$.

Assumiamo ora (ipotesi induttiva) che la proposizione sia vera all'iterazione k, e cioè che, per ogni $j \in N : \pi_j < +\infty$, si abbia un cammino $P = s \to v_1 \to v_2 \to \cdots \to p(v) \to v$ di lunghezza pari a π_v (vedi Figura 3).

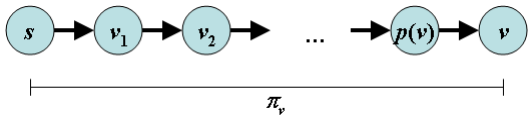

Figure 3: Ipotesi induttiva.

All'iterazione $k + 1$ si aggiorna un $\pi_j : \pi_j > \pi_i + c_{ij}$. Se tale condizione è rispettata, deve essere $\pi_i < +\infty$ e, per ipotesi induttiva, esiste un cammino $P = s \rightsquigarrow p(i) \to i$ di lunghezza pari a π_i. Dopo aver aggiornato $\pi_j = \pi_i + c_{ij}$ e $p(j) = i$, possiamo dire che (vedi Figura 4):

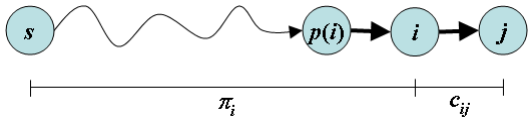

Figure 4: Passo induttivo.

esiste un cammino $P' = P \to j = s \rightsquigarrow p(i) \to i \to j$ da s aj; il predecessore di j su questo cammino è $i = p(j)$; il costo del cammino è $\pi_i + c_{ij} = \pi_j$. Le tre condizione valgono quindi per il nodo j la cui etichetta è stata aggiornata al passo $k + 1$ e continua a valere per gli

[3]Il principio di induzione afferma che, data una proposizione Π definita su un numero naturale k allora:

$$\begin{cases} \Pi(1) = true \\ \Pi(k) = true \Rightarrow \Pi(k+1) = true \end{cases} \quad \Rightarrow \quad \Pi(n) = true,\ \forall\ n \geq 1$$

$\Pi(k) = true$ è detta *ipotesi induttiva*.

altri nodi con etichetta finita, visto che tali etichette e i relativi puntatori non sono stati modificati. Per induzione, quindi, la proprietà vale per tutte le iterazioni $n \geq 1$. ■

4.2 Convergenza e complessità

Per il Lemma 1, se $\pi_j < +\infty$, π_j è la lunghezza di un cammino ammissibile da s a j. In ogni caso, π_j rappresenta sempre un limite superiore (upper bound) alla lunghezza del cammino minimo da s a j. Inoltre, se l'algoritmo itera, allora un'etichetta viene diminuita (strettamente). Consideriamo adesso due casi, relativi all'esistenza o meno di cicli di costo negativo nel grafo.

Se esiste un ciclo negativo, per i nodi j nel ciclo sarà sempre possibile diminuire le relative etichette, come risulta evidente dall'esempio in Figura 5.

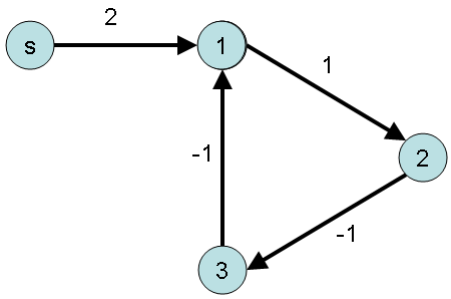

Figure 5: Esempio di flusso su rete.

Inizializzazioni: $\pi_s = 0$; $\pi_1 = \pi_2 = \pi_3 = +\infty$;
Iterazione 1: arco $(s, 1)$: $\pi_1 = 2$; $p(1) = s$.
Iterazione 2: arco $(1, 2)$: $\pi_2 = 3$; $p(2) = 1$.
Iterazione 3: arco $(2, 3)$: $\pi_3 = 2$; $p(3) = 2$.
Iterazione 4: arco $(3, 1)$: $\pi_1 = 1$; $p(1) = 3$.
Iterazione 5: arco $(1, 2)$: $\pi_2 = 2$; $p(2) = 1$.
Iterazione 6: arco $(2, 3)$: $\pi_3 = 1$; $p(3) = 2$.
Iterazione 7: arco $(3, 1)$: $\pi_1 = 0$; $p(1) = 3$.
Iterazione 8: arco $(1, 2)$: $\pi_2 = 1$; $p(2) = 1$.
Iterazione 9: arco $(2, 3)$: $\pi_3 = 0$; $p(3) = 2$.
Iterazione 10: arco $(3, 1)$: $\pi_1 = -1$; $p(1) = 3$.
...

Quindi, in presenza di un ciclo negativo, l'algoritmo non converge.

Consideriamo adesso il caso in cui non esistono cicli di costo negativo. In questo caso, il costo di un cammino minimo da s a d, se d è raggiungibile dall'origine s, è comunque

limitato. Inoltre, se un nodo non è raggiungibile da s, allora la sua etichetta non viene mai aggiornata rimane a $+\infty$ (se così non fosse, per il Lemma 1 dovrebbe esistere un cammino da s, cadendo in contraddizione). Tali nodi non raggiungibili, quindi, non fanno aumentare il numero di iterazioni.

Finché l'algoritmo itera, almeno un π_j viene diminuito. Se l'algoritmo iterasse all'infinito, questo π_j arriverebbe a scendere sotto il costo di un cammino ammissibile da s a j, il che non è possibile, per il Lemma 1. Pertanto, l'algoritmo converge in un numero finito di iterazioni.

Per quanto riguarda la complessità computazionale asintotica, nel caso di non esistenza di cicli negativi, bisogna considerare il numero di iterazioni nel caso peggiore. Supponiamo che i costi sugli archi siano interi[4]: ad ogni iterazione, un π_j scende di almeno una unità.

Osserviamo inoltre che:

Osservazione 1 *Se non esistono cicli negativi, un cammino minimo contiene al più* $|N|-1$ *archi.*

Infatti, se ci fossero più archi, necessariamente un nodo sarebbe toccato più di una volta; esisterebbe quindi un ciclo nel cammino di costo > 0, che contraddirrebbe l'ottimalità, o di costo $= 0$, che può essere eliminato senza perdere in ottimalità.

Sotto questa osservazione, è possible stabilire sia un limite superiore $\overline{M}$, sia un limite inferiore $\underline{M}$ per il costo del cammino minimo da s verso un qualsiasi nodo j. Ad esempio:

$$\overline{M} = (n-1) \max_{(i,j)\in A} c_{ij}$$

$$\underline{M} = \max\left\{0, (n-1) \min_{(i,j)\in A} c_{ij}\right\}$$

Possiamo pertanto porre convenzionalmente $+\infty = \overline{M}$ e, in questo modo, il numero massimo di iterazioni sarebbe di $\overline{M} - \underline{M}$ per ogni etichetta ($|N| - 1$ etichette in tutto). Ad ogni iterazione si seleziona un arco (si potrebbe fare con un massimo $|A|$ operazioni di confronto e scegliendo a caso uno degli archi che violano il vincolo duale) e si effettuano due operazioni di assegnazione (tempo costante). Abbiamo pertanto dimostrato che:

Proprietà 3 *Se il grafo non presenta cicli di costo negativo, l'algoritmo* label correcting generico *converge in* $O((|N| - 1)\,|A|\,(\overline{M} - \underline{M}))$.

Si fa notare che la complessità dipende da $\overline{M} - \underline{M}$, a sua volta dipendente dai costi sugli archi, che sono *parametri* (e non dimensione) del problema. Tale differenza potrebbe essere molto elevata e, di conseguenza, la convergenza potrebbe essere molto lenta. Si tratta infatti di una complessità computazionale *pseudo-polinomiale*.

[4]Se i costi sono razionali, basta moltiplicarli tutti per il minimo comune multiplo. Se fossero irrazionali, potrebbero esserci problemi numerici di convergenza.

4.3 Correttezza dell'algoritmo label correcting

Dimostriamo adesso la correttezza dell'algoritmo label correcting, facendo ricorso alla teoria della dualità.

Proprietà 4 *L'algoritmo label correcting risolve il problema del cammino minimo da un nodo origine s a un nodo destinazione d.*

Dimostrazione: Alla fine dell'algoritmo label correcting abbiamo a disposizione una soluzione ammissibile duale π e dei puntatori per ogni nodo i, che costruiscono un cammino di costo π_i da s a i. Consideriamo una soluzione del problema primale ottenuta ponendo $x_{ij} = 1$ se (i, j) è un arco del cammino P da s a d ottenuto partendo da d e seguendo a ritroso la catena dei puntatori ai nodi come definiti dall'algoritmo label correcting; $x_{ij} = 0$ per tutti gli altri archi. Le x_{ij} così definite soddisfano i vincoli di bilanciamento dei flussi. Abbiamo quindi a disposizione due soluzioni x e π ammissibili primale e duale, rispettivamente. Inoltre, il valore della funzione obiettivo del problema primale è:

$$\sum_{(i,j)\in A} c_{ij}x_{ij} = \sum_{(i,j)\in P} c_{ij} = \pi_d = \pi_d - 0 = \pi_d - \pi_s$$

Pertanto abbiamo una coppia di soluzioni primale-duale ammissibili con valori della funzione obiettivo coincidenti. Le sue soluzioni sono pertanto ottime (vedi Corollario 1 della teoria della dualità in programmazione lineare) e, in particolare, P è una cammino minimo da s a d. ■

4.4 Albero dei cammini minimi

Abbiamo visto il modello del cammino minimo da un'origine s a una destinazione d

$$\begin{aligned}
\min \quad & \sum_{(i,j)\in A} c_{ij}x_{ij} \\
\text{s.t.} \quad & \\
& \sum_{(i,v)\in A} x_{iv} - \sum_{(v,j)\in A} x_{vj} = \begin{cases} -1, & v = s; \\ +1, & v = d; \\ 0, & v \in N \setminus \{s, d\}. \end{cases} \\
& x_{ij} \in \mathbb{R}_+ \quad \forall\, (i,j) \in A
\end{aligned}$$

ed il corrispondente duale

$$\begin{aligned}
\max \quad & \pi_d - \pi_s \\
\text{s.t.} \quad & \pi_j - \pi_i \leq c_{ij} \quad \forall\, (i,j) \in A \\
& \pi_v \in \mathbb{R} \quad \forall\, v \in N
\end{aligned}$$

Se consideriamo il problema del cammino mimino dalla stessa origine s all'origine d', si ottiene la seguente coppia di modelli del primale

$$\begin{aligned} \min \quad & \sum_{(i,j)\in A} c_{ij}x_{ij} \\ \text{s.t.} \quad & \\ & \sum_{(i,v)\in A} x_{iv} - \sum_{(v,j)\in A} x_{vj} = \begin{cases} -1, & v = s; \\ +1, & v = d'; \\ 0, & v \in N \setminus \{s, d'\}. \end{cases} \\ & x_{ij} \in \mathbb{R}_+ \quad \forall\, (i,j) \in A \end{aligned}$$

e del duale

$$\begin{aligned} \max \quad & \pi_{d'} - \pi_s \\ \text{s.t.} \quad & \pi_j - \pi_i \leq c_{ij} \quad \forall\, (i,j) \in A \\ & \pi_v \in \mathbb{R} \quad \forall\, v \in N \end{aligned}$$

In particolare, i vincoli dei due problemi duali sono esattamente gli stessi e, di conseguenza:

> *Lo stesso algoritmo label correcting fornisce delle etichette ammissibili duali, per* tutte *le possibili destinazioni del cammino minimo.*

Inoltre, per il Lemma 1, alla fine dell'algoritmo sono disponibili, attraverso i puntatori $p(\cdot)$, cammini da s verso *tutti* i nodi $j \in N$. Quindi, è possibile partire da d' e costruire (a ritroso), un cammino da s a d', di costo $\pi_{d'}$. Questo cammino, con le stesse motivazioni utilizzate per dimostrare la correttezza dell'algoritmo label correcting, è un cammino minimo da s a d' e $\pi_{d'}$ è la sua lunghezza. Abbiamo pertanto dimostrato che

Proprietà 5 *L'algoritmo label correcting fornisce la lunghezza π_j di un cammino minimo da un'origine s verso* tutti *i nodi $j \in N$ e, attraverso i puntatori $p(\cdot)$, un cammino minimo da s, verso* tutti *gli altri nodi.*

È facile osservare come i puntatori $p(\cdot)$ definiscano una struttura gerarchica tra i nodi e, in particolare, un albero con radice in s. L'unico percorso nell'albero tra s e un nodo j definisce un cammino minimo da s verso j, il cui costo è π_j. Tale albero è detto *albero dei cammini minimi*. Possiamo pertanto affermare che

> Dato un grafo pesato e un nodo origine s, l'algoritmo label correcting permette di costruire l'albero dei cammini minimi con radice in s.

Esempio 3 *Con riferimento al grafo in Figura 2, l'albero dei cammini minimi è raffigurato in Figura 6.*

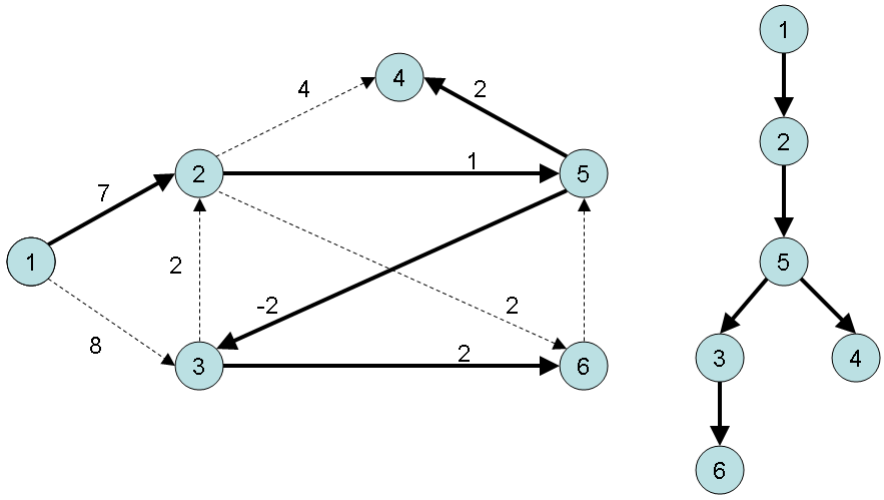

Figure 6: Albero dei cammini minimi per il grafo in Figura 2.

4.5 Grafo dei cammini minimi e cammini minimi alternativi

Attraverso la memorizzazione dei predecessori $p(\cdot)$, l'algoritmo label correcting ci permette di ottenere *uno* dei cammini minimi da s verso gli altri nodi del grafo. In realtà, ricorrendo alla teoria della dualità, è possibile ricavare *tutti* i cammini minimi da s verso un qualsiasi altro nodo.

Dato un grafo $G = (N, A)$ e le etichette ottime π derivate dall'algoritmo label correcting in relazione ad un nodo origine s, si costruisca il grafo $G_{s,\pi} = (N, A_{s,\pi})$ dove $A_{s,\pi} = \{(i, j) \in A : \pi_j = \pi_i + c_{ij}\}$.

Esempio 4 *, sia G il grafo in Figura 7a, dove sono indicate anche le etichette ottime relative all'origine* 1*, determinate dall'algoritmo label correcting. È subito evidente che esistono due cammini minimi equivalenti da* 1 *a* 4*. Il corrispondente grafo $G_{1,\pi}$ è indicato in Figura 7b. Si noti che entrambi i cammini minimi da* 1 *a* 4 *appartengano a G_π.*

Se consideriamo un qualsiasi cammino P da s verso un altro nodo $j \in N$ sul grafo $G_{s,\pi}$, possiamo costruire una soluzione primale del problema del cammino minimo da s a j ponendo $x_{ij} = 1$ per gli archi $(i, j) \in P$ e $x_{ij} = 0$ altrimenti. Tale soluzione è ammissibile primale e, inoltre, valgono le condizioni di complementarietà primale duale. Infatti

$$(\pi_j - \pi_i - c_{ij})x_{ij} = 0, \ \forall \ (i, j) \in A$$

visto che $x_{ij} = 1$ solo su archi tali che $(\pi_j - \pi_i - c_{ij}) = 0$, e che, se $(\pi_j - \pi_i - c_{ij}) < 0$, allora $x_{ij} = 0$. Abbiamo quindi a disposizione una coppia di soluzioni ammissibili primale-duale e in scarti complementari. Le due soluzioni sono pertanto ottime e, in particolare:

Un qualsiasi *cammino da s a j su $G_{s,\pi}$ è un cammino minimo da s a j.*

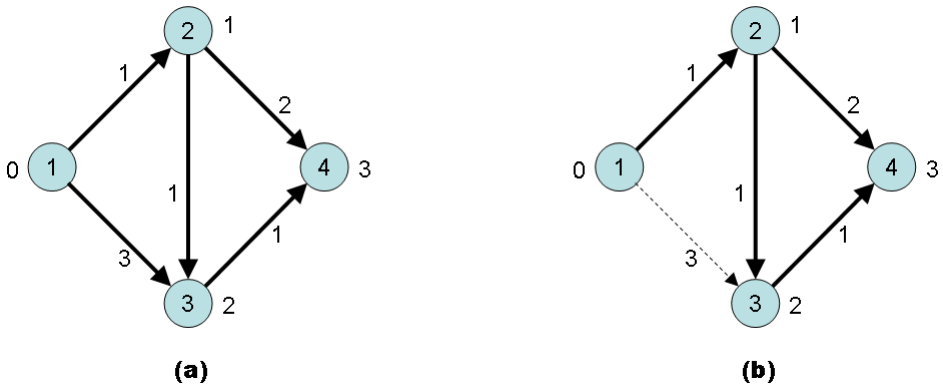

Figure 7: Un esempio di grafo dei cammini minimi.

In effetti, è possibile dimostrare anche il contrario e cioè che se P è un cammino minimo da s a j, allora P è contenuto in $G_{s,\pi}$. $G_{s,\pi}$ è pertanto detto *grafo dei cammini minimi* con origine in s e si ha:

Proprietà 6 *Dato un grafo dei cammini minimi rispetto a un nodo origine s, i cammini su tale grafo da s verso un altro nodo j sono* tutti e soli *i cammini minimi da s a j.*