Capitolo 8
Reti di flusso

Un grafo orientato è spesso un utile modello per rappresentare trasferimenti di grandezze fra luoghi diversi. In questi casi i luoghi sono rappresentati dai nodi e gli archi rappresentano le possibilità di trasferimento. È naturale allora associare agli archi dei numeri che rappresentano le grandezze trasferite.

In altri casi un grafo orientato può servire come modello per rappresentare vincoli possibili fra coppie di grandezze, più esattamente fra le loro differenze. In questo caso le grandezze sono associate ai nodi e l'arco rappresenta l'esistenza del vincolo fra le due grandezze.

L'aspetto notevole della teoria delle reti di flusso è che questi due modi diversi di 'vedere' un grafo orientato sono intimamente legati fra loro e costituiscono nient'altro che l'aspetto primale e quello duale del medesimo problema.

La teoria delle reti di flusso trova le sue origini in alcuni problemi di trasporto di tipo particolare studiati da Kantorovich [1939], Hitchcock [1941] e Koopmans [1947]. È tuttavia negli anni 50 che la teoria si sviluppa pienamente ad opera soprattutto di Dantzig, Ford e Fulkerson (si vedano Ford e Fulkerson [1962] e Dantzig [1963]). La bibliografia sulle reti di flusso è vastissima. Fra i molti testi ci limitiamo a citare i seguenti: Lawler [1976], Gondran e Minoux [1984], Rockafellar [1984], Bazaraa et al. [1990], Glover et al. [1992], Ahuja et al. [1993], Jungnickel [1999].

8.1. Definizioni

Una rete di flusso è un grafo orientato $G(N, E)$ a cui sono aggiunte delle strutture che si riferiscono alle nozioni di flusso e di potenziale. Si intende come *flusso* in una rete un'arbitraria funzione $x : E \to \mathbb{R}$. Si noti che x_e può anche essere negativo. È importante sottolineare che l'orientazione dell'arco e non costituisce una direzione 'obbligata' per il flusso x_e, ma semplicemente rappresenta una convenzione sul segno di x_e. Quindi lo stesso flusso è rappresentato sia da x_e che da $-x_e$ con e orientato in verso opposto.

In ogni nodo i vi è del flusso positivo che esce lungo un certo arco e, se $e = (i, j)$ e $x_e > 0$ oppure $e = (j, i)$ e $x_e < 0$, e del flusso negativo che entra se $e = (i, j)$ e $x_e < 0$ oppure $e = (j, i)$ e $x_e > 0$. La differenza fra il flusso uscente e quello entrante prende il nome di *divergenza* nel nodo i. Quindi la divergenza è una funzione $\mathbb{R}^E \to \mathbb{R}^N$ e viene indicata con $\operatorname{div} x$. Non è difficile vedere che

$$y := \operatorname{div} x = A\,x$$

dove A è la matrice d'incidenza nodi-archi. La divergenza nel nodo i del flusso x viene indicata con $y_i := \operatorname{div}_i(x)$. Il *principio della divergenza totale* stabilisce che $\sum_{i \in N} y_i = 0$ qualunque sia il flusso, come è facile verificare.

I flussi per i quali la divergenza è nulla in ogni nodo prendono il nome di *circolazioni*. Ogni circolazione è quindi un elemento di $\mathcal{C}$, sottospazio dei circuiti, e può dunque essere

rappresentata come combinazione lineare di circuiti. Ogni flusso può essere trasformato in una circolazione su una *rete aumentata*, che, rispetto alla rete originaria, ha in più un nodo k e archi (k, i), $\forall i \in N$, sui quali il flusso vale y_i.

In una rete si definisce per ogni arco e un *intervallo di capacità* $[c_e^-, c_e^+]$ (con estremi eventualmente illimitati) e il flusso x viene detto ammissibile rispetto alle capacità se $c_e^- \leq x_e \leq c_e^+$, $\forall e \in E$. I valori c_e^+ e c_e^- prendono il nome di *capacità in avanti* e *capacità all'indietro* dell'arco.

Per ogni taglio $Q(S)$ generato dal sottoinsieme S di nodi, il flusso passante attraverso il taglio è dato da

$$\sum_{e \in Q^+(S)} x_e - \sum_{e \in Q^-(S)} x_e = e(Q(S))\, x = u(S)\, A\, x = u(S)\, y =: y(S)$$

dove $e(Q(S))$ è il vettore d'incidenza del taglio $Q(S)$, $u(S)$ è il vettore d'incidenza del sottoinsieme S e $y(S)$ $(= \sum_{i \in S} y_i)$ viene definita come *divergenza di x nell'insieme S* e rappresenta il flusso netto che viene trasferito dall'insieme S verso il suo complementare. Le capacità sugli archi inducono in modo naturale un intervallo di *capacità di taglio*:

$$[c^-(Q), c^+(Q)] := \left[\min_x e(Q)\, x\,,\ \max_x e(Q)\, x\right] =$$

$$= \left[\sum_{e \in Q^+} c_e^- - \sum_{e \in Q^-} c_e^+\,,\ \sum_{e \in Q^+} c_e^+ - \sum_{e \in Q^-} c_e^-\right]$$

Quindi si dirà che un flusso x è ammissibile per una capacità di taglio se

$$c^-(Q(S)) \leq e(Q(S))\, x \leq c^+(Q(S))$$

ovvero

$$c^-(Q(S)) \leq y(S) \leq c^+(Q(S))$$

Si noti che $y(S) = -y(N \backslash S)$ e che $c^-(Q(S)) = -c^+(Q(N \backslash S))$, quindi un flusso è ammissibile per tutte le capacità di taglio se

$$y(S) \leq c^+(Q(S)) \quad \forall S \subset N$$

Ovviamente un flusso ammissibile per le capacità è anche ammissibile per tutte le capacità di taglio. Vedremo più avanti che un flusso ammissibile per tutte le capacità di taglio è anche ammissibile per le capacità.

Si intende come *potenziale* in una rete un'arbitraria funzione $u : N \to \mathbb{R}$. Per ogni arco $e = (i, j)$ si definisce *tensione* nell'arco e la quantità

$$v_e := u_j - u_i$$

La tensione in una rete è pertanto una funzione $v : \mathbb{R}^N \to \mathbb{R}^E$. Si noti che $v = -u\, A$ e quindi ogni tensione è un elemento di $\mathcal{Q}$, sottospazio dei tagli, e può dunque essere rappresentata come combinazione lineare di tagli. Si ricava quindi (vedi capitolo 2) che tensioni e circolazioni sono ortogonali fra loro. Inoltre vale, per qualsiasi flusso e potenziale, $u\, y + v\, x = 0$.

In una rete si definisce inoltre per ogni arco e un *intervallo di espansione* (in inglese *span interval*) $[d_e^-, d_e^+]$ (con estremi eventualmente illimitati) e il potenziale (e conseguentemente la tensione) viene detto ammissibile rispetto alle espansioni se

$$d_e^- \leq u(j) - u(i) = v_e \leq d_e^+ \qquad \forall e = (i, j)$$

I valori d_e^+ e d_e^- prendono il nome di *lunghezza in avanti* e *lunghezza all'indietro* dell'arco. Per ogni cammino $P : s \to t$ la differenza di potenziale $u(t) - u(s)$ su P è data da

$$u(t) - u(s) = \sum_{e \in P^+} v_e - \sum_{e \in P^-} v_e = e(P)\, v$$

Le espansioni sugli archi inducono in modo naturale un intervallo di *espansione di cammino*

$$[d^-(P), d^+(P)] = \left[\min_v e(P)\, v\,,\, \max_v e(P)\, v\right] =$$

$$\left[\sum_{e \in P^+} d_e^- - \sum_{e \in P^-} d_e^+\,,\, \sum_{e \in P^+} d_e^+ - \sum_{e \in P^-} d_e^-\right] \tag{8.1}$$

dove $d^+(P)$ viene definita come *lunghezza del cammino* $P : s \to t$ e quindi risulta anche che $-d^-(P)$ è la lunghezza del cammino $-P : t \to s$.

Siccome per ogni circuito C vale $e(C)\, v = 0$, l'ammissibilità di un potenziale implica $d^-(C) \le 0 \le d^+(C)$ per ogni circuito C. Si noti inoltre che $d^-(C) = -d^+(-C)$ e quindi un potenziale è ammissibile rispetto alle espansioni solo se $d^+(C) \ge 0$, $\forall C$. Vedremo più avanti che questa condizione è anche sufficiente all'esistenza di un potenziale ammissibile.

8.1 Esercizio. Data una rete con fissati valori di divergenza sui nodi si progetti un algoritmo di complessità lineare che calcola un flusso ammissibile in assenza di vincoli di capacità. (suggerimento: si sfruttino strutture ad albero). ∎

8.2. Problemi primale e duale

Definita una rete di flusso, consideriamo ora una classe particolare di problemi di ottimizzazione. In questa classe la funzione da minimizzare (definita sul flusso oppure sulle tensioni) è convessa, lineare a tratti e separabile arco per arco. Queste ipotesi danno luogo ad una teoria ricca di risultati interessanti e, allo stesso tempo, il modello risultante è applicabile in molti casi di interesse pratico.

L'ipotesi di convessità permette di trasferire l'eventuale vincolo di capacità o di espansione nella definizione stessa di funzione obiettivo. Quindi, anche se non esplicitamente presente, il vincolo va comunque inteso operante. Ad esempio, se in un arco il flusso è vincolato nell'intervallo $[0, c]$ ed al flusso x è associato un costo lineare $d\, x$, a tale arco viene associata la seguente funzione convessa di costo

$$f(x) = \begin{cases} +\infty & \text{se } x < 0 \\ d\, x & \text{se } 0 \le x \le c \\ +\infty & \text{se } x > c \end{cases} \tag{8.2}$$

Ovviamente si possono considerare casi più complessi di funzioni di costo. Tuttavia vedremo più avanti che ogni rete con funzioni di costo convesse e lineari a tratti può essere trasformata in una rete con un numero più elevato di archi in cui la funzione di costo è del tipo espresso in (8.2), cioè lineare con un vincolo di capacità e quindi il problema è di fatto un problema di programmazione lineare. Consideriamo dapprima il seguente problema.

8.2 Definizione. *Si definisce come* problema primale *il seguente problema*

$$\varphi := \min\ \Phi(x) := \sum_{e\in E} f_e(x_e)$$
$$A\,x = b$$

dove ogni $f_e(x_e) : \mathbb{R} \to \mathbb{R}\cup+\infty$ *è una funzione convessa e lineare a tratti del flusso sull'arco* e.

Il problema primale è quindi definito soltanto sul flusso. Un flusso x viene detto *ammissibile* se $A\,x = b$ e $\Phi(x) < +\infty$. Come già detto un'ovvia condizione necessaria di ammissibilità è data da $b(S) \le c(Q(S))$ per ogni sottoinsieme $S \subset N$, dove le capacità sono dedotte dalle funzioni f_e. Possiamo derivare la funzione duale dalla funzione Lagrangiana $\Phi(x) + u\,(A\,x - b)$:

$$\Psi(u,v) := \inf_x \Phi(x) + u\,(A\,x - b) = -\sum_{i\in N} u_i b_i + \sum_{e\in E} \inf_{x_e}\big(f_e(x_e) - (u_j - u_i)\,x_e\big) =$$
$$-\sum_{i\in N} u_i\,b_i - \sum_{e\in E} \sup_{x_e}\big(v_e\,x_e - f_e(x_e)\big) = -\sum_{i\in N} u_i\,b_i - \sum_{e\in E} g_e(v_e)$$

dove $g(v) := \sup_x v\,x - f(x)$ è la funzione coniugata di $f(x)$ (vedi sezione 4.9) e soddisfa la relazione $f(x) = \sup_v v\,x - g(v)$ (cioè f è anche la funzione coniugata di g). Possiamo a questo punto definire il seguente problema.

8.3 Definizione. *Si definisce come* problema duale *il seguente problema*

$$\psi := \sup\ \Psi(u,v)$$
$$v = -u\,A$$

.

Si noti che il problema duale è definito sui potenziali e le tensioni. Quindi fra flussi e tensioni vi è una precisa relazione di dualità. Le funzioni $g_e(v)$ possono incorporare intervalli di espansione ed una tensione v viene quindi detta ammissibile se esiste u tale che $v = -u\,A$ e $\Psi(u,v) > -\infty$. Come già detto un'ovvia condizione necessaria di ammissibilità è data da $d(C) \ge 0$ per ogni circuito C, dove le lunghezze sono dedotte dalle funzioni g_e.

Per la dualità debole è sempre vero che $\psi \le \varphi$. Tuttavia le ipotesi di convessità e linearità a tratti comportano la dualità forte a meno che il problema primale e quello duale siano ambedue non ammissibili.

Un legame molto stretto fra le soluzioni del problema primale e quelle del duale è fornito dalle seguenti condizioni di ottimalità. Nelle notazioni che seguono si assume tacitamente che $\hat{x}$ soddisfa $A\,\hat{x} = b$ e che $\hat{v}$ soddisfa $\hat{v} = -\hat{u}\,A$ per qualche potenziale $\hat{u}$.

8.4 Lemma. *La condizione*

$$g_e(\hat{v}_e) + f_e(\hat{x}_e) = \hat{v}_e\,\hat{x}_e \qquad \forall e \in E \tag{8.3}$$

è necessaria e sufficiente all'ottimalità di $\hat{x}$ *e* $\hat{v}$.

Dimostrazione.

$$\varphi = \psi \iff \sum_e f_e(\hat{x}_e) = -\hat{u}\,b - \sum_e g_e(\hat{v}_e) \iff \sum_e f_e(\hat{x}_e) = -\hat{u}\,A\,\hat{x} - \sum_e g_e(\hat{v}_e) \iff$$
$$\iff \sum_e f_e(\hat{x}_e) = \hat{v}\,\hat{x} - \sum_e g_e(\hat{v}_e) \iff f_e(\hat{x}_e) = \hat{v}_e\,\hat{x}_e - g_e(\hat{v}_e) \qquad \forall e$$

Siano f^- e f^+ rispettivamente le derivate sinistra e destra di f.

8.5 Lemma. *La condizione*

$$f_e^-(\hat{x}_e) \le \hat{v}_e \le f_e^+(\hat{x}_e) \qquad \forall e \in E$$

è necessaria e sufficiente all'ottimalità di $\hat{x}$ *e* $\hat{v}$.

Dimostrazione. Da (8.3) e dalla definizione di funzione coniugata ogni $\hat{x}_e$ massimizza la funzione concava $h(x_e) := \hat{v}_e x_e - f_e(x_e)$. Questo è equivalente a $h^-(\hat{x}_e) \ge 0$, $h^+(\hat{x}_e) \le 0$, cioè $\hat{v}_e - f_e^-(\hat{x}_e) \ge 0$ e $\hat{v}_e - f_e^+(\hat{x}_e) \le 0$. ∎

Per ogni $e \in E$ si può definire il seguente insieme $\Gamma_e \subset \mathbb{R}^2$ chiamato *curva caratteristica* dell'arco e:

$$\Gamma_e := \big\{(x_e, v_e) : f_e^-(x_e) \le v_e \le f_e^+(x_e)\big\} \tag{8.4}$$

Si noti che ogni curva Γ_e è semplicemente il grafico della derivata di f_e più tutti i segmenti verticali necessari a riempire i salti di discontinuità (si veda un esempio di funzione $f(x)$ e corrispondente Γ in figura 8.1). Quindi, poiché ogni f_e è convessa, ogni curva Γ_e è monotona nel senso che $(x, v) + \mathbb{R}^2_+ + \mathbb{R}^2_- \supset \Gamma$, $\forall (x, v) \in \Gamma$. Inoltre la linearità a tratti implica che ogni curva Γ_e è a scalini.

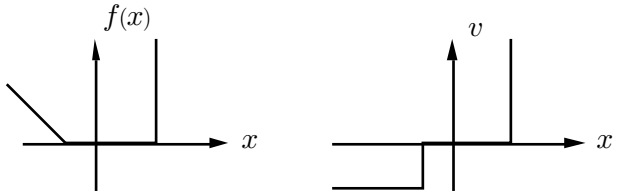


Figura 8.1

Una curva Γ_e (o equivalentemente la funzione f_e che la definisce) si può assegnare semplicemente tramite due liste finite di numeri crescenti, quella dei valori di x corrispondenti ai tratti verticali di Γ (ai punti di rottura di f_e), che chiameremo valori di capacità $c_e^1, \dots, c_e^p$ di Γ_e, e quella dei valori di v corrispondenti ai tratti orizzontali di Γ (alle pendenze di f_e), che chiameremo valori di lunghezza $d_e^1, \dots, d_e^q$ di Γ_e (si noti che $|p - q| \le 1$). Inoltre è richiesta l'informazione se il primo tratto di Γ_e è verticale oppure orizzontale. Come si vede la dimensione dei dati, dal punto di vista della complessità computazionale, è lineare rispetto al numero totale di 'scalini' delle curve caratteristiche.

8.6 Esercizio. Si dimostri che ogni curva Γ si può anche ottenere dal grafico della derivata della corrispondente funzione coniugata g (intesa come funzione di v) più tutti i segmenti orizzontali necessari a riempire i salti di discontinuità della derivata di g. ∎

Una conseguenza immediata della definizione (8.4) e del lemma 8.5 è il seguente risultato fondamentale:

8.7 Teorema. $(\hat{x}, \hat{v})$ *sono ottimi se e solo se* $(\hat{x}_e, \hat{v}_e) \in \Gamma_e$, $\forall e$. ∎

Come corollario di questo risultato si ha

8.8 Corollario
– (condizione primale) $\hat{x}$ *è ottimo se e solo se esiste* $\hat{u}$ *tale che* $\hat{v}_e \in [f_e^-(\hat{x}_e), f_e^+(\hat{x}_e)]$, $\forall e \in E$;
– (condizione duale) $\hat{u}$ *è ottimo se e solo se esiste* $\hat{x}$ *tale che* $\hat{x}_e \in [g_e^-(\hat{v}_e), g_e^+(\hat{v}_e)]$, $\forall e \in E$.
∎

Quindi per verificare l'ottimalità di una soluzione si può risolvere un problema di ammissibilità. Come vedremo più avanti i due problemi di ammissibilità del corollario 8.8 possono essere risolti tramite due problemi particolarmente importanti in molti problemi di ottimizzazione combinatoria, cioè i problemi del cammino minimo e del massimo flusso.

8.9 Esercizio. Sia dato un grafo orientato i cui archi costituiscono un ciclo orientato. Ad ogni arco e sono associati un costo c_e ed un intervallo di capacità $[0, \infty)$. Ad ogni nodo i è associato un valore di divergenza b_i ($\sum_i b_i = 0$). Sia $c_e \geq 0$, $\forall e$. Si progetti un algoritmo lineare che trovi il flusso di costo minimo, e si dimostri l'ottimalità della soluzione. Dimostrare che la soluzione trovata è ottima anche nel caso di costi generici purché $\sum_e c_e \geq 0$. ∎

8.10 Esercizio. I risultati esposti sono validi, a parte qualche ipotesi tecnica (come la stabilità) anche per generiche funzioni convesse. Si interpreti un arco come una resistenza elettrica, il flusso come una corrente elettrica lungo un arco e la tensione come una tensione elettrica ai capi dell'arco. In queste condizioni vale per ogni arco la legge di Ohm, cioè $v_e/x_e = r_e$ con r_e resistenza dell'arco e. Si può intepretare la legge di Ohm come un minimo di una opportuna grandezza fisica? ∎

8.11 Esercizio. Si consideri il problema $\min\{\sum_e f_e(x_e) : A\,x = b\}$ dove però A è una matrice qualsiasi (e non una matrice d'incidenza). Definendo $v_j := u\,A^j$ come tensione della colonna j per assegnati potenziali di riga u, si ridefinisca la condizione in termini di curva caratteristica (della colonna) e si confronti con le condizioni di complementarità. ∎

8.3. Trasformazioni di reti

Una rete può essere trasformata in un'altra rete equivalente tramite una serie di operazioni elementari sugli archi e sulle curve Γ. Per rete equivalente si intende una rete la cui soluzione ottima può fornire attraverso un calcolo elementare o in modo diretto la soluzione del problema originale. Le operazioni elementari sono le seguenti:

i. Inversione: l'orientazione di una arco $e = (i, j)$ può essere invertita e la curva Γ_e viene sostituita da $-\Gamma_e$.

ii. Traslazione di flusso: la curva Γ_e può essere traslata verso destra della quantità Δ e corrispondentemente b_i viene aumentato di Δ e b_j viene diminuito di Δ.

iii. Fusione e divisione in parallelo: due archi paralleli e_1 e e_2 possono essere fusi in un singolo arco e, la cui curva Γ_e è definita da

$$\Gamma_e := \{(x, v) : x = x_1 + x_2\,,\ (x_1, v) \in \Gamma_{e_1}\,,\ (x_2, v) \in \Gamma_{e_2}\} \tag{8.5}$$

Analogamente un arco e può essere diviso in due archi paralleli e_1 e e_2 tali che (8.5) sia soddisfatta.

iv. Fusione e divisione in serie: due archi in serie e_1 e e_2 (senza altri archi incidenti nel nodo comune e con conservazione del flusso nel nodo comune) possono essere fusi in un singolo arco e, la cui curva Γ_e è definita da

$$\Gamma_e : \{(x, v) : v = v_1 + v_2\,,\ (x, v_1) \in \Gamma_{e_1}\,,\ (x, v_2) \in \Gamma_{e_2}\} \tag{8.6}$$

Analogamente un arco e può essere diviso in due archi in serie e_1 e e_2 tali che (8.6) sia soddisfatta.

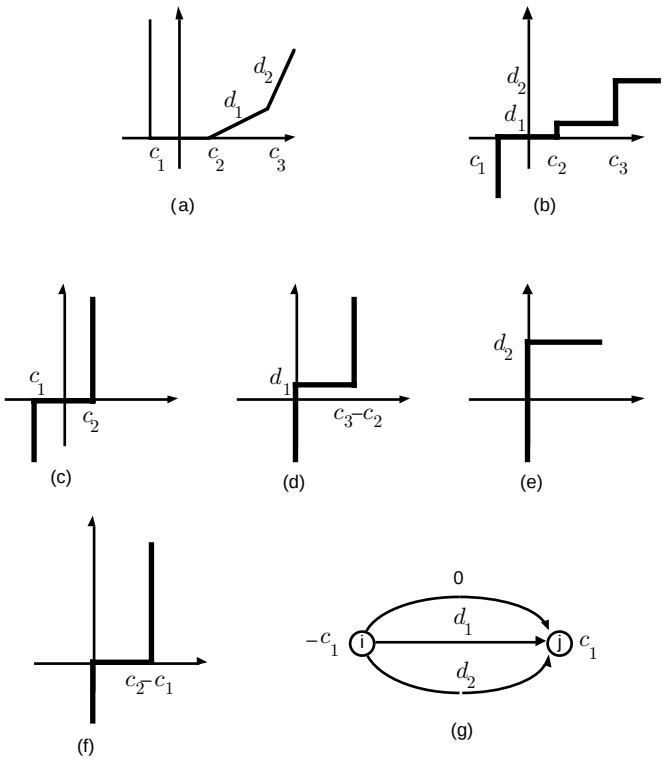


FIGURA 8.2

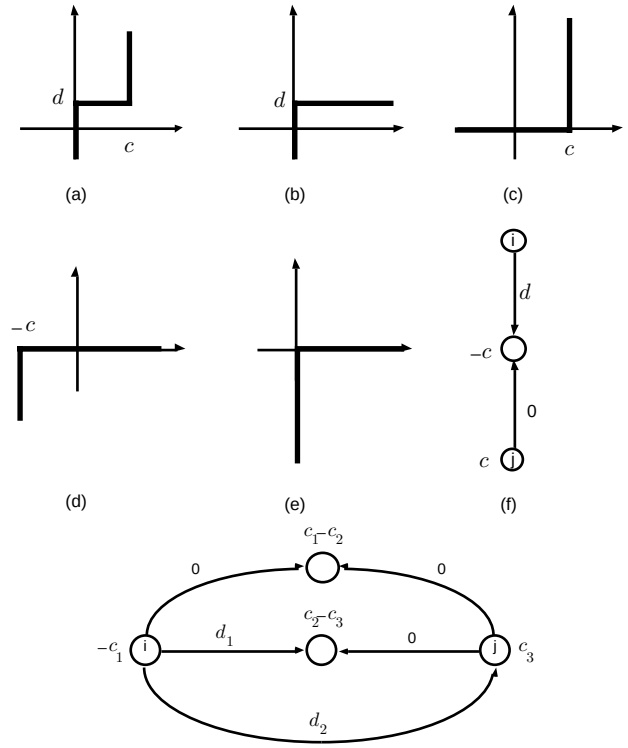


FIGURA 8.3

Queste operazioni elementari possono essere usate sia per ridurre una rete (ad esempio una rete serie-parallelo viene ridotta ad un singolo arco) o per estendere la rete tramite divisioni di archi in modo da avere lo stesso tipo di curve Γ su tutti gli archi. Infatti i due seguenti tipi di curva Γ sono particolarmente utili algoritmicamente. Una curva caratteristica viene detta *elementare* se corrisponde alla seguente funzione di costo

$$f(x) = \begin{cases} +\infty & \text{se } x < 0 \\ d\,x & \text{se } x \geq 0 \end{cases}$$

mentre viene detta *semplice* se

$$f(x) = \begin{cases} +\infty & \text{se } x < 0 \\ d\,x & \text{se } 0 \leq x \leq c \\ +\infty & \text{se } x > c \end{cases}$$

Se tutte le funzioni sono elementari o semplici il problema è un caso particolare di programmazione lineare e può essere anche risolto con uno speciale metodo del simplesso che verrà esaminato alla fine del capitolo. In figura 8.2 viene illustrata la trasformazione in archi semplici di un arco (i, j), la cui funzione di costo è rappresentata in (a) e la corrispondente curva caratteristica in (b). Questa curva presenta tre valori di capacità c_1, c_2, c_3 e tre valori di lunghezza 0, d_1 e d_2. L'arco può essere sostituito da tre archi paralleli con curve caratteristiche rispettivamente in (c), (d) ed (e). Le curve in (d) ed (e) hanno la semiretta verticale di sinistra naturalmente sovrapposta all'asse verticale. La curva in (c) deve essere traslata verso destra di $-c_1$ (o equivalentemente verso sinistra di c_1) e si ottiene così la curva in (f). Questa traslazione comporta la variazione della divergenza nei due nodi come indicato in (g). (come si sarebbe dovuto procedere se la curva caratteristica presentava in $-\infty$ un tratto orizzontale?). In figura 8.3 viene illustrata la trasformazione di un arco semplice (i, j) in archi elementari. L'arco, con curva caratteristica indicata in (a), viene dapprima trasformato in due archi in serie con curve caratteristiche rispettivamente in (b) e (c). La curva in (b) è già di tipo elementare. Per trasformare la curva in (c) in elementare bisogna dapprima invertire l'arco e ottenere la curva in (d) e traslare di c verso destra ottenendo (e). I due archi risultanti sono indicati in (f). Infine viene indicata la rete equivalente all'arco (i, j) della figura 8.2 formata tutta da archi elementari utilizzando le due trasformazioni in successione.

8.12 Esercizio. Si trovi il flusso di costo minimo per la rete in figura 8.4, dove le funzioni di costo per ogni arco sono del tipo $f_e(x_e) := c_e|x_e|$ (con c_e indicato vicino ad ogni arco) e i valori di b_i sono i numeri più grandi indicati vicino ai nodi. Risolvere mediante riduzioni serie-parallelo. ∎

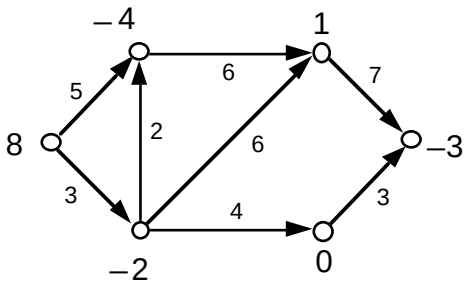


Figura 8.4

8.4. Rete residua

Data una rete G e assegnati un flusso x ed un potenziale u, la *Rete Residua* $RG(x,u)$ (o più semplicemente RG) di G viene definita nel modo seguente. I nodi e gli archi di RG sono gli stessi di G. Per ogni arco e vengono definiti un intervallo di *capacità residua* $[-\gamma_e^-, \gamma_e^+]$ ed un intervallo di *espansione residua* $[-\delta_e^-, \delta_e^+]$ dove le *lunghezze residue* δ^+ e δ^- e le *capacità residue* γ^+ e γ^- sono definite da:

$$\begin{aligned} \delta_e^+ &:= \max\{0\,;\, f_e^+(x_e) - v_e\} \qquad & \gamma_e^+ &:= \max\{0\,;\, g_e^+(v_e) - x_e\} \\ \delta_e^- &:= \max\{0\,;\, v_e - f_e^-(x_e)\} \qquad & \gamma_e^- &:= \max\{0\,;\, x_e - g_e^-(v_e)\} \end{aligned} \tag{8.7}$$

(si è adottata la convenzione che le derivate di una funzione convessa hanno valore $+\infty$ ($-\infty$) su punti alla 'destra' ('sinistra') del dominio effettivo). I valori definiti in (8.7) corrispondono, per le capacità, al massimo spostamento orizzontale possibile per un punto (x_e, v_e) purché lo spostamento termini sulla curva Γ_e e, per le lunghezze, al massimo spostamento verticale possibile per un punto (x_e, v_e) purché lo spostamento termini sulla curva Γ_e. Più in particolare dobbiamo considerare i seguenti casi (si veda anche la figura 8.5 dove si è indicato quali grandezze residue siano nulle tramite il simbolo 0 e quali siano positive con il simbolo $>$):

(a) Se un punto sta alla destra e sotto la curva solo spostamenti orizzontali verso sinistra e verticali verso l'alto possono essere ammessi, e quindi $\gamma_e^- > 0$, $\delta_e^+ > 0$, $\gamma_e^+ = 0$, $\delta_e^- = 0$.

(b) Se un punto sta alla sinistra e sopra la curva solo spostamenti orizzontali verso destra e verticali verso il basso possono essere ammessi, e quindi $\gamma_e^+ > 0$, $\delta_e^- > 0$, $\gamma_e^- = 0$, $\delta_e^+ = 0$.

(c) Se un punto sta strettamente all'interno di un tratto orizzontale della curva solo spostamenti orizzontali sono ammessi e quindi $\gamma_e^- > 0$, $\delta_e^+ = 0$, $\gamma_e^+ > 0$, $\delta_e^- = 0$.

(d) Se un punto sta strettamente all'interno di un tratto verticale della curva solo spostamenti verticali sono ammessi e quindi $\gamma_e^- = 0$, $\delta_e^+ > 0$, $\gamma_e^+ = 0$, $\delta_e^- > 0$.

(e) Se un punto sta su un estremo destro di un tratto orizzontale (o equivalentemente sull'estremo basso di un tratto verticale) la situazione è analoga al caso (a). In questo caso gli spostamenti ammessi sono pari all'ampiezza del tratto verticale o del tratto orizzontale.

(f) Se un punto sta su un estremo sinistro di un tratto orizzontale (o equivalentemente sull'estremo alto di un tratto verticale) la situazione è analoga al caso (b). In questo caso gli spostamenti ammessi sono pari all'ampiezza del tratto verticale o del tratto orizzontale.

Non si esclude nei casi indicati di ottenere capacità o lunghezze residue infinite. Questo avviene nei casi (a) e (b) quando lo spostamento ammesso non raggiunge mai la curva e nei rimanenti casi quando l'ampiezza del tratto orizzontale o verticale è infinita così da permettere uno spostamento infinito.

È importante notare che un arco ha lunghezza residua nulla se e solo se ha capacità residua positiva. La capacità residua $\gamma(Q)$ di un taglio Q e la lunghezza residua $\delta(P)$ di un cammino sono definite come nel caso generale usando i valori di capacità residua e lunghezza residua. Inoltre possiamo definire altri due concetti. La capacità residua di un cammino P in RG è definita come

$$\gamma(P) := \min\left\{ \min_{e\in P^+} \gamma_e^+ \,;\, \min_{e\in P^-} \gamma_e^- \right\}$$

e l'espansione residua di un taglio Q in RG come

$$\delta(Q) := \min\left\{ \min_{e\in Q^+} \delta_e^+ \,;\, \min_{e\in Q^+} \delta_e^- \right\}$$

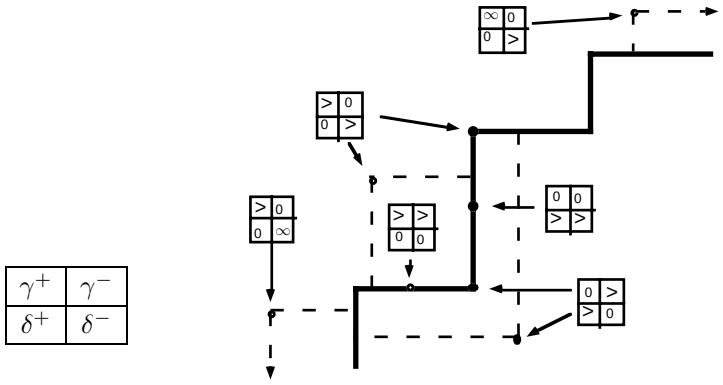


FIGURA 8.5

Quindi un cammino P in RG ha lunghezza residua nulla se e solo se ha capacità residua positiva ed un taglio Q in RG ha capacità residua nulla se e solo se ha espansione residua positiva. Inoltre cammini e tagli sono legati dalla seguente importante proprietà che non è altro che una versione particolare del lemma di Farkas, dovuta a Minty [1960]:

8.13 TEOREMA. (Lemma di Minty) *Assegnati due nodi s e t, si verifica esattamente una delle due seguenti condizioni:*
– esiste un cammino $P : s \to t$ di lunghezza residua nulla (e capacità residua positiva);
– esiste un taglio $Q : s \downarrow t$ di capacità residua nulla (ed espansione residua positiva).

DIMOSTRAZIONE. La dimostrazione è di tipo costruttivo. Si definisca $S := \{s\}$. Se esiste qualche arco nel taglio generato da S con lunghezza residua nulla si aggiunge ad S l'altro estremo dell'arco. Questa procedura termina se si aggiunge ad S il nodo t, nel qual caso si è ottenuto un cammino di lunghezza nulla oppure se nel taglio generato da S non esiste alcun arco a lunghezza residua nulla. In questo caso ogni arco del taglio ha capacità residua nulla e si è trovato il taglio Q a capacità residua nulla. ∎

Come conseguenza del teorema, per ogni coppia di nodi s e t esiste sempre la possibilità o di variare il flusso di una quantità positiva su un cammino da s a t, oppure di variare il potenziale di una quantità positiva sui nodi non in S.

L'importanza di circuiti a lunghezza residua nulla e di tagli a capacità residua nulla consiste nel fatto che gli uni costituiscono direzioni di discesa per il problema primale e gli altri direzioni di ascesa per il problema duale. Consideriamo la derivata direzionale $D\Phi(x, \Delta x)$ della funzione primale secondo una direzione di flusso Δx. Siccome le uniche direzioni ammissibili sono circolazioni, e le circolazioni si possono esprimere come combinazioni lineari di circuiti, possiamo limitarci a considerare direzioni Δx che siano vettori d'incidenza di un circuito. Quindi

$$D\Phi(x, \Delta x) = \sum_{e \in C} Df_e(x_e, \Delta x_e) = \sum_{e \in C} \max\left\{ f_e^-(x_e)\,\Delta x_e\,;\, f_e^+(x_e)\,\Delta x_e \right\} =$$

$$\sum_{e \in C^+} f_e^+(x_e) - \sum_{e \in C^-} f_e^-(x_e)$$

Dalla proprietà $\sum_{e \in C^+} v_e - \sum_{e \in C^-} v_e = 0$ e dalle definizioni (8.7) abbiamo

$$D\Phi(x, \Delta x) = \sum_{e \in C^+} (f_e^+(x_e) - v_e) + \sum_{e \in C^-} (v_e - f_e^-(x_e)) \le \sum_{e \in C^+} \delta_e^+ + \sum_{e \in C^-} \delta_e^- = \delta(C) \quad (8.8)$$

Quindi se $\delta(C) = 0$ si ha $D\Phi(x, \Delta x) \leq 0$. La diseguaglianza in (8.8) è stretta se $(x_e, v_e) \notin \Gamma_e$ per almeno uno degli archi del circuito C. Abbiamo così dimostrato:

8.14 Teorema. *Se esiste un circuito C di lunghezza residua nulla ed esiste $e \in C$ tale che $(x_e, v_e) \notin \Gamma_e$, allora, aumentando il flusso lungo il circuito C di una quantità positiva, la funzione primale diminuisce di una quantità positiva.* ∎

Per il problema duale dobbiamo considerare direzioni ammissibili $(\Delta u, \Delta v)$ della funzione duale $\Psi(u, v)$. Quindi Δv deve appartenere allo spazio dei tagli e possiamo limitarci a considerare vettori d'incidenza $\Delta u = e(\bar{S})$ e $\Delta v = e(Q)$ per un certo insieme di nodi S (dove $\bar{S}$ è l'insieme complementare) che definisce il taglio Q. Quindi

$$D\Psi(u, v, \Delta u, \Delta v) = -\sum_{i \in \bar{S}} b_i - \sum_{e \in Q} Dg_e(v_e, \Delta v_e) = b(S) - \sum_{e \in Q^+} g_e^+(v_e) + \sum_{e \in Q^-} g_e^-(v_e)$$

Da

$$b(S) = e(S)\, b = e(S)\, A\, x = e(Q)\, x = \sum_{e \in Q^+} x_e - \sum_{e \in Q^-} x_e$$

e dalle definizioni (8.7) si ha

$$D\Psi(u, v, \Delta u, \Delta v) = \sum_{e \in Q^+} (x_e - g_e^+(v_e)) + \sum_{e \in Q^-} (g_e^-(v_e) - x_e) \geq -\sum_{e \in Q^+} \gamma_e^+ - \sum_{e \in Q^-} \gamma_e^- = -\gamma(Q) \tag{8.9}$$

Quindi se $\gamma(Q) = 0$ si ha $D\Psi(u, v, \Delta u, \Delta v) \geq 0$. La diseguaglianza in (8.9) è stretta se $(x_e, v_e) \notin \Gamma_e$ per almeno uno degli archi del taglio Q. Abbiamo così dimostrato:

8.15 Teorema. *Se esiste un taglio Q di capacità residua nulla ed esiste $e \in Q$ tale che $(x_e, v_e) \notin \Gamma_e$, allora, aumentando il potenziale secondo il taglio Q di una quantità positiva, la funzione duale aumenta di una quantità positiva.* ∎

8.5. Massimo flusso

Il problema del massimo flusso (MFP) è un particolare problema primale in cui l'obiettivo è la massimizzazione del flusso che può essere trasferito da un determinato nodo s, indicato come *sorgente*, ad un altro determinato nodo t, indicato come *pozzo*. I vincoli a cui è sottoposto il flusso sono quello di capacità su tutti gli archi (supponiamo che per ogni intervallo di capacità $[c_e^-, c_e^+]$ sia $c_e^- \leq 0$ e $c_e^+ \geq 0$) e la conservazione del flusso in tutti i nodi tranne la sorgente ed il pozzo.

È utile, solo a scopo formale, aggiungere alla rete un arco e_0 dal pozzo alla sorgente con intervallo di capacità illimitato $(-\infty, \infty)$. In questo modo l'obiettivo è la massimizzazione del flusso in e_0 e la conservazione del flusso deve essere soddisfatta su tutti i nodi (sorgente e pozzo inclusi).

Nel problema del massimo flusso simmetrico (SMFP) la limitazione del flusso in ogni arco dipende dal valore assoluto del flusso, quindi $-c_e^- = c_e^+ =: c_e$. Nel problema del massimo flusso diretto (DMFP) il flusso può circolare solo nel verso dell'arco e quindi $c_e^- = 0$, $c_e^+ =: c_e$. Si noti che, in base all'ipotesi sul segno delle capacità, una soluzione ammissibile, corrispondente al flusso nullo, è immediatamente disponibile.

Il problema del massimo flusso può essere espresso come

$$\begin{aligned} \min \quad & -x_{e_0} + \sum_{e \in E \setminus e_0} f_e(x_e) \\ & A\,x = 0 \end{aligned}$$

dove ogni $f_e(x_e)$ (tranne e_0) è definita come

$$f_e(x_e) := \begin{cases} 0 & \text{se } c_e^- \le x_e \le c_e^+ \\ +\infty & \text{altrimenti} \end{cases}$$

e ovviamente $f_{e_0}(x_{e_0}) := -x_{e_0}$. Le funzioni coniugate sono date da

$$g_e(v_e) = \max\left\{c_e^- v_e; c_e^+ v_e\right\}, \quad e \neq e_0, \qquad g_{e_0}(v_{e_0}) = \begin{cases} 0 & \text{se } v_{e_0} = -1 \\ +\infty & \text{altrimenti} \end{cases}$$

(si noti che per il SMFP $g_e(v_e) = c_e|v_e|$, $e \neq e_0$). La funzione g_{e_0} ha l'effetto di imporre $u_s = 0$ e $u_t = 1$ (il potenziale è definito a meno di una costante additiva). Quindi il problema duale è equivalente a

$$\begin{aligned} \min \quad & \sum_{e \neq e_0} \max\{c_e^- v_e\,,\, c_e^+ v_e\} \\ & u_s = 0 \qquad u_t = 1 \end{aligned} \tag{8.10}$$

oppure

$$\begin{aligned} \min \quad & \sum_{e \neq e_0} -c_e^- v_e^- + c_e^+ v_e^+ \\ & v^+ - v^- = -u\,A \\ & v_e^+ \ge 0\,, \quad v_e^- \ge 0 \qquad \forall e \\ & u_s = 0 \qquad u_t = 1 \end{aligned} \tag{8.11}$$

In seguito alla totale unimodularità della matrice dei vincoli in (8.11) i valori di potenziale possono essere ristretti senza perdita di generalità ai valori interi. Vogliamo ora dimostrare che addirittura i valori di potenziale possono essere ristretti ai valori 0 e 1. Si consideri una soluzione ammissibile tale che $\max_i u_i = k > 1$. Sia $S := \{i : u_i = k\}$ e consideriamo il taglio Q definito da S. Si ha $v_e \le -1$ se $e \in Q^+$ e $v_e \ge 1$ se $e \in Q^-$. Un abbassamento uniforme del potenziale su S varia le tensioni sul taglio in modo da non peggiorare la funzione obiettivo, essendo i costi non negativi. Quindi si può porre $u_i := u_i - 1$ per $i \in S$. Proseguendo ricorsivamente e ragionando in modo simmetrico per i valori negativi del potenziale, si perviene ad una soluzione di soli 1 e 0 a costo non peggiore. Quindi possiamo restringere i potenziali di (8.11) ai soli valori 0-1 con la garanzia di non perdere l'ottimo.

Questo fatto stabilisce una corrispondenza biunivoca fra i valori di potenziale e i sottoinsiemi $S := \{i \in N : u_i = 0\}$. Inoltre ogni sottoinsieme S determina un taglio $Q(S)$ e la tensione è diversa da zero solo sugli archi del taglio. In questo modo la somma in (8.10) è semplicemente la capacità del taglio $Q(S)$.

Quindi il problema duale è il seguente problema denominato *problema del minimo taglio*:

$$\min\{c(Q(S)) : S \subset N,\, s \in S,\, t \notin S\}$$

cioè si tratta di determinare un sottoinsieme $S \subset N$ tale che $s \in S$ e $t \notin S$ la cui capacità di taglio $c(Q(S))$ sia minima. Si noti che si è potuta stabilire una equivalenza fra un problema combinatorio quale quello del minimo taglio e un problema continuo quale quello del massimo

Algoritmo di Ford-Fulkerson
input(G, c^-, c^+); $(* \; c^- \leq \mathbf{0} \leq c^+ \; *)$
  $x_e := 0$, $\forall e \in E$;
  repeat
    if $\exists \; P : s \to t : \gamma(P) > 0$
    then $x_e := x_e + \gamma(P)$, $e \in P^+$; $x_e := x_e - \gamma(P)$, $e \in P^-$;
  until $\exists \; Q : s \downarrow t : \gamma(Q) = 0$;
output(x, Q).

flusso. Questa possibilità è essenzialmente dovuta alla totale unimodularità della matrice dei vincoli. La dualità forte può essere inoltre dimostrata per altra via tramite il seguente ragionamento.

Dato un flusso ammissibile x e il valore di potenziale $u = 0$, si consideri la rete residua. In base al teorema 8.13 o esiste un cammino $P : s \to t$ di lunghezza residua nulla, e in questo caso P viene detto *cammino aumentante*, in quanto si può aumentare il flusso di una quantità positiva sfruttando P e quindi x non è ottimo, oppure esiste un taglio $Q : s \downarrow t$ di capacità residua nulla. Quindi il flusso sugli archi del taglio (per definizione di capacità residua) deve essere $x_e = c_e^+$ se $e \in Q^+$, $x_e = c_e^-$ se $e \in Q^-$. Quindi il flusso che attraversa il taglio è esattamente uguale alla capacità dello stesso taglio e, siccome nessun flusso può essere superiore a nessuna capacità di taglio, tale flusso è necessariamente ottimo. Abbiamo quindi dimostrato il celebre *teorema di Ford-Fulkerson*.

8.16 Teorema. *Il massimo flusso è uguale alla minima capacità di taglio.* ∎

La dimostrazione stessa del teorema di Ford-Fulkerson suggerisce un algoritmo per la risoluzione del problema (algoritmo di Ford-Fulkerson). Si tratterebbe cioè di scoprire se esiste un cammino aumentante nella rete residua e, se esiste, aumentare il flusso lungo il cammino finché non si raggiunge l'ottimo. Tuttavia, in questa forma "ingenua", l'algoritmo può non funzionare o funzionare male. È stato dimostrato dagli stessi Ford a Fulkerson che, in presenza di valori di capacità irrazionali, l'algoritmo può non terminare. Naturalmente, essendo il flusso limitato superiormente dalla minima capacità di taglio ed aumentando il flusso di una quantità positiva ad ogni iterazione, il valore del flusso tende ad un limite. Tuttavia, alquanto sorprendentemente, tale limite può essere inferiore alla minima capacità di taglio! Queste considerazioni sono ovviamente teoriche, dato che ogni algoritmo implementato su calcolatore non può che far uso di dati razionali, ovvero, senza perdita di generalità, di dati interi (tramite un riscalamento di tutti i valori). Quindi, supponendo i dati interi, ad ogni iterazione il flusso aumenta almeno di uno e l'algoritmo deve pertanto terminare in un numero di iterazioni al più il valore stesso del massimo flusso. Lo svantaggio di questo risultato è che un tale algoritmo non è polinomiale. Dipendendo dai valori dei dati è solamente pseudopolinomiale.

Il semplice accorgimento che permette di ovviare agli inconvenienti appena esposti consiste nel cercare cammini aumentanti con il minimo numero di archi (Edmonds e Karp [1972]). In questo modo si ottiene un algoritmo polinomiale di complessità $O(n^2\, m)$ oppure $O(n\, m \log n)$ con opportune (e complicate) strutture dati. Questo valore di complessità viene ulteriormente abbassato nell'approccio di Dinič [1970] e successivamente di Karzanov [1974].

Dato un flusso ammissibile si costruisce un grafo ausiliario a livelli RP con la seguente proprietà: un nodo j si trova al livello k se il cammino più breve da s a j a capacità residua positiva (nella rete residua) ha k archi; un arco esiste dal nodo i al livello k al nodo j al livello $k+1$ se e solo se esiste l'arco (i, j) (o (j, i)) nella rete ariginale e la capacità residua

Algoritmo di Dinič-Karzanov
input(G, c^-, c^+); $(* \; c^- \leq \mathbf{0} \leq c^+ \; *)$
  $x_e := 0$, $\forall e \in E$;
  repeat
    if $\exists$ rete a livelli $RP : s \to t$
    then begin
      trova flusso bloccante ξ su RP;
      $x_e := x_e + \xi_e$, $e \in P^+$; $x_e := x_e - \xi_e$, $e \in P^-$;
    end;
  until $\exists$ taglio ottimo;
output(x, Q).

da i a j è positiva. Inoltre si definisce *flusso bloccante* sulla rete a livelli un flusso in cui tutti i cammini $s \to t$ con archi da un livello a quello successivo hanno capacità residua nulla.

Successivamente si trova un flusso bloccante su RP. Questa routine opera nel seguente modo: per ogni nodo i sia γ_i^+ (γ_i^-) la somma delle capacità residue degli archi di RP uscenti da (entranti in) i. La capacità del nodo $i \in RP$ è definita come

$$\gamma_i := \min\{\gamma_i^+ \; ; \; \gamma_i^-\}$$

Poi si seleziona un nodo $\bar{k}$ di minima capacità $\bar{\gamma}$ e si spinge una quantità di flusso $\bar{\gamma}$ da $\bar{k}$ al pozzo e dalla sorgente a $\bar{k}$. La prima operazione ("push") viene eseguita inviando tutto il flusso in arrivo ad un nodo verso i nodi adiacenti del livello successivo con l'accortezza di inviare il flusso su un arco solo dopo aver saturato l'arco precedente (operazione di saturazione); in questo modo al più un arco viene riempito solo parzialmente (operazione di non saturazione). L'operazione può essere eseguita senza eccedere la capacità residua di nessun arco poiché $\bar{k}$ è il nodo di minima capacità. La seconda operazione ("pull") è del tutto simile e viene eseguita all'indietro da $\bar{k}$ verso la sorgente.

Alla fine di queste operazioni sono rimossi da RP tutti gli archi che sono stati oggetto di un'operazione di saturazione e vengono aggiornate le capacità residue degli archi oggetto di un'operazione di non saturazione. Vengono aggiornate in modo corrispondente le capacità di nodo e gli archi con capacità nulla sono rimossi da RP insieme con gli archi incidenti ad essi. Questo processo di aggiornare le capacità residue e di rimuovere nodi e archi ad un certo punto termina o perché RP è diventato vuoto o perché i nodi rimanenti hanno tutti capacità strettamente positive.

Nel primo caso la routine termina e il flusso inviato su RP è proprio il flusso bloccante che si cercava. Nel secondo caso si tratta di eseguire di nuovo le operazioni di "push-pull" a partire dal valore corrente di flusso.

La routine deve terminare perché il numero totale di operazioni di saturazione è limitato superiormente da m ed il numero di operazioni di non saturazione è limitato superiormente da n all'interno di ogni operazione "push-pull"; poiché almeno un nodo (quello di minima capacità) viene rimosso in ogni operazione "push-pull" il numero di tali operazioni è limitato da n. Di conseguenza il numero totale di operazioni di non saturazione è limitato da n^2. Questo dimostra che la complessità della routine che trova il flusso bloccante su RP è $O(n^2)$. (vi sono algoritmi più complicati in grado di trovare un flusso bloccante con complessità $O(m \log n)$ che per reti sparse risulta migliore).

L'osservazione cruciale nel derivare la complessità dell'algoritmo complessivo del massimo flusso è che, ad ogni iterazione in cui si costruisce la rete a livelli, il numero dei livelli deve crescere. Infatti l'aver aumentato il flusso di una quantità pari al flusso bloccante impedisce

che vi siano ancora cammini con lo stesso numero di archi della precedente rete a livelli. Siccome il numero di livelli non può superare n, la routine non può essere chiamata più di n volte e la complessità globale è quindi $O(n^3)$.

8.17 Esempio. Si consideri la rete in figura 8.6 con i seguenti valori simmetrici di capacità:

$$c_{1,2}=18,\quad c_{1,3}=15,\quad c_{1,4}=17,\quad c_{2,3}=3,\quad c_{2,4}=1,\quad c_{2,5}=10,\quad c_{2,7}=1,$$
$$c_{3,4}=6,\quad c_{3,6}=10,\quad c_{3,8}=1,\quad c_{4,5}=10,\quad c_{4,6}=9,\quad c_{4,7}=10,\quad c_{4,9}=14,\quad c_{4,8}=3,$$
$$c_{5,7}=8,\quad c_{7,8}=3,\quad c_{7,9}=19,\quad c_{8,9}=15,\quad c_{8,6}=9$$

La prima rete a livelli è rappresentata in figura 8.7. Solo gli archi che appartengono ad un cammino sorgente-pozzo di due archi sono evidenziati. Il nodo di minima capacità è il nodo 4 con capacità 14. L'operazione di push-pull porta quindi il flusso a 14, che risulta ovviamente bloccante. Si procede quindi a costruire una nuova rete a livelli residua, che dovrà necessariamente avere un numero maggiore di livelli. La nuova rete è rappresentata in figura 8.8. Il nodo 2 è di minima capacità con valore 1. Successivamente la rete a livelli diventa quella in figura 8.9. Il nodo di minima capacità è il nodo 3 con valore 1. Si può ancora spedire del flusso sulla rete a livelli residua (figura 8.10). Il nodo di minima capacità è il nodo 1 con valore 3. Il flusso così ottenuto è bloccante.

La nuova rete a quattro livelli è in figura 8.11. Le operazioni di push-pull sono in questo caso quattro e interessano i nodi di minima capacità 4, 3 e 5 con valori 7, 8 e 8 rispettivamente. Le reti a livelli dopo le prime due operazioni di push-pull sono indicate nelle figure 8.12 e 8.13.

Si trova quindi una nuova rete con cinque livelli (figura 8.14), sulla quale si operano 2 operazioni push-pull sui nodi 6 e 2 con valori di flusso 1 e 2 rispettivamente. L'ultima rete a sei livelli (di fatto un cammino) è in figura 8.15. Si esegue un'unica operazione di push-pull dal nodo 3 con valore 1.

La successiva costruzione della rete a livelli porta ad identificare un taglio di minima capacità, indicato in figura 8.16, che quindi prova l'ottimalità del flusso trovato che vale:

$$x_{1,2}=14,\quad x_{1,3}=15,\quad x_{1,4}=17,\quad x_{2,3}=2,\quad x_{2,4}=1,\quad x_{2,5}=10,\quad x_{2,7}=1,$$
$$x_{3,4}=6,\quad x_{3,6}=10,\quad x_{4,5}=-2,\quad x_{4,7}=10,\quad x_{4,9}=14,\quad x_{4,8}=3,\quad x_{4,6}=-1,$$
$$x_{5,7}=8,\quad x_{7,8}=0,\quad x_{7,9}=19,\quad x_{3,8}=1,\quad x_{8,9}=13,\quad x_{8,6}=-9$$

∎

Si può considerare anche una versione più generale del problema del massimo flusso in cui il vincolo sulle capacità viene rilassato a $c_e^- \le c_e^+$ senza imporre nulla al segno di c_e^- e c_e^+. In questo caso si presenta il problema di trovare un flusso iniziale ammissibile. Dopodiché il massimo flusso può venire calcolato come nel caso già considerato.

In generale il calcolo di un flusso ammissibile rispetto ad assegnati vincoli di capacità, e quindi anche la verifica della condizione di ottimalità stabilita nel corollario 8.8, possono essere eseguiti risolvendo un altro problema di massimo flusso su una rete derivata da quella originale.

Dapprima si assegna ad ogni arco un flusso $\bar{x}$ ammissibile rispetto agli intervalli di capacità. Se tale flusso non soddisfa il vincolo di divergenza $\bar{y} = A\,\bar{x} = b$, si passa a risolvere un MFP su una rete estesa $\hat{G}$ ottenuta aggiungendo due nodi s e t e i seguenti archi:

(s,i) se $\bar{y}_i < b_i$, con intervallo di capacità $[0, b_i - \bar{y}_i]$

(i,t) se $\bar{y}_i > b_i$, con intervallo di capacità $[0, \bar{y}_i - b_i]$

Agli archi originali vengono assegnati gli intervalli di capacità $[c_e^- - \bar{x}_e, c_e^+ - \bar{x}_e]$. Sia $\Delta := \sum_{i:\bar{y}_i > b_i} \bar{y}_i - b_i$.

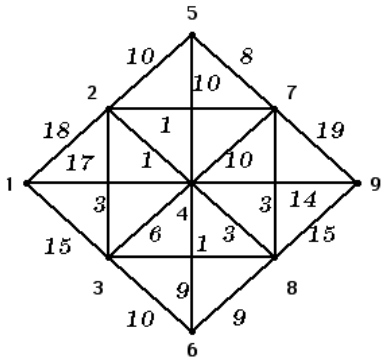


FIGURA 8.6

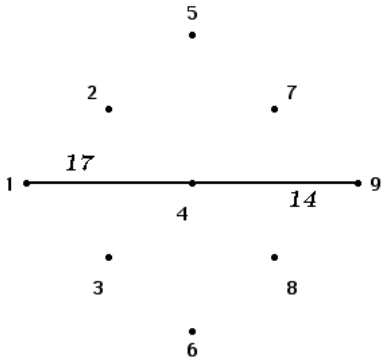


FIGURA 8.7

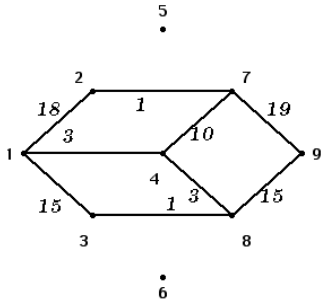


FIGURA 8.8

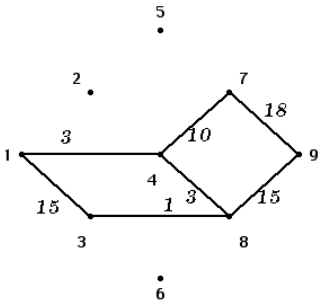


FIGURA 8.9

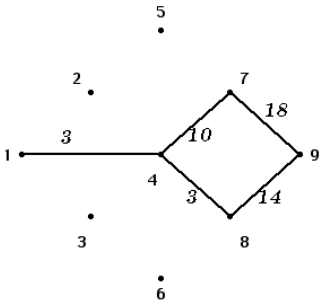


FIGURA 8.10

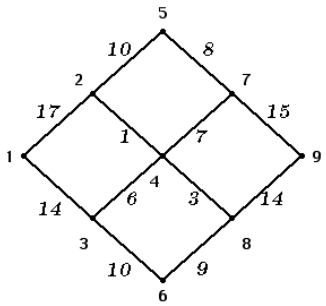


FIGURA 8.11

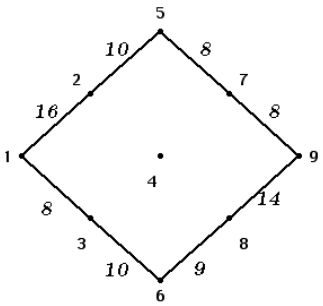


FIGURA 8.12

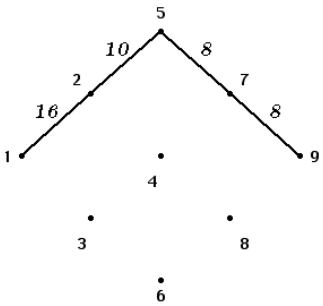


FIGURA 8.13

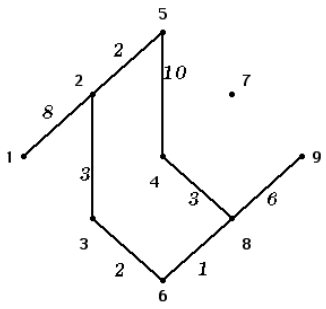


FIGURA 8.14

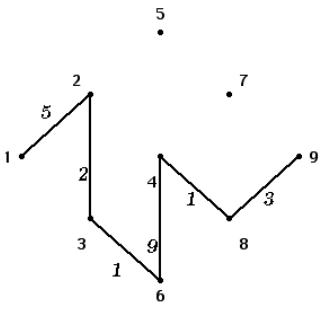


FIGURA 8.15

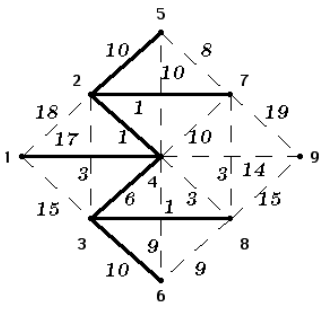


FIGURA 8.16

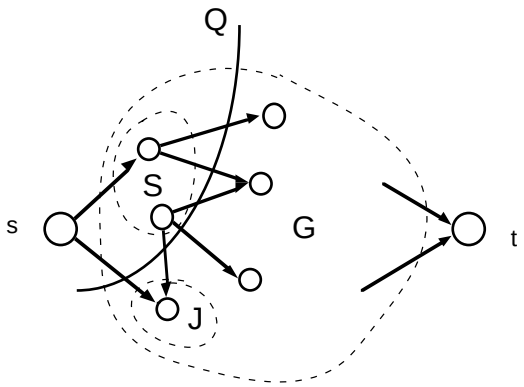


FIGURA 8.17

8.18 COROLLARIO. *Sia $\hat{x}$ la soluzione ottima del MFP su $\hat{G}$ con valore di massimo flusso $\hat{\varphi}$. G ammette un flusso ammissibile se e solo se $\hat{\varphi} = \Delta$.*

DIMOSTRAZIONE. Si noti che $A\hat{x} = 0$ su $\hat{G}$. Se $\hat{\varphi} = \Delta$, allora tutti gli archi (s, i) sono saturi e quindi $A\hat{x} = b - \bar{y}$ su G, da cui $\bar{x} + \hat{x}$ è ammissibile rispetto ai vincoli di divergenza e di capacità delle rete originaria. Viceversa, se esiste un flusso ammissibile $\tilde{x}$ in G allora $\tilde{x} - \bar{x}$ è ammissibile in $\hat{G}$ con tutti gli archi (s, i) saturi, e quindi $\hat{x}$ è l'ottimo del MFP in $\hat{G}$. ∎

Dimostriamo ora che, nel caso non esista flusso ammissibile in G, il MFP fornisce un taglio $Q(S)$ in G di minima capacità che funge da 'sbarramento' per il flusso $b(S)$, cioè si ha $b(S) > c(Q(S))$.

8.19 COROLLARIO. *La condizione $b(S) \leq c(Q(S))$, $\forall S \subset N$, è necessaria e sufficiente all'esistenza di un flusso ammissibile.*

DIMOSTRAZIONE. La necessità della condizione è ovvia. Si supponga che non esista flusso ammissibile in G, cioè $\hat{\varphi} < \Delta$. Il taglio di minima capacità è necessariamente generato da un insieme $\hat{S}$ di nodi che deve contenere anche altri nodi oltre ad s (si faccia riferimento alla figura 8.17). Sia $\hat{S} = S \cup \{s\}$. Si indichi con $Q(\hat{S})$ il taglio generato da $\hat{S}$ in $\hat{G}$ e con $Q(S)$ il taglio generato da S in G. Gli archi di $\hat{G}$ che sono in $Q(\hat{S})$ ma non in $Q(S)$ sono tutti del tipo (s, i) con $i \notin S$. Si definisca $J = \{i : (s, i) \in \hat{G} \wedge i \notin S\}$. Per semplicità si supponga che tutti gli archi del taglio siano orientati nel senso del taglio. La capacità $\hat{c}(Q(\hat{S}))$ (in $\hat{G}$) è quindi data da:

$$\hat{c}(Q(\hat{S})) = \sum_{e \in Q(S)} (c_e^+ - \bar{x}_e) + \sum_{i \in J} (b_i - \bar{y}_i) = c(Q(S)) - \sum_{e \in Q(S)} \bar{x}_e + \sum_{i \in J} (b_i - \bar{y}_i)$$

dove $c(Q(S))$ è la capacità di $Q(S)$ in G. Per definizione $\sum_{i \in S} \bar{y}_i = \sum_{e \in Q(S)} \bar{x}_e$ (abbiamo supposto solo archi uscenti dal taglio). Quindi

$$\Delta > \hat{c}(Q(\hat{S})) = c(Q(S)) - \sum_{i \in S} \bar{y}_i + \sum_{i \in J} (b_i - \bar{y}_i) = c(Q(S)) - \sum_{i:(s,i) \in \hat{G}} \bar{y}_i + \sum_{i \in J} b_i =$$

$$c(Q(S)) - \sum_{i:(s,i) \in \hat{G}} \bar{y}_i + \sum_{i \in J} b_i + \sum_{i \in S} b_i - \sum_{i \in S} b_i =$$

$$c(Q(S)) - \sum_{i:(s,i)\in\hat{G}} \bar{y}_i + \sum_{i:(s,i)\in\hat{G}} b_i - \sum_{i\in S} b_i = c(Q(S)) + \Delta - b(S)$$

e quindi $b(S) > c(Q(S))$. ∎

Se la condizione del corollario 8.8 non è soddisfatta allora il vettore d'incidenza h del sottoinsieme $N\backslash S$, cioè $h_i := 0$ se $i \in S$ e $h_i = 1$ se $i \notin S$, con S corrispondente al minimo taglio $Q(S)$, è una direzione di ascesa per la funzione duale nello spazio delle variabili duali u con derivata direzionale uguale a $b(S) - c(Q(S))$.

8.20 Esercizio. Si modelli come un problema di massimo flusso il problema di determinare il massimo numero di cammini disgiunti negli archi fra due nodi assegnati s e t in un grafo non orientato, e si dimostri quindi che tale numero è uguale al minimo numero di archi che sconnettono s da t. ∎

8.6. Cammino minimo

Introdurremo il problema del cammino minimo definendo prima il suo problema duale, cioè il problema della massima tensione, che risulta naturalmente formulato come problema di rete di flusso e che, rispetto al problema del cammino minimo, svolge lo stesso ruolo del problema del massimo flusso rispetto al problema della minima capacità.

Il problema della massima tensione (MTP) è un particolare problema duale in cui l'obiettivo è la massimizzazione della differenza di potenziale fra un determinato nodo t, indicato come *pozzo*, ed un altro determinato nodo s, indicato come *sorgente*. I vincoli a cui è sottoposta la tensione sono soltanto quelli di espansione su tutti gli archi (supponiamo che per ogni intervallo di espansione $[d_e^-, d_e^+]$ sia $d_e^- \le 0$ e $d_e^+ \ge 0$).

Nel problema della massima tensione simmetrico (SMTP) la limitazione della tensione in ogni arco dipende dal valore assoluto della tensione, quindi $-d_e^- = d_e^+ =: d_e$. Nel problema della massima tensione diretto (DMTP) la tensione può espandersi solo nel verso dell'arco e quindi $d_e^- = 0$, $d_e^+ =: d_e$. Si noti che, in base all'ipotesi sul segno delle lunghezze, una soluzione ammissibile, corrispondente alla tensione nulla, è immediatamente disponibile. Si deve quindi risolvere

$$\begin{aligned} \max \quad & u_t - u_s \\ & d_e^- \le v_e \le d_e^+ \qquad \forall e \in E \end{aligned}$$

ovvero

$$\begin{aligned} \max \quad & -\sum_{i\in N} u_i\, b_i - \sum_{e\in E} g_e(v_e) \\ & v = -u\, A \end{aligned}$$

dove

$$b_i = \begin{cases} 1 & \text{se } i = s \\ -1 & \text{se } i = t \\ 0 & \text{altrimenti} \end{cases} \qquad \text{e} \qquad g_e(v_e) = \begin{cases} 0 & \text{se } d_e^- \le v_e \le d_e^+ \\ +\infty & \text{altrimenti} \end{cases}$$

La funzione f_e coniugata di g_e è $f_e(x_e) := \max\{d_e^- x_e; d_e^+ x_e\}$ ($= d_e|x_e|$ per il SMTP) e il flusso deve essere conservato su tutti i nodi tranne la sorgente e il pozzo dove $b_s := 1$ e $b_t := -1$ rispettivamente.

Quindi nel problema primale si minimizza il costo di una unità di flusso che viene trasferita dalla sorgente al pozzo, cioè

$$\begin{aligned} \min \quad & \sum_{e\in E} \max\left\{d_e^- x_e; d_e^+ x_e\right\} \\ & A\, x = b \end{aligned} \tag{8.12}$$

Algoritmo della Massima Tensione
input(G, d^-, d^+); $(* \; d^- \leq \mathbf{0} \leq d^+ \; *)$
  $u_i := 0$, $\forall i \in N$;
  repeat
    if $\exists Q(S) : s \downarrow t$: $\delta(Q) > 0$
    then $u_i := u_i + \delta(Q)$, $i \in N \backslash S$
  until $\exists P : s \to t$: $\delta(P) = 0$
output(u, P).

e il costo di ogni arco è d_e^+ se il flusso nell'arco è positivo, altrimenti il costo è $-d_e^-$. L'unimodularità totale garantisce che, se esiste una soluzione ottima, ne esiste una di valore intero. Inoltre una soluzione generica di un sistema lineare $A\,x = b$ si ottiene sommando una soluzione particolare di $A\,x = b$ con una soluzione generica di $A\,x = 0$. Ogni vettore d'incidenza $e(P)$ di un cammino $P : s \to t$ è una soluzione particolare di $A\,x = b$. Quindi una generica soluzione ammissibile di (8.12) è data da un cammino più una circolazione. Se ora escludiamo dalle soluzioni ammissibili le soluzioni con circolazioni non nulle, abbiamo la garanzia di non perdere l'ottimo dato che i costi sono non negativi.

Allora il problema primale consiste nel determinare un cammino $P : s \to t$, la cui lunghezza $d^+(P)$, definita come in (8.1), sia minima e viene pertanto definito come *problema del cammino minimo*.

Come nel problema del massimo flusso si è potuta stabilire una equivalenza fra un problema combinatorio, quale quello del cammino minimo, e un problema continuo, quale quello della massima tensione, grazie alla totale unimodularità della matrice dei vincoli.

La dualità forte può essere inoltre dimostrata per altra via tramite il seguente ragionamento. Dati un potenziale u ammissibile rispetto agli intervalli di espansione (e peraltro arbitrario) e il flusso $x = 0$, si consideri la rete residua. In base al teorema 8.13 o esiste un taglio $Q : s \downarrow t$ di capacità residua nulla ed espansione residua positiva e in questo caso Q viene detto *taglio aumentante*, in quanto si può aumentare il potenziale in t di una quantità positiva sfruttando Q e quindi x non è ottimo, oppure esiste un cammino $P : s \to t$ di lunghezza residua nulla. Quindi la tensione sugli archi del cammino (per definizione di lunghezza residua) è $v_e = d_e^+$, $\forall e \in P^+$, e $v_e = d_e^-$, $\forall e \in P^-$. Quindi, sommando le tensioni lungo il cammino si ha

$$u_t - u_s = \sum_{e \in P^+} v_e - \sum_{e \in P^-} v_e = \sum_{e \in P^+} d_e^+ - \sum_{e \in P^-} d_e^- = d(P)$$

Per ogni altro cammino P' che unisce s e t si ha $d_e^- \leq v_e \leq d_e^+$, $\forall e \in P'$, e quindi

$$d(P') = \sum_{e \in P'^+} d_e^+ - \sum_{e \in P'^-} d_e^- \geq \sum_{e \in P'^+} v_e - \sum_{e \in P'^-} v_e = u_t - u_s = d(P)$$

dimostrando l'ottimalità del cammino P. Si è quindi dimostrato un teorema che è l'analogo del teorema di Ford-Fulkerson nel caso del cammino minimo.

8.21 Teorema. *Il cammino minimo è uguale alla massima tensione.* ∎

Anche in questo caso la dimostrazione stessa del teorema suggerisce un algoritmo per la risoluzione del problema della massima tensione. Si tratterebbe cioè di scoprire se esiste un taglio aumentante nella rete residua e, se esiste, aumentare il potenziale secondo il taglio

Algoritmo di Dijkstra
input(G, d^-, d^+); $(* \; d^- \leq \mathbf{0} \leq d^+ \; *)$
$d_{ij} = d_e^+$, $d_{ji} = -d_e^-$, $\forall e = (i,j) \in E$; $d_{ij} = +\infty$, $\forall (i,j) \notin E$;
$\rho_i := \infty$, $i \in N$, $i \neq s$; $\rho_s := 0$;
$S := \{s\}$; $k := s$; $previous(s) = s$;
repeat
for $j \notin S$ **do**
begin
if $\rho_j > \rho_k + d_{kj}$
then begin
$\rho_j := \rho_k + d_{kj}$;
$previous(j) = k$
end
end;
$k := \text{argmin}_{j \notin S} \rho_j$;
$S := S \cup \{k\}$;
until $t \in S \vee \rho_k = \infty$
output(ρ,*previous*)

finché non si raggiunge l'ottimo. A differenza del problema del massimo flusso, anche in questa forma semplice l'algoritmo è corretto.

Il nucleo dell'algoritmo è costituito dalla routine che aumenta i potenziali sul sottoinsieme complementare $N \backslash S$. Un incremento uniforme Δ del potenziale $N \backslash S$ deve arrestarsi non appena si verifichi $v_e + \Delta = d_e^+$ oppure $v_e - \Delta = d_e^-$ su qualche arco del taglio, la cui lunghezza residua diventerà nulla nell'iterazione successiva. Quindi la ricerca del taglio successivo viene effettuata a partire dall'albero dei cammini residui a lunghezza nulla che viene via via formato. Tale albero può essere semplicemente aggiornato aggiungendo l'arco (o gli archi) appena resi a lunghezza nulla.

Sia $S' \supset S$ l'insieme di nodi corrispondente al taglio trovato nell'iterazione successiva e sia $S'' := S' \backslash S$. Per trovare Δ dobbiamo calcolare

$$\Delta = \min\{ \min_{e \in Q^+(S)} d_e^+ - v_e \, ; \, \min_{e \in Q^-(S)} -d_e^- + v_e \}$$

La notazione può essere semplificata scrivendo $d_{ij} := d_e^+$ se $i \in S$, $j \notin S$ e $d_{ji} := -d_e^-$ se $i \notin S$, $j \in S$ (la notazione non è ambigua poiché possiamo assumere senza perdita di generalità che esiste al più un arco per ogni coppia di nodi: infatti se vi sono più archi paralleli per una coppia di nodi, tutti gli archi possono essere sostituiti da un nuovo arco e' con $d_{e'}^- := \max_e d_e^-$ e $d_{e'}^+ := \min_e d_e^+$). Allora si ha

$$\Delta = \min_{\substack{i \in S \\ j \notin S}} \{ \, d_{ij} - u_j + u_i \, \}$$

Si noti che, in base all'algoritmo, $u_j = \bar{u}$, $\forall j \notin S$, per un certo valore costante $\bar{u}$. Il valore successivo $\bar{u}'$ viene ovviamente calcolato come $\bar{u}' := \bar{u} + \Delta$, cioè

$$\bar{u}' := \min_{j \notin S} \min_{i \in S} d_{ij} + u_i = \min_{j \notin S} \rho_j =: \bar{\rho} \tag{8.13}$$

dove $\rho_j := \min_{i \in S} d_{ij} + u_i$. Quindi si ha $u_i' = u_i$ se $i \in S$ e $u_i' = \bar{\rho}$ se $i \notin S$. I valori ρ_j, per $j \notin S'$, possono essere efficientemente aggiornati secondo

$$\rho'_j := \min_{i\in S'} d_{ij} + u'_i = \min_{i\in S\cup S''} d_{ij} + u'_i =$$

$$= \min\{\min_{i\in S} d_{ij} + u_i \,;\, \min_{i\in S''} d_{ij} + u'_i\} = \min\{\rho_j \,;\, \bar\rho + \min_{i\in S''} d_{ij}\} = \min\{\rho_j \,;\, \bar\rho + d_{kj}\}$$

con $d_{kj} := \min_{i\in S''} d_{ij}$. Inoltre, ogni volta che $\rho'_j = \bar\rho + d_{kj} < \rho_j$, si crea un puntatore da j a k (assegnando *previous*$(j) := k$) per poter ricostruire il cammino minimo una volta raggiunto il pozzo.

Complessivamente l'algoritmo richiede un tempo $O(n^2)$. In fatti il ciclo 'repeat' può essere eseguito al più n volte, poiché S'' è costituito da almeno un nodo. Inoltre l'aumento di tensione su $N\backslash S$ richiede il calcolo di $\bar\rho$ ((8.13)) che può essere effettuato semplicemente in tempo $O(n)$. L'aggiornamento di ρ_j è eseguito in totale (su tutti i cicli 'repeat') non più di m volte. Da queste considerazioni discende la complessità di $O(n^2)$. Questo valore non può essere abbassato per grafi densi (cioè quando $m = \Omega(n^2)$) perché la sola lettura dei dati richiede nel caso peggiore un tempo $O(n^2)$.

Per grafi sparsi (cioè $m = O(n)$) è più conveniente usare una struttura a 'heap' per i valori ρ_j. In questo modo il calcolo di $\bar\rho$ richiede tempo costante. Tuttavia bisogna aggiornare lo 'heap' ad ogni aggiornamento di ρ_j e ad ogni rimozione della radice dello 'heap'. Quindi discende una complessità globale $O(m \log n)$. Questo algoritmo è dovuto a Dijkstra [1959]. Per un'alternativa derivazione dell'algoritmo di Dijkstra si veda anche la sezione 9.3.

Si noti che il valore finale di ρ_j corrisponde alla minima distanza dalla sorgente al nodo j, per tutti i nodi appartenenti all'insieme S e che l'insieme di archi $\{(i,j) : j \in S,\, j \neq s,\, i = \textit{previous}(j)$ costituisce l'albero dei cammini minimi dalla sorgente a tutti i nodi in S. L'algoritmo può tuttavia essere continuato fino a calcolare le minime distanze dalla sorgente a tutti i nodi, cioè finché $S = N$.

8.22 Esempio. Si consideri il grafo in figura 8.18 dove i numeri accanto agli archi rappresentano le distanze. Si tratta di risolvere un problema simmetrico. In figura 8.19 si rappresenta l'albero dei cammini minimi in S ad ogni iterazione dell'algoritmo. ∎

8.23 Esercizio. Si consideri un problema di cammino minimo da un nodo s ad un nodo t in cui il costo dell'arco (i,j) si esprime come $a_i + b_j$ (quindi vi sono un costo d'entrata nel nodo ed un costo d'uscita). Si dimostri che tale problema è equivalente ad un problema in cui il costo dell'arco (i,j) si esprime come c_j (quindi vi è solo un costo d'entrata nel nodo). Si dimostri che la complessità computazionale dell'algoritmo di Dijkstra si riduce a $O(m + n \log n)$ per questo problema (anche per grafi densi). ∎

Si può considerare anche una versione più generale del problema del cammino minimo in cui il vincolo sulle lunghezze viene rilassato a $d_e^- \leq d_e^+$ senza imporre nulla al segno di d_e^- e d_e^+. In questo caso si presenta il problema di trovare un potenziale iniziale ammissibile. L'algoritmo di Dijkstra si basa sull'ipotesi di lunghezze non negative e non è applicabile se l'ipotesi non è soddisfatta.

Se alcune lunghezze sono negative si può sempre applicare l'algoritmo generale, dopo aver calcolato un potenziale ammissibile. A sua volta il calcolo in generale di un potenziale ammissibile rispetto ad assegnati intervalli di espansione $[d_e^-, d_e^+]$, e quindi la verifica delle condizioni primali di ottimalità espresse nel corollario 8.8 si può effettuare tramite un SPP.

Assegnato un potenziale u, per ogni nodo i si definisca

$$\sigma_i(e) := \begin{cases} \max\{0\,;\, d_e^- - v_e\} & \text{se } e \in E_i^+ \\ \max\{0\,;\, v_e - d_e^+\} & \text{se } e \in E_i^- \end{cases} \qquad \text{e} \qquad \sigma_i := \max_{e\in E_i} \sigma_i(e)$$

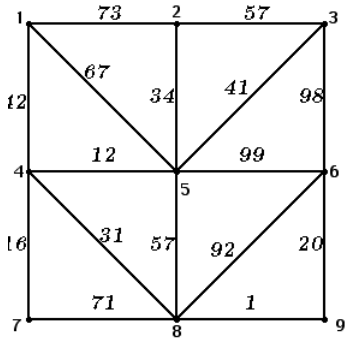


FIGURA 8.18

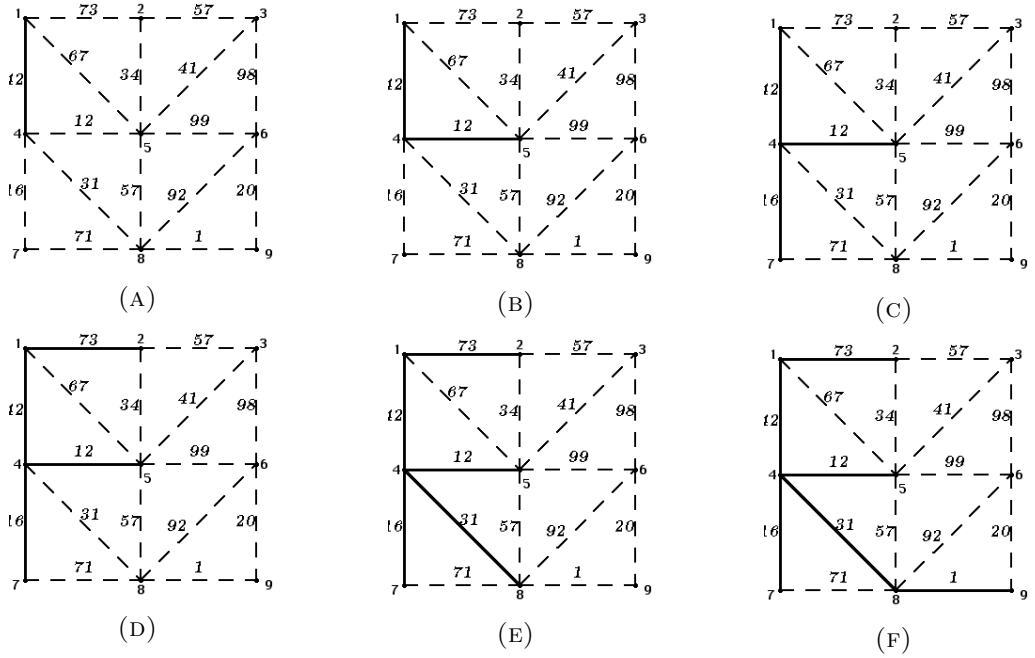


FIGURA 8.19

Se $\sigma_i = 0$ diciamo che il nodo i è bilanciato. Se tutti i nodi sono bilanciati allora ovviamente u è ammissibile. Se invece esiste un nodo s con $\sigma_s > 0$ si passa a risolvere un DSPP in RG da s verso tutti i nodi adiacenti ad s. A questo scopo sia σ_{sj} il massimo su tutti i $\sigma_s(e)$ relativi ad archi e incidenti sia in s che in j. Se $\rho_j \geq \sigma_{sj}$ per tutti gli j adiacenti a s allora il potenziale calcolato in questo modo viene aggiunto al potenziale corrente aggiornandone il valore in modo tale che il nodo s diventa bilanciato. Se invece $\rho_j < \sigma_{sj}$ per qualche nodo adiacente j, allora il problema non è ammissibile.

Infatti sia P il cammino minimo da s a j e sia C il circuito ottenuto aggiungendo a P l'arco $e' = (s, j) \in C^-$ (il caso $e' = (j, s)$ è simile). Allora si ha:

$$0 > \rho_j - \sigma_{sj} = \sum_{e\in P^+} \delta_e^+ + \sum_{e\in P^-} \delta_e^- - d_{e'}^- + v_{e'} \geq \sum_{e\in C^+} d_e^+ - v_e + \sum_{e\in C^-} v_e - d_e^- =$$
$$= \sum_{e\in C^+} d_e^+ - \sum_{e\in C^-} d_e^- - \Big(\sum_{e\in C^+} v_e + \sum_{e\in C^-} -v_e \Big) = \sum_{e\in C^+} d_e^+ - \sum_{e\in C^-} d_e^- = d(C)$$

e ovviamente non si può trovare nessuna soluzione in quanto i vincoli di espansione sul circuito C sono troppo 'stretti' da poter produrre una tensione ammissibile. Vale inoltre

8.24 Corollario. *La condizione $d(C) \geq 0$, $\forall C$, è necessaria e sufficiente all'esistenza di un potenziale ammissibile.*

Dimostrazione. La necessità è ovvia. Per ciò che riguarda la sufficienza si noti che $d(C) \geq 0$ implica $\rho_j \geq \sigma_{sj}$, come si può vedere dalle precedenti disuguaglianze, e siccome $\rho_j \leq \sigma_{sj}$ si ha $\rho_j = \sigma_{sj}$. ∎

Se la condizione 8.8 non è soddisfatta allora il vettore d'incidenza del circuito C è una direzione di discesa per la funzione obiettivo primale nello spazio dei flussi con derivata direzionale pari a $d^+(C)$.

8.7. L'algoritmo 'out-of-kilter'

In questa sezione si studierà un particolare algoritmo per risolvere il problema primale nella sua generalità. Vedremo che tale algoritmo ha come componenti a più basso livello i problemi del massimo flusso e del cammino minimo. Sebbene l'algoritmo proposto sia soltanto pseudopolinomiale, tuttavia ha una particolare semplicità ed eleganza di struttura ed è effettivamente il punto di partenza per algoritmi più complessi.

L'idea base dell'algoritmo 'out-of-kilter' deriva dal fatto che la curva caratteristica Γ_e è a scalini, dalla condizione di ottimalità $(x_e, v_e) \in \Gamma_e$, $\forall e$, e dal teorema 8.13. Dati dei valori iniziali (x_e, v_e), si cerca di 'spingerli' verso Γ_e, o variando il flusso lungo un circuito in modo costante, cioè mantenendo inalterato il vincolo di divergenza, oppure variando la tensione su un taglio incrementando uniformemente il potenziale su uno degli insiemi di nodi che definiscono il taglio.

Un arco e viene detto *in-kilter* se $(x_e, v_e) \in \Gamma_e$; altrimenti viene detto *out-of-kilter*. Si consideri un arco (i, j) out-of-kilter. Se (x_e, v_e) è situato a destra e sotto Γ_e allora si ponga $s := i$ e $t := j$. Altrimenti (se cioè sta sopra e a sinistra di Γ_e) si ponga $s := j$ e $t := i$.

In base al lemma di Minty (teorema 8.13), o esiste un cammino $P : s \to t$ a capacità residua positiva oppure esiste un taglio $Q : s \downarrow t$ a espansione residua positiva. Nel primo caso è possibile spedire del flusso positivo da s a t lungo P e da t a s lungo l'arco (i, j). Così facendo il vincolo di divergenza non viene violato in quanto si aggiunge alla soluzione precedente una circolazione e i punti (x, v) su tutti gli archi del circuito si avvicinano di una quantità positiva alle rispettive curve caratteristiche. Inoltre il teorema 8.14 garantisce una diminuzione positiva della funzione primale. Nel secondo caso è possibile aumentare in modo uniforme di una quantità positiva il potenziale sui nodi "dall'altra parte" del taglio così da variare la tensione sugli archi del taglio avvicinando i punti (x, v) alle rispettive curve caratteristiche. Inoltre il teorema 8.15 garantisce un aumento positivo della funzione duale.

Le due operazioni sopra descritte possono essere rese più efficaci se, nel primo caso, anziché spedire del flusso solo lungo il cammino P si calcola il flusso massimo da s a t (con l'avvertenza di bloccare il calcolo non appena il flusso supera il valore di capacità

```
Algoritmo out-of-kilter
input(G, Γ);
    u_i := 0, ∀i ∈ N; sia x : A x = b;
    while ∃ e = (h, k) : (x_e, v_e) ∉ Γ_e do
        if δ_e^+ > 0
        then (s, t) := (h, k)  else (s, t) := (k, h) ;
        while (x_e, v_e) ∉ Γ_e do
        begin
            calcola cammino minimo s → t (∗ ρ_j come nell'Algoritmo di Dijkstra∗);
            if ρ_t = +∞ then output("primale non ammissibile");
            u_i := u_i + min {ρ_i; ρ_t}, ∀i;
            if (x_e, v_e) ∉ Γ_e
            then begin
                calcola massimo flusso ξ : s → t → s;
                if ξ = +∞ then output("duale non ammissibile");
                x := x + ξ;
            end
        end
    end
output(x, u).
```

residua nell'arco out-of-kilter in questione), e nel secondo caso si calcola il cammino minimo $s \to t$ rispetto alle lunghezze residue e si aggiornano i potenziali aggiungendo i valori di distanza minima.

Iterando in questo modo possono succedere tre fatti alternativi: l'arco va in kilter, il problema del massimo flusso è illimitato, il problema del cammino minimo è inammissibile. Nel primo caso si passa a considerare un successivo arco non in kilter e se non ce ne sono si è evidentemente raggiunta l'ottimalità. Nel secondo caso si riesce a far circolare una quantità illimitata di flusso senza raggiungere la curva caratteristica. Questo fatto sta ad indicare che il problema duale è non ammissibile. Nel terzo caso si può far aumentare il potenziale su alcuni nodi di una quantità arbitraria e la tensione sugli archi del taglio generato da questi nodi senza raggiungere la curva caratteristica. Questo fatto sta ad indicare che il problema primale è non ammissibile.

Conviene spendere qualche parola di più sul secondo e sul terzo caso. Sembrerebbe che la possibilità di aumentare in modo illimitato il flusso corrisponda ad una situazione di primale illimitato. In effetti è proprio questo il significato di un aumento indiscriminato di flusso in base al teorema 8.14, ma solo limitatamente ad un sottoinsieme di archi. Può avvenire che altri archi della rete non ammettano flusso ammissibile. Altrettanto vale per il terzo caso. Come esempio si consideri una rete di 4 nodi. Fra il nodo 1 e il nodo 2 ci siano due archi paralleli e_1 ed e_2 con intervalli di espansione rispettivamente $[1, 2]$ e $[-2, -1]$. Poi vi siano gli archi in serie $e_2 = (2, 3)$ e $e_4 = (3, 4)$ con intervalli di capacità rispettivamente $[1, 2]$ e $[-2, -1]$. Supponiamo vi sia conservazione del flusso nei nodi. Ovviamente non vi può essere tensione ammissibile sia su e_1 che su e_2 e analogamente non vi può essere flusso ammissibile sia su e_3 che su e_4. Quindi primale e duale sono contemporaneamente non ammissibili. Tuttavia l'algoritmo out-of-kilter troverebbe un flusso illimitato se applicato all'arco e_1 (sul circuito formato dai due archi paralleli) e troverebbe un potenziale illimitato se applicato all'arco e_4. L'algoritmo quindi scopre certamente la non ammissibilità di un problema. Però per verificare se l'altro problema è illimitato o non ammissibile bisogna, a questo punto, risolvere un problema di ammissibilità.

L'algoritmo out-of-kilter non ha complessità polinomiale. Se i dati sono interi (o razionali) è garantito soltanto un avvicinamento alla curva caratteristica di almeno un'unità per almeno un arco. Quindi l'algoritmo termina in un numero finito di passi. Tuttavia il numero di passi potrebbe essere nel caso peggiore proporzionale ai valori di capacità e/o espansione e quindi solamente pseudopolinomiale.

8.25 Esempio. Sia data la rete in figura 8.20 con le curve caratteristiche indicate e sia $b = (8, 3, -5, -6)$. Come soluzione iniziale conviene usare un potenziale nullo ed un qualsiasi valore di flusso che sia ammissibile per le divergenze (ma non necessariamente per le capacità!). Sia la soluzione iniziale quella riportata in figura 8.21. In figura 8.22 sono disegnati tratteggiati gli archi out-of-kilter rispetto alla soluzione iniziale.

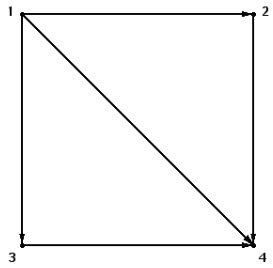
Si inizi l'iterazione con l'arco (1,3). Per i valori assegnati c'è un cammino a lunghezza residua nulla fra 1 e 3. Quindi eseguiamo un massimo flusso da 1 a 3. Il risultato porta ad un'unità di flusso fatta circolare fra i nodi $1 \to 2 \to 4 \to 3 \to 1$. Con questo aggiornamento del flusso l'arco (1,3) non entra ancora in kilter. Dalla teoria sappiamo che deve esistere un cammino minimo di lunghezza residua positiva. Risolvendo si trova il cammino $1 \to 2 \to 4 \to 3$ di lunghezza 1 con $\rho = \{0, 0, 1, 1\}$ e $w = \rho$, $u := u + w = \{0, 0, 1, 1\}$ e $v = \{0, 1, 1, 0, 1\}$. La soluzione corrente di flusso e potenziale è riportata in figura 8.23. Siccome l'arco non è in kilter eseguiamo ora un massimo flusso da 1 a 3. Il risultato porta a due unità di flusso fatte circolare fra i nodi $1 \to 2 \to 4 \to 3 \to 1$ (si noti che si tratta del cammino minimo appena trovato, perché?). Con questi nuovi valori l'arco (1,2) è entrato in kilter (ma non (1,3)). Dobbiamo eseguire nuovamente un cammino minimo che vale 2 direttamente da 1 a 3, con $\rho = \{0, 1, 2, 1\}$, $w = \rho$, $u := u + w = \{0, 1, 3, 2\}$ e $v = \{1, 3, 2, -1, 1\}$. La soluzione corrente di flusso e potenziale è riportata in figura 8.24.

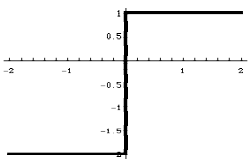
Stavolta (1,3) è entrato in kilter e rimane soltanto l'arco (1,4). Eseguendo un massimo flusso otteniamo un'unità di flusso fatta circolare fra i nodi $1 \to 2 \to 4 \to 1$. Quindi eseguiamo un cammino minimo che risulta essere $1 \to 2 \to 4$ di lunghezza 1, $w = \rho = \{0, 0, 0, 1\}$, $u = \{0, 1, 3, 3\}$ e $v = \{1, 3, 3, 0, 2\}$. Essendo (1,4) ancora out-of-kilter bisogna eseguire un massimo flusso che fa circolare un'unità di flusso fra i nodi $1 \to 2 \to 4 \to 1$ (nuovamente il cammino minimo della precedente iterazione!).

Con quest'ultima iterazione tutti gli archi sono stati messi in kilter e si è raggiunta l'ottimalità. La soluzione ottima di flusso e potenziale è riportata in figura 8.25.

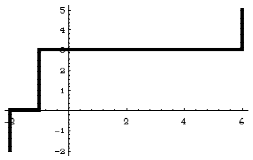
In figura 8.26 è raffigurata l'iterazione per l'arco (1,3). La seguente tabella riporta l'iterazione completa dei valori (x, v) per tutti gli archi. L'asterisco indica che l'arco entra in kilter.

	(1,2)	(1,3)	(1,4)	(3,4)	(2,4)
0	(-3,0)	(5,0)	(6,0)	(0,0)*	(0,0)*
1	(-2,0)	(4,0)	(6,0)	(-1,0)	(1,0)
2	(-2,0)	(4,1)	(6,1)	(-1,0)	(1,1)
3	(0,0)*	(2,1)	(6,1)	(-3,0)	(3,1)
4	(0,1)	(2,3)*	(6,2)	(-3,-1)	(3,1)
5	(1,1)	(2,3)	(5,2)	(-3,-1)	(4,1)
6	(1,1)	(2,3)	(5,3)	(-3,0)	(4,2)
7	(2,1)	(2,3)	(4,3)*	(-3,0)	(5,2)

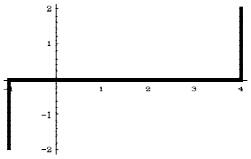




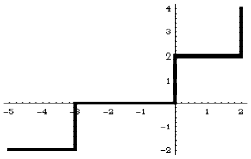
Γ_{12}



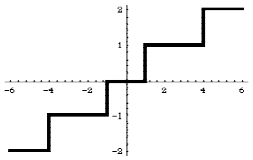
Γ_{13}



Γ_{14}



Γ_{34}



Γ_{24}

FIGURA 8.20

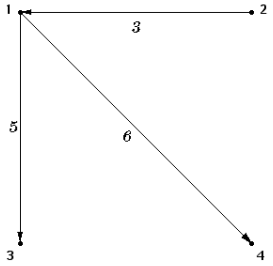


FIGURA 8.21

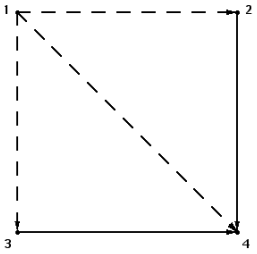


FIGURA 8.22

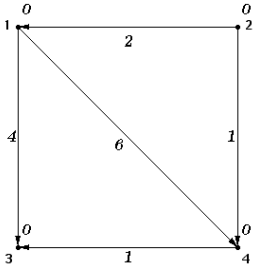


FIGURA 8.23

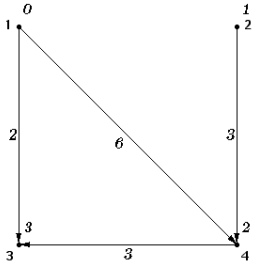


FIGURA 8.24

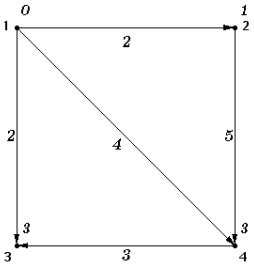


FIGURA 8.25

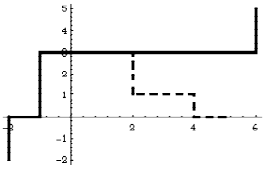


FIGURA 8.26

8.8. Algoritmi a gradazione dei costi

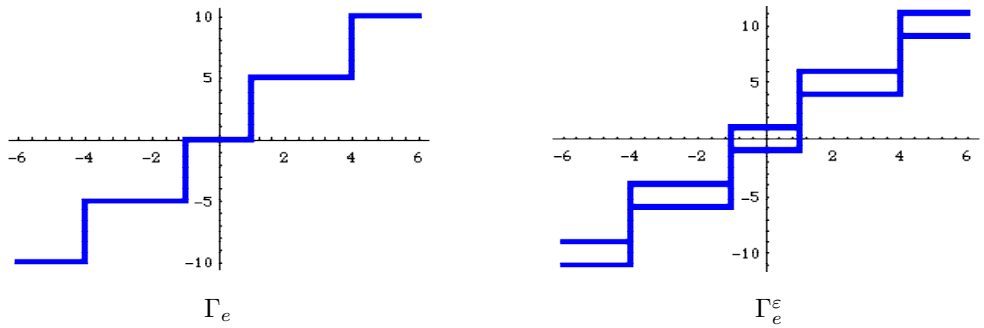
Per ottenere degli algoritmi polinomiali dall'idea generale dell'algoritmo out-of-kilter sono ovviamente necessarie delle opportune modifiche. Si sono rivelate particolarmente efficaci quelle tecniche che si servono di soluzioni ε-ottime, cioè soluzioni che sono ottime o ammissibili a meno di un'approssimazione ε. L'idea di fondo consiste nel diminuire in iterazioni successive il valore di ε fino ad un valore che garantisca l'ottimalità o l'ammissibilità. Se la diminuzione consiste, ad esempio, in un dimezzamento, il numero di queste iterazioni è chiaramente limitato superiormente da un valore che è logaritmico nei dati del problema. Essenziale è quindi la possibilità di trasformare in tempo polinomiale una soluzione 2ε-ottima in una ε-ottima in modo da pervenire globalmente ad un algoritmo polinomiale.

In questa sezione esamineremo un particolare algoritmo che deriva direttamente dall'algoritmo out-of-kilter modificando il concetto di ottimalità rispetto alle tensioni. Supponiamo che i dati del problema siano interi (cioè i dati riguardanti le funzioni lineari a tratti $f_e(x)$ ovvero le curve Γ_e). Inoltre ridefiniamo tutti i valori di lunghezza nel seguente modo: $d_e := (n+1)\, d_e$. Questo espediente permetterà all'algoritmo di usare sempre numeri interi.

Sia $\Upsilon^\varepsilon := \{(0, v) \in \mathbb{R}^2 : -\varepsilon \le v \le \varepsilon\}$. Definiamo come curva caratteristica ε-approssimata l'insieme $\Gamma^\varepsilon := \Gamma + \Upsilon^\varepsilon$. Possiamo allora definire:

8.26 Definizione. *Siano dati un flusso x ed un potenziale u. Un arco $e \in A$ si dice ε-ottimo se $(x_e, v_e) \in \Gamma_e^\varepsilon$. Il flusso ed il potenziale si dicono ε-ottimi se tutti gli archi sono ε-ottimi.* ∎

In figura 8.27 sono rappresentate una curva Γ e la sua versione approssimata con $\varepsilon = 1$. in questa versione sono disegnate sovrapposte due curve Γ ottenute spostando in alto e in basso la curva originale di $\varepsilon = 1$. Sono valori ε-ottimi tutti quelli situati fra e sulle curve.



Γ_e Γ_e^ε

Figura 8.27

Una proprietà importante dell'ε-ottimalità che garantisce la terminazione dell'algoritmo in un numero finito di passi è costituita dal seguente risultato:

8.27 Lemma. *Se un flusso $\hat{x}$ ed un potenziale $\hat{u}$ sono 1-ottimi, allora $\hat{x}$ è ottimo.*

Dimostrazione. Il corollario 8.8 stabilisce che $\hat{x}$ è ottimo se e solo se esiste una tensione v che soddisfa la condizione $v_e \in [f_e^-(\hat{x}_e), f_e^+(\hat{x}_e)]$. In altri termini, se il problema della tensione ammissibile con intervalli di espansione $[f_e^-(\hat{x}_e), f_e^+(\hat{x}_e)]$ ammette soluzione, abbiamo la garanzia dell'ottimalità di $\hat{x}$. A sua volta l'ammissibilità di questo problema è legata alla condizione espressa nel corollario 8.24, cioè $d(C) \ge 0$ per ogni circuito C, con $d(C)$ dato da (8.1).

Algoritmo a gradazione dei costi
input(G, Γ);
  $u_i := 0$, $\forall i \in N$; sia $x : A\,x = b$, $f_e(x) < +\infty$, $\forall e \in E$;
  if $\nexists x$ ammissibile **then output**("primale non ammissibile");
  $f_e := (n+1)\, f_e$, $\forall e$;
  $\hat{D} := \max_e \max\{f_e^-(\hat{x}_e); -f_e^+(\hat{x}_e); 0\}$;
  $\varepsilon := 2^t$, con $2^t \le \hat{D} < 2^{t+1}$;
  while $\varepsilon \ge 1$ **do**
  begin
    while $\exists\, e = (h,k) : (x_e, v_e) \notin \Gamma^\varepsilon$ **do**
    begin
      if $\delta^+_{h,k} > 0$
      then $(s,t) := (h,k)$ **else** $(s,t) := (k,h)$;
      calcola cammino minimo $P : s \to t$;
      if $L(P) = 0$
      then begin
        calcola massimo flusso $\xi : s \to t \to s$;
        if $\xi = +\infty$ **then output**("primale illimitato");
        $x := x + \xi$;
        calcola cammino minimo $P : s \to t$;
      end
      $u_i := u_i + \min\{\rho_i; \rho_t\}$, $\forall i$;
    end;
    $\varepsilon := \varepsilon/2$;
  end
  $[d_e^-, d_e^+] := [f_e^-(x_e), f_e^+(x_e)]$;
  calcola potenziale u ammissibile;
output(x, u).

Per ciò che riguarda l'ε-ottimalità si ha ovviamente l'analoga condizione necessaria e sufficiente $0 \le d(C) + \varepsilon\,|C|$. Siccome per ipotesi $d(C)$ è multiplo intero di $(n+1)$ e $|C| \le n$, nel momento in cui vale $0 \le d(C) + |C|$ vale anche $0 \le d(C)$. ∎

Si noti che il lemma non asserisce nulla in merito all'ottimalità del potenziale $\hat{u}$, 1-ottimo assieme al flusso $\hat{x}$. In generale, proprio perché 1-ottimo, questo potenziale non è ottimo. Per trovare l'ottimo è necessario risolvere un problema di tensione ammissibile, di cui peraltro abbiamo la garanzia dell'ammissibilità.

Si supponga ora di avere a disposizione un flusso $\hat{x}$ ed un potenziale $\hat{u}$ 2ε-ottimi. Vogliamo ricavare un flusso $\tilde{x}$ ed un potenziale $\tilde{u}$ ε-ottimi. A questo fine useremo lunghezze residue $\tilde{\delta}_e^+$ e $\tilde{\delta}_e^-$ definite in modo diverso da (8.7) e cioè

$$\tilde{\delta}_e^+ := \begin{cases} \delta_e^+ + \varepsilon & \text{se } \delta_e^+ > 0 \\ 0 & \text{se } \delta_e^+ = 0 \end{cases} \qquad \tilde{\delta}_e^- := \begin{cases} \delta_e^- + \varepsilon & \text{se } \delta_e^- > 0 \\ 0 & \text{se } \delta_e^- = 0 \end{cases}$$

dove δ_e^+ e δ_e^- sono definite come in (8.7). Le capacità residue sono invariate. Si noti che il lemma di Minty 8.13 rimane valido anche con questa ridefinizione delle lunghezze residue. In particolare possiamo affermare che:

8.28 Lemma. *Assegnati due nodi s e t, il cammino minimo $P : s \to t$ ha lunghezza $L(P)$*

tale che $L(P) = 0$ *oppure* $L(P) \geq \varepsilon$.

DIMOSTRAZIONE. La tesi deriva immediatamente dal lemma di Minty e dalla nuova definizione di lunghezza residua dato che ogni arco, se non ha lunghezza nulla, è lungo almeno ε. ∎

L'importante conseguenza del lemma 8.28 è che è sufficiente un solo aggiornamento di potenziale per trasformare un arco da 2ε-ottimo in ε-ottimo.

8.29 LEMMA. *Se un arco* $e = (s,t)$ *è* 2ε*-ottimo,* $\tilde{\delta}_e^+ > 0$ *ed esiste un cammino* $P : s \to t$ *di lunghezza positiva (oppure* $\tilde{\delta}_e^- > 0$ *e* $P : t \to s$*) , allora, aggiornando i potenziali secondo il cammino minimo, l'arco diventa* ε*-ottimo.*

DIMOSTRAZIONE. Denotiamo $d_e^- := f_e^-(\hat{x}_e)$ e $d_e^+ := f_e^+(\hat{x}_e)$ (all'interno dei segmenti orizzontali della curva caratteristica ovviamente si ha $d_e^- = d_e^+$). In base all'ipotesi di 2ε-ottimalità e di $\tilde{\delta}_e^+ > 0$, si deve avere

$$d_e^- - 2\varepsilon \leq v_e < d_e^+ \tag{8.14}$$

Quindi $\delta_e^+ = d_e^+ - v_e$ e $\tilde{\delta}_e^+ = d_e^+ - v_e + \varepsilon$. Ovviamente la lunghezza ρ_t del cammino minimo da s a t non può superare $\tilde{\delta}_e^+$. Inoltre in base all'ipotesi sulla lunghezza del cammino minimo $s \to t$ si ha $\rho_t \geq \varepsilon$ e quindi

$$\varepsilon \leq \rho_t \leq d_e^+ - v_e + \varepsilon \tag{8.15}$$

L'aggiornamento del potenziale porta quindi (sull'arco $e = (s,t)$) ad una nuova tensione v'_e data $v'_e := v_e + \rho_t$ e combinando (8.14) e (8.15) si ottiene

$$d_e^- - \varepsilon \leq v'_e \leq d_e^+ + \varepsilon$$

cioè l'ε-ottimalità dell'arco e. La dimostrazione per il caso $\delta_e^- > 0$ è del tutto simile. ∎

Come nell'algoritmo out-of-kilter, se il cammino minimo $s \to t$ ha lunghezza residua nulla, si può aggiungere una circolazione al flusso corrente risolvendo un problema di massimo flusso, con la garanzia che, aggiornato il flusso, esiste un cammino minimo $s \to t$ di lunghezza positiva. Si noti inoltre che non solo si rende ε-ottimo un arco particolare (i,j), ma nella stessa iterazione diventano ε-ottimi (o rimangono tali) anche tutti gli archi (i,k) con $\delta_e^+ > 0$ e tutti gli archi (k,i) con $\delta_e^- > 0$. Questo implica che sono necessarie al più tante iterazioni quanti sono i nodi ed in ogni iterazione è richiesto il calcolo di un cammino minimo oppure il calcolo di un massimo flusso e di un cammino minimo.

Rimane da stabilire come inizializzare l'algoritmo. Se si parte con un valore di potenziale nullo si può pensare di usare un valore iniziale di ε pari a

$$\hat{D} := (n+1) \max_e \max \left\{ f_e^-(\hat{x}_e);\ -f_e^+(\hat{x}_e);\ 0 \right\}$$

che corrisponde alla massima distanza verticale dei punti $(x_e, 0)$ dalle rispettive curve caratteristiche (riscalate del fattore $(n+1)$). Il problema che sorge è che tale valore potrebbe essere infinito se per qualche arco il flusso è al di fuori dell'eventuale intervallo di capacità. Per evitare questo inconveniente bisogna trovare inizialmente un flusso ammissibile per le capacità (oltre che per i vincoli di divergenza) risolvendo un problema di massimo flusso. Una volta trovato il flusso conviene usare, anziché $\hat{D}$, il valore 2^t con t tale che $2^t \leq \hat{D} < 2^{t+1}$, in modo da avere sempre valori interi per ε.

Siamo quindi in grado di stabilire la complessità computazionale dell'algoritmo delineato. Sia $D := \max_e \max |d_e^i|$ e siano $S(n,m)$ e $M(n,m)$ le rispettive complessità computazionali di un algoritmo di cammino minimo e di un algoritmo di massimo flusso su una rete con n nodi e m archi.

8.30 Teorema. *L'algoritmo a gradazione dei costi trova un flusso ed un potenziale ottimi in tempo* $O(\log(n\,D)\,n\,(S(n,m)+M(n,m)))$.

Dimostrazione. Dal valore iniziale $\varepsilon = 2^t \le (n+1)\hat{D} \le (n+1)D$ si ricava $t \le \log_2(n+1)D$. Quindi per arrivare ad $\varepsilon = 1$ c'è bisogno di un numero di iterazioni $O(\log(n\,D))$. In ogni iterazione è richiesto per ogni nodo il calcolo, nel caso peggiore, di un cammino minimo e di un massimo flusso. ∎

8.31 Esempio. Sia data la stessa rete dell'esempio 8.25 ma con i valori di costo moltiplicati per $n+1=5$.

Con questi dati si procede a calcolare un flusso ammissibile che è rappresentato in figura 8.28. L'arco più distante dalla curva caratteristica è l'arco $(1,3)$ con valore 15. Quindi il valore iniziale di ε è 8. Risultano non 8-ottimi solo gli archi (1,3) e (3,4). Prendendo in esame l'arco (1,3) si vede che il cammino minimo $1 \to 3$ ha lunghezza nulla. Quindi è necessario calcolare un massimo flusso che fa circolare 2 unità di flusso fra i nodi 3, 1 e 4. Il flusso risultante è rappresentato in figura 8.29. Questa operazione inoltre spinge in kilter l'arco (3,4) e quindi resta solo un'operazione di cammino minimo per ottenere una soluzione 8-ottima. Si ottiene $\rho = (0, 13, 23, 23) =: u$ e $v = (13, 23, 23, 0, 10)$.

A questo punto ε viene dimezzato a 4 e gli archi non 4-ottimi sono (1,3), (1,2) e (2,4). Prendiamo in esame l'arco (1,2). Essendo $\delta_{12}^- > 0$ bisogna considerare cammini da 2 verso 1. Eseguendo il cammino minimo si ottiene $\rho = (12, 0, 0, 0)$ e quindi $u := u+\rho = (12, 13, 23, 23)$ e $v = (1, 11, 11, 0, 10)$. Con questa operazione anche l'arco (1,3) diventa 4-ottimo. Rimane l'arco (2,4) per il quale è richiesto un cammino minimo da 4 verso 2. Si ottiene $\rho = (0, 8, 0, 0)$ da cui $u = (12, 21, 23, 23)$, $v = (9, 11, 11, 0, 2)$ e la soluzione è 4-ottima.

Si riduce ε a 2. Gli archi non 2-ottimi sono (1,3), (1,2) e (2,4). Prendendo in esame gli archi uscenti da 2 con un unico calcolo di cammino minimo si rendono 2-ottimi (1,2) e (2,4). Si ottiene $\rho = (5, 0, 5, 5)$ da cui $u = (17, 21, 28, 28)$ e $v = (4, 11, 11, 0, 7)$. L'arco (1,3) viene reso 2-ottimo con un cammino minimo che fornisce $\rho = (0, 3, 3, 3)$ da cui $u = (17, 24, 31, 31)$ e $v = (7, 14, 14, 0, 7)$.

Inizia l'ultima iterazione con $\varepsilon = 1$. Gli archi (1,2) e (2,4) non sono 1-ottimi. Da 2 a 1 il cammino minimo ha lunghezza nulla. Quindi si opera un massimo flusso che fa circolare un'unità fra i nodi 4, 3, 1 e 2. con questa operazione l'arco (2,4) è diventato 1-ottimo. Rimane ancora da eseguire un cammino minimo $2 \to 1$. Si ottiene $\rho = (3, 0, 3, 3)$, $u = (20, 24, 34, 34)$ e $v = (4, 14, 14, 0, 10)$.

A questo punto la soluzione ottenuta è 1-ottima e il flusso è ottimo in base al lemma 8.27. Dobbiamo però ancora calcolare il potenziale ottimo. Con il flusso ottimo assegnato gli intervalli di espansione che si calcolano sono $[5, 5]_{12}$, $[15, 15]_{13}$ $[0, \infty)_{14}$, $[0, 0]_{34}$, $[5, 10]_{24}$. Il risultante problema della tensione ammissibile dà come risultato $u = (0, 5, 15, 15)$ e $v = (5, 15, 15, 0, 10)$ (che andrebbero poi divisi per $n+1 = 5$). La soluzione finale è rappresentata in figura 8.30. Si verifichi che, pur essendo questa soluzione diversa da quella dell'esercizio precedente, sono ambedue ottime. ∎

8.9. Pseudoflussi

Questo approccio è basato sull'idea di modificare un flusso non ammissibile rispetto alla divergenza mantenendolo costantemente in kilter fino a farlo diventare ammissibile. Definiamo *pseudoflusso* un flusso che non soddisfa il vincolo $Ax = y = b$. Uno pseudoflusso è ottimo se $(x_e, v_e) \in \Gamma_e$ per ogni arco. Il metodo che useremo richiede che la rete sia di tipo elementare. Quindi bisogna preliminarmente trasformare la rete come precedentemente descritto.

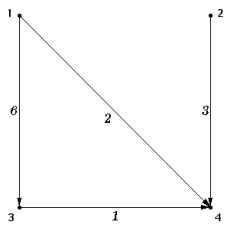


FIGURA 8.28

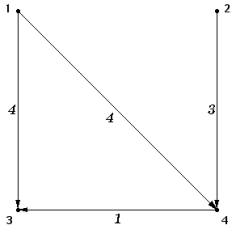


FIGURA 8.29

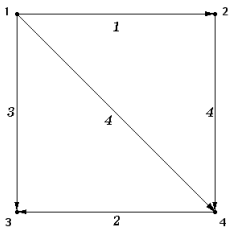


FIGURA 8.30

Supponiamo quindi che ogni curva caratteristica sia l'unione delle due semirette $\{\mathbb{R}_+, d_e\} \cup \{0, d_e - \mathbb{R}_-\}$. Se $d_e \geq 0$ per ogni arco allora uno pseudoflusso ottimo è immediatamente disponibile. Basta prendere $x_e = 0$ per ogni arco e $u_i = 0$ per ogni nodo. Se invece qualche valore d_e è negativo si tratta di risolvere un problema di tensione ammissibile rispetto agli intervalli di espansione $(-\infty, d_e]$.

Se non esiste potenziale ammissibile allora il problema duale non è ammissibile. Altrimenti si procede nel seguente modo: definiamo la quantità $z_i := b_i - y_i > 0$ *difetto* di flusso al nodo i e la quantità $z_i := b_i - y_i < 0$ *eccesso* di flusso al nodo i. Si noti che $\sum_{i \in N} z_i = 0$. Sia $Z := \max |z_i|$. Inizialmente, dato che il flusso è nullo, si ha $Z = \max_i |b_i| =: B$.

L'idea dell'algoritmo consiste nell'aumentare il flusso su un cammino da un nodo s in difetto di flusso ad un nodo t in eccesso di flusso in modo da ribilanciare il vincolo di divergenza. Tuttavia, se si vuole mantenere la condizione di ottimalità, si può aumentare il flusso su un arco solo se $v_e = d_e$. Per ottenere questa condizione bisogna preliminarmente calcolare il cammino di lunghezza residua minima da s a t. Nel caso di reti elementari le lunghezze residue sono del seguente tipo

$$\delta_e^+ = \begin{cases} 0 & \text{if } x_e > 0 \\ d_e - v_e & \text{if } x = 0 \end{cases} \qquad \delta_e^- = \begin{cases} 0 & \text{if } x_e > 0 \\ +\infty & \text{if } x = 0 \end{cases}$$

Sia, come nel caso degli algoritmi precedenti, $w_j := \min\{\rho_j\,,\, \rho_t\}$ dove ρ_j è la minima lunghezza residua da s a j. Allora u può essere aggiornato come $u := u + w$, la condizione $(x_e, u_e) \in \Gamma_e$ rimane valida e del flusso può essere aumentato sul cammino minimo $s \to t$ poiché $v_e = d_e$ su tutti gli archi di tale cammino.

L'idea chiave dell'algoritmo riguarda la quantità di flusso da inviare sul cammino. Intuitivamente sembrerebbe conveniente inviare la maggior quantità possibile di flusso in modo da ribilanciare la divergenza o in s o in t. Però tale strategia si rivela inefficiente. Può succedere infatti che su un arco sia inviato un flusso pari a M e che in un secondo tempo sia inviato sullo stesso arco, ma in senso contrario, un flusso pari a $M - 1$. A questo punto solo un'unità di flusso è presente sull'arco e solo un'unità di flusso può successivamente essere fatta transitare se l'arco appartiene ad un cammino minimo, ma con orientazione opposta. Può quindi succedere che tutte le successive iterazioni non riescano a far transitare più di un'unità di flusso alla volta, rendendo quindi l'algoritmo pseudopolinomiale.

Una situazione come quella descritta non può avvenire se il flusso presente in ogni arco è un multiplo intero del flusso che si vuole inviare. Certamente non si vuole inviare solo un'unità di flusso alla volta. Conviene invece inviare quantità pari a $2^q, 2^{q-1}, 2^{q-2}, \ldots, 1$. Per decidere quale potenza di 2 usare si definisca un insieme di sorgenti $S(y, \varepsilon) := \{i \in N : z_i \geq \varepsilon\}$ ed un insieme di pozzi $T(y, \varepsilon) := \{i \in N : z_i \leq -\varepsilon\}$. Si sceglie il più alto valore di $\varepsilon = 2^q$ per cui gli insiemi non sono ambedue vuoti, cioè

$$\hat{q} = \max\{q : S(y, 2^q) \neq \emptyset \wedge T(y, 2^q) \neq \emptyset\}$$

Algoritmo pseudoflussi
begin
  let $S(y,\varepsilon) := \{i \in N : b_i - y_i \geq \varepsilon\}$;
  let $T(y,\varepsilon) := \{i \in N : b_i - y_i \leq -\varepsilon\}$;
  aggiungi archi artificiali;
  calcola tensione v^0 ammissibile;
  if $\nexists$ tensione ammissibile **then return**("duale non ammissibile");
  inizializza $x := 0$ and $v := v^0$;
  $B^+ := \max b_i$; $B^- := \max -b_i$; $B := \max |b_i|$;
  $\hat{q} := \max\{q : 2^q \leq \min\{B^+; B^-\}\}$;
  $\varepsilon := 2^{\hat{q}}$;
  while $\exists i : y_i \neq b_i$ **do**
  begin
    while $S(y,\varepsilon) \neq \emptyset \wedge T(y,\varepsilon) \neq \emptyset$ **do**
    begin
      seleziona $s \in S(y,\varepsilon)$;
      calcola cammino minimo $P : s \to t \in T(y,\varepsilon)$, con ρ_j distanze minime $s \to j$;
      $u := u + \min\{\rho_t, \rho_j\}$;
      for $e \in P^+$ **do** $x_e := x_e + \varepsilon$; **for** $e \in P^-$ **do** $x_e := x_e - \varepsilon$;
      $y_s := y_s + \varepsilon$; $y_t := y_t - \varepsilon$;
    end;
    $\varepsilon := \varepsilon/2$;
  end;
  if $\exists$ arco artificiale e con $x_e > 0$ **then return**("primale non ammissibile");
end

e si pone $\varepsilon := 2^{\hat{q}}$. Con questa definizione la proprietà $|z_i| \leq 2\varepsilon$ per ogni nodo dell'insieme deve valere o per $S(y,\varepsilon)$ o per $T(y,\varepsilon)$ (o per ambedue).

A questo punto si calcola il minimo cammino residuo da un nodo specificato $s \in S(y,\varepsilon)$ al più vicino nodo $t \in T(y,\varepsilon)$, si modificano i potenziali e si invia un flusso pari a ε sul cammino minimo. Questo comporta un aggiornamento nei valori di y, e cioè $y_s := y_s + \varepsilon$ e $y_t := y_t - \varepsilon$, ed eventualmente di $S(y,\varepsilon)$ e $T(y,\varepsilon)$.

Queste operazioni vengono ripetute finché $S(y,\varepsilon) = \emptyset$ oppure $T(y,\varepsilon) = \emptyset$. La proprietà precedentemente enunciata sui valori z_i garantisce che non sono necessarie più di n iterazioni per raggiungere questa condizione. Poi si dimezza ε e si prosegue allo stesso modo. L'interezza dei dati garantisce che $S(y,1)$ e $T(y,1)$ diventano vuoti nella medesima iterazione, dopodiché si ottiene il bilanciamento della divergenza, cioè $y = b$.

Può avvenire che in qualche iterazione si abbia un cammino minimo di lunghezza infinita, quando ogni cammino da s a $T(y,\varepsilon)$ ha lunghezza infinita. Sembrerebbe quindi che sia impossibile inviare del flusso in $T(y,\varepsilon)$ e quindi bilanciare la divergenza. Tuttavia sarebbe errato concludere che il problema primale non è ammissibile. Infatti $T(y,\varepsilon)$ non contiene tutti i pozzi, ma solo quelli il cui sbilanciamento non è meno di ε. Si consideri ad esempio il grafo in figura 8.31 dove i numeri accanto ai nodi rappresentano lo sbilanciamento attuale. Con $\varepsilon = 2$ vi è solo una sorgente e solo un pozzo e non c'è modo di trasferire flusso fra i due nodi. Tuttavia una soluzione ammissibile esiste e l'algoritmo la troverebbe con il valore $\varepsilon = 1$. Un modo per aggirare l'ostacolo consiste nell'introdurre archi artificiali ad altissimo costo che permettano di avere sempre cammini di lunghezza finita. Se, a fine algoritmo, qualche arco artificiale contiene del flusso positivo, allora il problema primale non è ammissibile.

Per valutare la complessità computazionale dell'algoritmo notiamo che vi sono globalmente

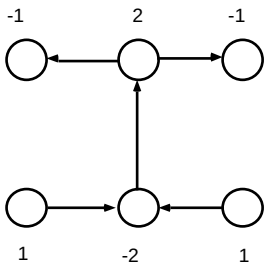


FIGURA 8.31

$O(\log B)$ iterazioni. Per ogni iterazione vi sono al più n cammini minimi da calcolare. Quindi abbiamo dimostrato che:

8.32 TEOREMA. *L'algoritmo pseudoflussi, applicato ad una rete elementare, ha complessità computazionale* $O(\log(B)\, n\, S(n,m))$. ∎

Se la rete assegnata è invece di tipo semplice bisogna tener conto che nella trasformazione in rete elementare si sono aggiunti tanti nodi quanti sono gli archi, per cui si ha:

8.33 COROLLARIO. *L'algoritmo pseudoflussi, applicato ad una rete semplice, ha complessità computazionale* $O(\log(B)\, m\, S(m,m))$. ∎

Con opportune modifiche si può rendere l'algoritmo fortemente polinomiale (Orlin [1993]). Senza descrivere questa versione dell'algoritmo ci limitiamo a fornire il risultato di complessità computazionale che è $O(n\ \log n\ S(n,m))$ per reti elementari e $O(m\ \log n\ S(n,m))$ per reti semplici.

8.34 ESEMPIO. Riconsideriamo la stessa rete dei due esempi precedenti. La rete risultante dopo le trasformazioni è raffigurata in figura 8.32 (fra i nodi 1 e 2 e fra i nodi 2 e 4 vi sono due archi paralleli e opposti rappresentati come un unico arco con doppia orientazione). I valori di b e d di tale rete sono:

$$b = (11,\ 7,\ 4,\ 4,\ -7,\ -1,\ -5,\ -3,\ -2,\ -3,\ -2,\ -3)$$

$$d_{2,1} = 2 \qquad d_{1,2} = 1, \qquad d_{1,6} = 0 \qquad d_{3,6} = 0 \qquad d_{1,5} = 3$$

$$d_{3,5} = 0 \qquad d_{1,7} = 0 \qquad d_{4,7} = 0 \qquad d_{4,3} = 2 \qquad d_{3,8} = 0$$

$$d_{4,8} = 0 \qquad d_{3,9} = 2 \qquad d_{4,9} = 0 \qquad d_{4,2} = 2 \qquad d_{2,10} = -1$$

$$d_{4,10} = 0 \qquad d_{2,11} = 0 \qquad d_{4,11} = 0 \qquad d_{2,12} = 1 \qquad d_{4,12} = 0 \qquad d_{2,4} = 2$$

Il potenziale iniziale è $u^0 = (1,1,1,1,1,1,1,1,1,0,1,1)$ ottenuto risolvendo un problema di cammino minimo da 10 a 2. Inoltre $\varepsilon = 4$, $S(0,4) = \{1,2,3,4\}$, $T(0,4) = \{5,7\}$. Nelle varie iterazioni tutti i cammini minimi tranne il cammino $1 \to 5$ con $\varepsilon = 4$ sono di lunghezza nulla. Il potenziale finale è $u = (1,2,4,4,4,1,1,4,4,1,2,3)$. Le iterazioni sono riassunte nella seguente tabella. In figura 8.33 vi è il flusso finale.

Rimane da calcolare il flusso della rete originaria a partire da quello delle rete elementare. Non è difficile vedere che per ogni arco originale (i,j) con primo valore di capacità c_1 il flusso è uguale alla somma del flusso degli archi uscenti da i in direzione j nella rete elementare meno il flusso su (j,i) (nella rete elementare) più ancora c_1. Quindi si ha $x_{1,2} = 2$ (come nella rete elementare), $x_{1,3} = 3 + 1 - 2 = 2$, $x_{1,4} = 5 - 1 = 4$. $x_{3,4} = 0 - 3 = -3$, $x_{2,4} = 1 + 3 + 2 + 3 - 4 = 5$. Per ciò che riguarda i potenziali ottimi basta notare che i potenziali della rete originale sono gli stessi dei rispettivi nodi della rete elementare. ∎

S	T	cammino min.	$b-y$
$\varepsilon = 4$			
{1,2,3,4}	{5,7}	1→7	(7,7,4,4,−7,−1,−1,−3,−2,−3,−2,−3)
{1,2,3,4}	{5}	1→5	(3,7,4,4,−3,−1,−1,−3,−2,−3,−2,−3)
$\varepsilon = 2$			
{1,2,3,4}	{5,8,9,10,11,12}	1→5	(1,7,4,4,−1,−1,−1,−3,−2,−3,−2,−3)
{2,3,4}	{8,9,10,11,12}	2→4→8	(1,5,4,4,−1,−1,−1,−1,−2,−3,−2,−3)
{2,3,4}	{9,10,11,12}	2→4→9	(1,3,4,4,−1,−1,−1,−1,0,−3,−2,−3)
{2,3,4}	{10,11,12}	2→10	(1,1,4,4,−1,−1,−1,−1,0,−1,−2,−3)
{3,4}	{11,12}	3→5→1→2→11	(1,1,2,4,−1,−1,−1,−1,0,−1,0,−3)
{3,4}	{12}	3→5→1→2→12	(1,1,0,4,−1,−1,−1,−1,0,−1,0,−1)
$\varepsilon = 1$			
{1,2,4}	{6,5,7,8,10,12}	1→6	(0,1,0,4,−1,0,−1,−1,0,−1,0,−1)
{2,4}	{5,7,8,10,12}	2→1→5	(0,0,0,4,0,0,−1,−1,0,−1,0,−1)
{4}	{7,8,10,12}	4→2→1→7	(0,0,0,3,0,0,0,−1,0,−1,0,−1)
{4}	{8,10,12}	4→8	(0,0,0,2,0,0,0,0,0,−1,0,−1)
{4}	{10,12}	4→2→10	(0,0,0,1,0,0,0,0,0,0,0,−1)
{4}	{12}	4→2→12	(0,0,0,0,0,0,0,0,0,0,0,0)

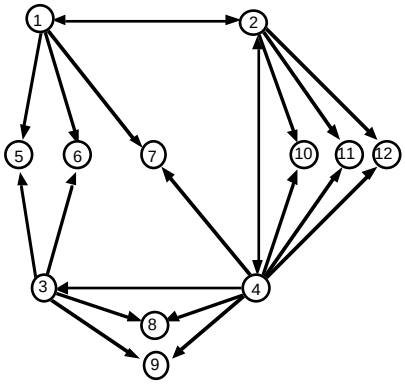


FIGURA 8.32

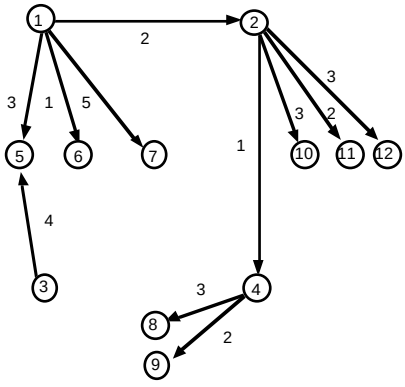


FIGURA 8.33

8.10. Assegnamento pesato

Il problema dell'assegnamento pesato può essere modellato come un problema di flusso di costo minimo su un grafo bipartito (N_1, N_2, E), dove i nodi di $N_!$ sono sorgenti con $b_i = 1$ e i nodi di N_2 sono pozzi con $b_i = -1$. Il grafo bipartito è normalmente completo (altrimenti si assegnano costi elevatissimi agli archi che non dovrebbero essere ammessi) e ad ogni arco viene associata una funzione di costo elementare.

Si può quindi applicare l'algoritmo pseudoflussi la cui esecuzione in questo caso risulta notevolmente semplificata, in quanto si parte subito con $\varepsilon = 1$. Inoltre, siccome i costi possono tutti essere traslati di una medesima quantità senza alterare l'ottimo (perché?) possiamo assumere senza perdita di generalità che tutti i costi sono non negativi e quindi l'algoritmo può iniziare con i valori di flusso e potenziale nulli.

In un grafo bipartito i cammini devono 'pendolare' fra N_1 e N_2. Le lunghezze residue da

N_1 a N_2 sono finite e possono essere anche nulle. Le lunghezza residua di una arco da N_2 a N_1 è invece o nulla o infinita. È nulla se nell'arco è presente del flusso, infinita se non c'è flusso.

All'inizio di un'iterazione generica vi sono m nodi bilanciati in N_1 ed altrettanti in N_2. Da ogni nodo bilanciato di N_1 parte un'unità di flusso diretta verso un ben determinato nodo bilanciato di N_2 e quest'unità di flusso si serve di un unico arco che rappresenta pertanto l'assegnamento. L'iterazione consiste nel cercare un cammino minimo da un nodo non ancora bilanciato di N_1 verso un nodo non ancora bilanciato di N_2. Vogliamo far vedere che alla fine dell'iterazione otterremo una situazione analoga con $m+1$ nodi bilanciati per parte ed ogni nodo di N_1 assegnato ad esattamente un nodo di N_2.

Se il cammino minimo da un nodo sbilanciato $s \in N_1$ verso un nodo sbilanciato t in N_2 consiste del solo arco (s,t) allora si immette un'unità di flusso su tale arco, si aggiornano i potenziali, e si trova la condizione cercata.

Se invece il cammino consiste di più archi, deve necessariamente percorrere archi da N_2 a N_1 che hanno flusso positivo (e quindi unitario) e corrispondono agli assegnamenti calcolati nelle precedenti iterazioni. L'unità di flusso ogni qualvolta va da N_2 ad N_1 'cancella' il flusso precedente e tutti gli assegnamenti dei nodi del cammino minimo vengono cambiati. Come si vede facilmente il numero di archi su cui si immette flusso è uno di più del numero degli assegnamenti cancellati.

L'assegnamento minimo quindi si trova semplicemente risolvendo n ($n = |N_1| = |N_2|$) problemi di cammino minimo e abbiamo dimostrato che:

8.35 Teorema. *Il problema dell'assegnamento può essere risolto con l'algoritmo pseudo-flussi in tempo* $O(n\,S(n,n^2))$. ∎

Quindi usando l'algoritmo di Dijkstra la complessità è $O(n^3)$. Facciamo notare che vi sono in letteratura algoritmi più veloci, ma molto più complessi nella struttura e funzionamento.

8.36 Esempio. Si consideri un problema di assegnamento con la seguente tabella di costi:

40	10	78	73	5	95
3	67	50	4	93	32
58	79	19	51	13	81
27	23	77	62	32	29
55	22	62	72	11	71
93	68	51	4	1	63

In figura 8.34 sono rappresentati i cammini minimi delle sei iterazioni insieme alla soluzione corrente di assegnamento e l'assegnamento finale. I valori di potenziale di ogni iterazione sono elencati nella seguente tabella:

u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	u_{10}	u_{11}	u_{12}
0	0	0	0	0	0	0	0	0	0	0	0
0	5	5	5	5	5	5	5	5	5	5	5
3	5	8	8	8	8	8	8	8	8	8	8
16	23	8	26	26	26	26	26	26	26	21	26
39	47	31	26	50	50	50	49	50	50	44	50
56	67	52	43	50	71	70	66	71	71	61	71
60	71	57	47	54	71	74	70	76	75	65	76

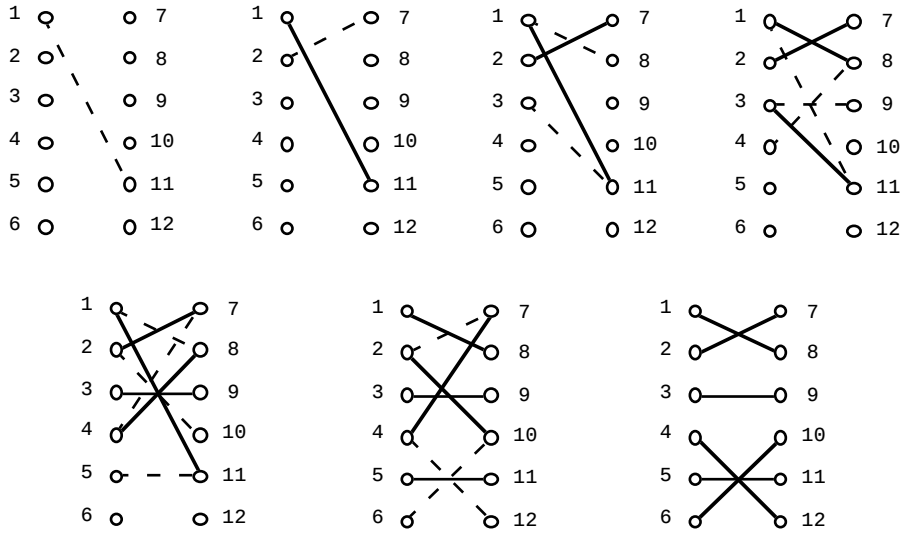


FIGURA 8.34

I valori di tensione sono riportati nella seguente tabella. Si noti come $v_{ij} \leq d_{ij}$ per ogni arco e come $v_{ij} = d_{ij}$ per gli archi assegnati (valori evidenziati in grassetto).

14	**10**	16	15	5	16
3	-1	5	4	-6	5
17	13	**19**	18	8	19
27	23	29	28	18	**29**
20	16	22	21	**11**	22
3	-1	5	**4**	-6	5

8.11. Programmazione lineare su reti di flusso

Il problema primale (8.2), se le funzioni f_j sono lineari a tratti e convesse, può essere trasformato nel seguente problema di programmazione lineare:

$$\begin{aligned} \min \quad & d\,x \\ & A\,x \;=\; b \\ & c^- \leq x \leq c^+ \end{aligned} \tag{8.16}$$

Bisogna notare che in (8.16) la matrice A non ha rango pieno e pertanto non è possibile adottare direttamente il metodo del simplesso. Un'alternativa potrebbe consistere nell'eliminazione di un'equazione da $A\,x = b$. Però normalmente non si opera in questo modo in quanto è più semplice aggiungere ad A una qualsiasi colonna della matrice identica ed introdurre quindi un'unica variabile ausiliaria s_i. Questa variabile può anche essere svincolata dato che deve necessariamente valere 0 per ogni flusso ammissibile.

Una matrice di base B di (8.16) deve contenere la colonna ausiliaria, altrimenti sarebbe singolare, e il grafo associato $G(B)$ deve essere un albero di supporto con radice associata al valore 1 della colonna ausiliaria (vedi capitolo 2). La possibilità di associare un albero di

supporto ad ogni soluzione di base permette di sviluppare il metodo del simplesso facendo uso esclusivamente della struttura della rete.

Si consideri innanzitutto il calcolo della variabile duale u relativa ad una base β. Si era visto nella sezione 7.6 che $u := d_\beta B^{-1}$. Tuttavia per problemi di reti di flusso è più opportuno identificare le variabili duali con i potenziali e questi, nella teoria che lega flussi e potenziali in condizioni di convessità, risultano essere l'opposto delle variabili duali come definite sopra. Pertanto, per problemi su reti di flusso, le variabili duali vengono definite come $u := -d_\beta B^{-1}$. Incidentalmente, questo cambiamento di segno è equivalente a moltiplicare per -1 entrambi i membri dell'equazione (8.16). Quindi (indicando con r la radice)

$$u = -d_\beta B^{-1} \quad \Longrightarrow \quad u B = -d_\beta \quad \Longrightarrow v_e = d_e \quad \forall e \in G(B), \quad u_r = 0$$

Il calcolo di u è pertanto immediato. Partendo dalla radice si pone il corrispondente potenziale uguale a zero. Poi si visitano i nodi dell'albero assegnando ricorsivamente i potenziali nel seguente modo:

$$u_j := u_i + d_e \qquad \text{se } e \in G(B),\ e \mapsto (i,j)$$

oppure

$$u_j := u_i - d_e \qquad \text{se } e \in G(B),\ e \mapsto (j,i)$$

I costi ridotti degli archi fuori base si calcolano come

$$\bar{d}_e := d_e - d_\beta B^{-1} A_e = d_e + u A_e = d_e - v_e$$

Quindi la condizione di ottimalità diventa

$$\begin{aligned} v_e &\le d_e \quad \text{se} \quad x_e = d_e^- \\ v_e &\ge d_e \quad \text{se} \quad x_e = d_e^+ \end{aligned}$$

Se la condizione di ottimalità non è verificata per un certo arco e_p, questo può entrare nella base successiva e viene pertanto aggiunto al grafo $G(B)$. Il nuovo sottografo $G(B) \cup e_p$ non è più un albero perché è stato formato un circuito C.

Il modo per migliorare il costo e ripristinare una base, ovvero un albero, consiste nel fare circolare del flusso nel circuito C in modo da rispettare il vincolo $A\,x = b$ e far sì che il flusso di un arco in base diventi uguale o all'estremo inferiore di capacità o a quello superiore. Più esattamente si procede così: supponendo di orientare il circuito C come e_p se $x_p = c_p^-$ oppure in modo opposto se $x_p = c_p^+$ e indicando con ξ il flusso aggiuntivo, si ha, negli archi del circuito, un flusso $x'_e := x_e + \xi$ se $e \in C^+$ oppure un flusso $x'_e := x_e - \xi$ se $e \in C^-$. Il problema è ora quello di calcolare il massimo valore di ξ per cui $c_e^- \le x'_e \le c_e^+$. Se il massimo non esiste il problema è illimitato, altrimenti in corrispondenza del massimo almeno uno dei flussi degli archi del circuito si trova in uno dei due valori estremi di capacità. Archi del circuito per cui si ottiene il massimo vengono detti *bloccanti*. Togliendo un arco bloccante si ripristina la base e la struttura di albero con radice, per cui l'algoritmo può a questo punto ripetersi.

8.37 Esercizio. Tenendo conto della struttura di albero di una base, si calcoli quante basi degeneri sono associate alla medesima soluzione di base per il problema dell'assegnamento pesato (grafo bipartito completo). Si usi il fatto che il numero di alberi di supporto di una grafo completo (non bipartito) è n^{n-2}. ∎

```
procedura calcola potenziali
    u(1) := 0;
    j := thread(1);
    while j ≠ 1 do
    begin
        i := pred(j)
        if (i, j) ∈ A
            then u_j := u_i + d_ij;
            else u_j := u_i − d_ji;
        j := thread(i);
    end
end.
```

```
procedura identifica ciclo
    i := h; j := k;
    while depth(i) > depth(j) do i := pred(i);
    while depth(i) > depth(j) do i := pred(i);
    while i ≠ j do
    begin
        i := pred(i); j := pred(j);
    end
end.
```

8.12. Struttura dell'algoritmo

Una rappresentazione dell'albero di base che permetta di eseguire in modo efficiente i passi dell'algoritmo precedentemente descritti si può effettuare tramite tre liste così strutturate:
- lista dei predecessori (indicata con *pred*): l'elemento i-esimo della lista punta al (cioè *pred(i)* contiene l'etichetta del) nodo che precede il nodo i nel cammino dalla radice al nodo i.
- lista delle profondità (indicata con *depth*): l'elemento i-esimo della lista contiene la profondità del nodo i (cioè il numero di archi nel cammino dalla radice al nodo i).
- lista dei successori (indicata con *thread*): l'elemento i-esimo della lista punta al (cioè *thread(i)* contiene l'etichetta del) nodo che segue il nodo i in una visita in profondità dell'albero a partire dalla radice (l'ultimo nodo della visita punta alla radice).

Il calcolo dei potenziali si effettua con complessità $O(n)$ sfruttando l'informazione contenuta in *thread* e *pred* come indicato nella procedura *calcola potenziali*.

Una volta scelto l'arco da far entrare in base, la determinazione del ciclo creato nell'albero dall'aggiunta dell'arco $e_p = (h, k)$ si effettua con complessità $O(n)$ come indicato nella procedura *identifica ciclo*. Questa procedura termina quindi non appena viene trovato il primo nodo comune, detto *apice*, ai due cammini da h e da k verso la radice. All'interno della procedura si possono inoltre inserire direttamente delle istruzioni per il calcolo del massimo flusso transitabile nel circuito e quindi anche degli archi bloccanti. Dato l'alto grado di degenerazione che tipicamente presentano i problemi definiti su reti di flusso è opportuno adottare delle strategie anticiclaggio. A tal fine si consideri la seguente definizione:

8.38 Definizione. *Un albero di base è* fortemente ammissibile *se da ogni nodo si può inviare verso la radice una quantità positiva di flusso senza violare le capacità. Alternativamente un albero di base è fortemente ammissibile se ogni arco vuoto dell'albero è diretto verso la radice ed ogni arco saturo dell'albero è diretto verso le foglie.* ∎

Facciamo prima vedere come si possa assicurare che da una base fortemente ammissibile si passi ad un'altra base fortemente ammissibile con la seguente regola per la determinazione dell'arco che deve uscire dalla base: l'arco che deve uscire dalla base è l'ultimo fra gli archi bloccanti, visitando il circuito secondo la sua orientazione partendo dall'apice.

Per dimostrare che questa regola mantiene la proprietà di forte ammissibilità si indichi con C_1 la parte di circuito dalla radice al nodo h (seguendo l'orientazione del circuito), con C_2 quella dal nodo k all'apice (vedi figura 8.35) e si considerino i casi:

1) l'arco bloccante (p, q) è in C_1: si definisca come C_{11} la parte di C_1 dall'apice a p e con C_{12} quella da q a h. Gli archi di C_{11} mantengono ovviamente la proprietà di forte

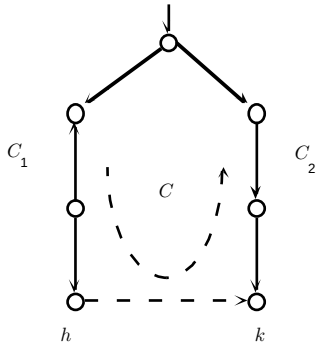


FIGURA 8.35

ammissiblità perché il flusso che eventualmente è stato aggiunto nel ciclo non può vuotare un arco diretto verso le foglie o saturare un arco diretto verso la radice. Gli archi di C_{12} vengono appesi a C_2 nel nuovo albero. Nessun arco di C_{12} o C_2 diretto verso la radice (nel nuovo albero) può essere saturo perché, in base alla regola, (p, q) è stato l'ultimo arco bloccante. Similmente nessun arco C_{12} o C_2 diretto verso le foglie può essere vuoto.

2) l'arco bloccante (p, q) è in C_2: si definisca come C_{21} la parte di C_2 da k a p e con C_{22} quella da q all'apice. L'arco (p, q) non può essere bloccante con un flusso nullo nel circuito perché sarebbe violata la proprietà di forte ammissibilità, quindi l'aumento di flusso è positivo e di conseguenza è possibile far circolare del flusso positivo negli archi di C_{21} verso l'apice nel nuovo albero. Per gli archi di C_1 e C_{22} valgono i ragionamenti del caso precedente.

Si noti che se l'iterazione è degenere solo il caso 1) può verificarsi. In questo caso il potenziale varia soltanto sui nodi di C_{12} e su quelli che sono appesi a questi. Vogliamo far vedere che i potenziali possono soltanto aumentare di una quantità positiva durante una serie di iterazioni degeneri consecutive. Indichiamo con u' i potenziali prima del cambio di base e con u'' quelli dopo il cambio di base. Supponiamo che l'arco $e_p = (h, k)$ entrante in base sia vuoto, quindi $\bar{d}'_p = d_p - u'_k + u'_h < 0$. All'iterazione successiva (h, k) è in base, quindi $d_p - u''_k + u''_h = 0$. In base all'osservazione precedente per cui l'arco uscente (p, q) si trova in C_1, si ha $u'_k = u''_k$ e cioè $u'_h < u''_h$. Quindi il potenziale dei nodi di C_{12} e di quelli che sono appesi a questi aumenta di $u''_h - u'_h = -\bar{d}'_p$. Supponiamo ora che l'arco entrante sia saturo, $\bar{d}'_p = d_p - u'_k + u'_h > 0$. Mantenendo l'orientazione del ciclo coerente con la discussione e la notazione precedente abbiamo che $k \in C_1$ e $h \in C_2$. Quindi $u'_h = u''_h$ e cioè $u'_k < u''_k$ che comporta la medesima conclusione del caso precedente.

Il numero di iterazioni degeneri consecutive non può essere illimitato dato che ad ogni iterazione degenere il valore $\sum_i u_i$ aumenta di una quantità finita e quindi non vi possono essere ripetizioni di basi (basi uguali devono avere lo stesso valore $\sum_i u_i$). Analogamente il numero di iterazioni non degeneri non può essere illimitato, perché il costo della funzione obiettivo decresce di una quantità finita e quindi basi uguali non possono ripresentarsi.

Ovviamente bisogna inizializzare l'algoritmo con una base fortemente ammissibile. In generale non è detto che una tale base esista. Nell'esercizio seguente il lettore è invitato a trovare delle condizioni di esistenza. Da queste, una volta ammessa l'esistenza di una base fortemente ammissibile, non è dificile trovare un algoritmo che ne calcoli una.

8.39 ESERCIZIO. Sia data una soluzione ammissibile di flusso e si consideri la rete residua. Si contraggano le componenti fortemente connesse della rete residua formando un grafo con pseudonodi ed archi. Si dimostri che ogni soluzione di flusso sul grafo ha valori invarianti

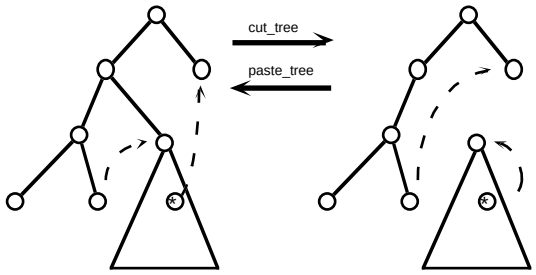


FIGURA 8.36

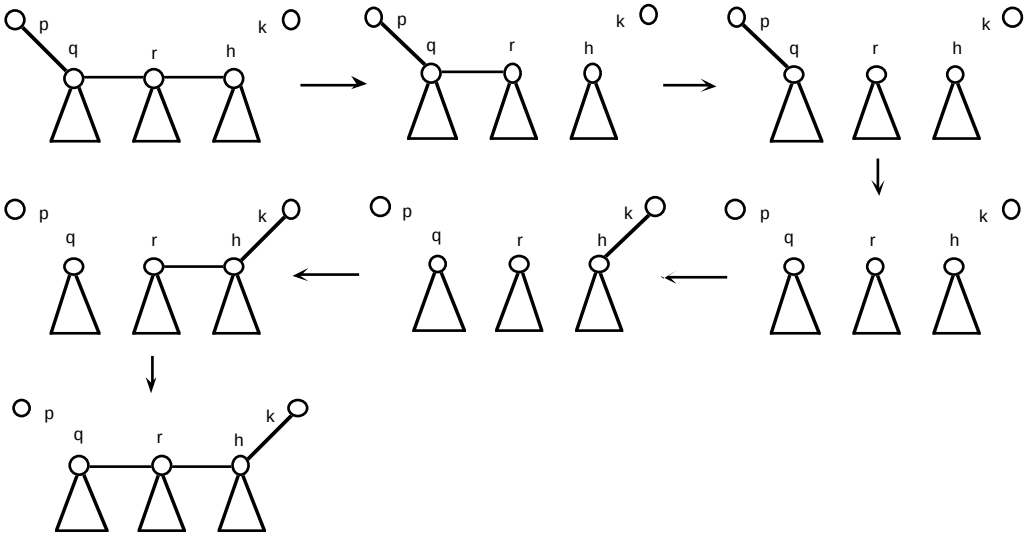


FIGURA 8.37

sugli archi che connettono gli pseudonodi (quindi il problema è decomponibile in tanti sottoproblemi quanti sono gli pseudonodi). Si dimostri che esiste una base fortemente ammissibile (per tutto il grafo) se e solo se esiste un unico pseudonodo con solo archi entranti. In ogni caso, se il problema viene decomposto secondo gli pseudonodi allora una base fortemente ammissibile esiste per ogni sottografo corrispondente ad uno pseudonodo. ∎

Vediamo ora come aggiornare efficientemente le liste che rappresentano l'albero di base. Nel seguito indichiamo con $\tilde{C}$ la parte di circuito compresa fra l'arco (p, q) uscente dalla base e l'arco (h, k) entrante in base. Supponiamo, per esemplificare, che $\tilde{C} = \{q, r, h\}$, con q, r e h nodi e siano T_q, T_r e T_h i sottoalberi appesi ai rispettivi nodi.

– Aggiornamento di *pred*: è sufficiente cambiare i puntatori dei nodi di $\tilde{C}$. In particolare $pred(h) := k$, $pred(r) := h$, $pred(q) := r$. Gli altri valori di *pred* rimangono invariati.

– Aggiornamento di *depth*: si ha $d = depth(h)$, $depth(h) := depth(k)+1$, $\Delta = depth(h)-d$, $depth(r) := depth(h)+1$, $depth(q) := depth(r)+1$ (dove i valori sono quelli immediatamente aggiornati). Si noti che $depth(r)$ è variato di $\Delta + 2$ e $depth(q)$ di $\Delta + 4$. Conseguentemente $depth(i) := depth(i) + \Delta$, $i \in T_h$, $depth(i) := depth(i) + \Delta + 2$, $i \in T_r$, $depth(i) := depth(i) + \Delta + 4$, $i \in T_q$. Gli altri valori di *depth* rimangono invariati.

– Aggiornamento di *thread*: si definiscano dapprima le due procedure $cuttree(T, t)$ e $pastetree(T_1, T_2, t)$. La prima riceve in ingresso la lista dei successori di un albero T e un nodo t. In uscita fornisce le liste dei successori dei due alberi T_1 e T_2 che si ottengono

staccando dall'albero T il sottoalbero appeso nel nodo t (t diventa radice di T_2 e l'arco che in T collega t e *pred(t)* non è presente né in T_1 né in T_2). La seconda riceve in ingresso le liste dei successori di due alberi T_1 e T_2 e un nodo t di T_1. In uscita fornisce la lista dei successori dell'albero che si ottiene attaccando l'albero T_2 al nodo t dell'albero T_1 tramite un nuovo arco che collega la radice di T_2 a t. Supponiamo inoltre che, per una maggiore efficienza della scansione delle liste dei successori, ogni lista dei successori possieda un'intestazione dove viene indicato il nodo radice e l'ultimo nodo della visita (cioè quello che punta alla radice). Indichiamo con $T(0)$ e $T(*)$ tali nodi per il generico albero T.

La procedura *cuttree*(T, t) opera nel seguente modo: sia $i :=$ *pred(t)*, poi si itera $i :=$ *thread(i)* finché si trova q tale che *thread(q)* $= t$; poi si pone $i := t$ e si itera $i :=$ *thread(i)* finché *depth(i)* $\geq$ *depth(t)*. Sia j l'ultimo nodo visitato per cui *depth(j)* $< d$. I nodi visitati da t a j costituiscono il sottoalbero T_2 appeso al nodo t. Il "taglio" di T_2 si effettua semplicemente copiando da T le liste dei successori per T_1 e T_2 e modificando in T_1 *thread(q)* $:=$ *thread(j)* e $T_1(*) := q$ nel caso che $T(*) = j$ e modificando in T_2 *thread(j)* $:= t$, $T_2(0) := t$ e $T_2(*) := j$ (si veda la figura 8.36).

La procedura *pastetree*(T_1, T_2, t) opera nel seguente modo: si pone $i := t$ e si itera $i :=$ *thread(i)* finché *depth(i)* $\geq$ *depth(t)*. Sia q l'ultimo nodo visitato per cui *depth(q)* $< d$. La lista dei successori del nuovo albero T viene copiata da T_1 e T_2 modificando in T *thread*$(T_2(*))$ $:=$ *thread(q)* e *thread(q)* $:= T_2(0)$ (si veda la figura 8.36).

Tramite le due procedure la lista dei successori dell'albero di base viene modificata attraverso i seguenti passi: si staccano dall'albero T_h, T_r e T_q nell'ordine; poi si appendono nell'ordine T_h a k, T_r a h e T_q a r. A questo punto l'aggiornamento della lista dei successori è completato (figura 8.37).

Si noti che il calcolo dei potenziali può essere accelerato dato che variano soltanto i potenziali nei sottoalberi T_p, T_r e T_h (e di una quantità costante per ogni sottoalbero). Per il calcolo dell'arco da far entrare in base si usa normalmente la seguente strategia: gli archi vengono scanditi (secondo un ordine arbitrario) e quelli che violano la condizione di ottimalità vengono inseriti in un insieme di "candidati". L'inserimento termina quando l'insieme raggiunge una cardinalità prefissata K. Dopodiché ad ogni iterazione si esaminano soltanto i costi ridotti di tutti gli archi candidati. Quando la condizione di ottimalità è verificata per tutti gli archi candidati si passa a generare un altro insieme di archi candidati a partire dall'ultimo arco che era stato scandito nella precedente generazione dell'insieme candidato. Questa regola realizza un compromesso fra la regola cosiddetta di Dantzig, cioè di scegliere ad ogni iterazione l'arco di minor costo ridotto (caso $K := m$), e la regola di scegliere il primo arco a costo ridotto negativo in una scansione circolare degli archi (caso $K := 1$). Lo svantaggio della regola di Dantzig è che per reti con molti archi è computazionalmente oneroso dover calcolare i costi ridotti per tutti gli archi ad ogni passo. Lo svantaggio dell'altra regola è che possono essere richieste molte più iterazioni. Una opportuna scelta di K può costituire un efficace compromesso.

8.40 Esempio. Si consideri le rete di figura 8.38 dove bisogna inviare 10 unità di flusso dal nodo 1 al nodo 12. Le capacità degli archi sono $c_{1,2} = 8$, $c_{1,5} = 5$, $c_{2,3} = 6$, $c_{2,6} = 1$, $c_{3,4} = 8$, $c_{4,8} = 7$, $c_{5,6} = 6$, $c_{5,9} = 4$, $c_{6,3} = 5$, $c_{6,10} = 2$, $c_{6,7} = 1$, $c_{7,8} = 3$, $c_{7,11} = 2$, $c_{8,12} = 9$, $c_{9,10} = 5$, $c_{10,7} = 6$, $c_{10,11} = 4$, $c_{11,12} = 9$ e i costi sono $d_{1,2} = 5$, $d_{1,5} = 2$, $d_{2,3} = 10$, $d_{2,6} = 3$, $d_{3,4} = 6$, $d_{4,8} = 9$, $d_{5,6} = 3$, $d_{5,9} = 3$, $d_{6,3} = 8$, $d_{6,10} = 1$, $d_{6,7} = 1$, $d_{7,8} = 4$, $d_{7,11} = 2$, $d_{8,12} = 5$, $d_{9,10} = 6$, $d_{10,7} = 4$, $d_{10,11} = 8$, $d_{11,12} = 2$.

La soluzione di base corrente sia data dall'albero rappresentato in figura 8.39 (in figura sono indicati anche i potenziali, i costi e le etichette dei nodi). Si noti che il flusso corrispondente a questa base è univocamente individuato dalla struttura dell'albero ed è un flusso pari a 6 unità sugli archi (1,2), (2,3), (3,4), (4,8) e (8,12) e 4 unità sugli archi (1,5), (5,9),

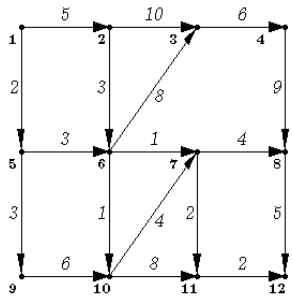


FIGURA 8.38

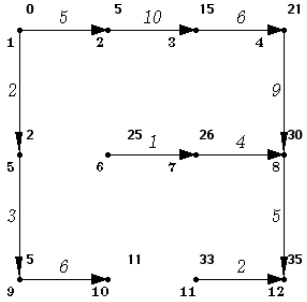


FIGURA 8.39

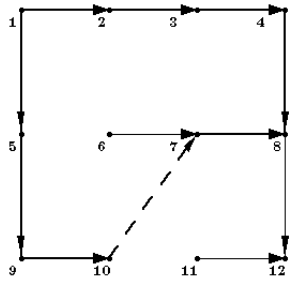


FIGURA 8.40

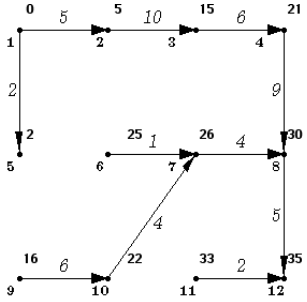


FIGURA 8.41

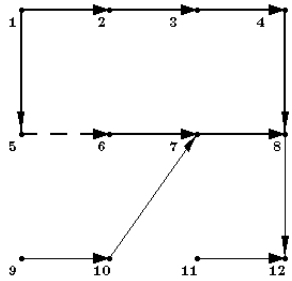


FIGURA 8.42

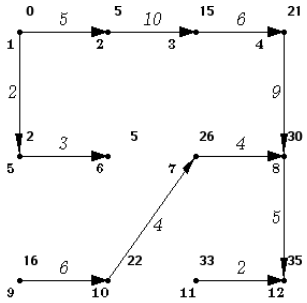


FIGURA 8.43

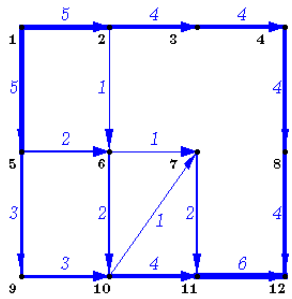


FIGURA 8.44

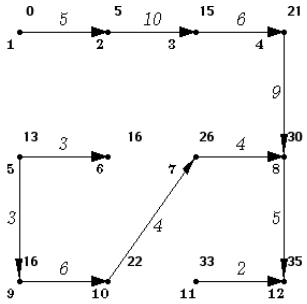


FIGURA 8.45

(9,10), (10,11) e (11,12). La base è fortemente ammissibile. Le liste sono

$$\begin{aligned} \mathit{pred} &= \{1,1,2,3,1,7,8,4,5,9,12,8\} \\ \mathit{depth} &= \{0,1,2,3,1,6,5,4,2,3,6,5\} \\ \mathit{thread} &= \{5,3,4,8,9,1,6,12,10,2,7,11\} \end{aligned}$$

La soluzione non è ottima. Gli archi che possono entrare in base sono (2,6), (5,6), (7,11) oppure (10,7), con costi ridotti (in valore assoluto) rispettivamente di 3, 3, 2 e 4. Scegliendo l'arco (10,7) si genera il ciclo rappresentato in figura 8.40. L'iterazione è degenere e nessuna unità di flusso può essere fatta circolare nel circuito a causa dell'unico arco bloccante (5,9). Il nuovo albero è rappresentato in figura 8.41. le liste sono:

$$\begin{aligned} \mathit{pred} &= \{1,1,2,3,1,7,8,4,10,7,12,8\} \\ \mathit{depth} &= \{0,1,2,3,1,6,5,4,7,6,6,5\} \\ \mathit{thread} &= \{5,3,4,8,2,1,10,12,6,9,7,11\} \end{aligned}$$

Anche questa soluzione non è ottima. Gli archi che possono entrare in base sono (2,6), (5,6) oppure (7,11) con i medesimi costi ridotti della precedente iterazione. Scegliendo l'arco (5,6) si genera il ciclo rappresentato in figura 8.42. L'iterazione non è degenere e una unità di flusso può circolare nel circuito con i due archi bloccanti (1,5) e (6,7). La regola anticiclaggio sceglie l'arco (6,7). Il nuovo albero è rappresentato in figura 8.43.

Dopo altre 7 iterazioni di cui 3 degeneri (queste cifre possono variare a seconda della scelta degli archi da far entrare in base) si ottiene la soluzione ottima il cui flusso si trova in figura 8.44 e l'albero di base in figura 8.45.

8.13. Massimo flusso e cammino minimo rivisitati

Il problema del massimo flusso si può porre facilmente nella forma di problema di flusso di costo minimo. Basta introdurre un'arco e_0 senza vincoli di capacità dalla sorgente al pozzo e risolvere il seguente problema:

$$\begin{aligned} \min \quad & x_0 \\ & (\, a_0 \quad A\,) \begin{pmatrix} x_0 \\ x \end{pmatrix} = 0 \\ & c^- \le x \le c^+ \end{aligned}$$

dove A è la matrice d'incidenza della rete originaria.

Sia la sorgente la radice dell'albero di base. Inoltre l'arco e_0, non avendo vincoli di capacità, deve essere sempre contenuto in $G(B)$. Questo implica a sua volta che il sottografo $G(B)\backslash e_0$ è costituito da due alberi disgiunti connessi rispettivamente alla sorgente e al pozzo. Ogni base individua quindi, tramite i due alberi, una partizione $[S, N\backslash S]$ dei nodi della rete.

Il calcolo delle variabili duali è immediato, infatti si ha $u_i = 0$, se $i \in S$, e $u_i = 1$, se $i \notin S$. Ne risulta che gli archi candidati ad entrare in base sono gli archi del taglio $Q(S)$, in quanto i coefficienti di costo sono nulli per tutti gli archi originari (e_0 deve rimanere sempre in base) e la tensione sugli archi è esattamente uguale al vettore incidenza $e(S)$ del taglio $Q(S)$.

La condizione di ottimalità è pertanto

$$x_e = c_e^+ \quad \text{se} \quad e \in Q^+(S) \qquad \text{e} \qquad x_e = c_e^- \quad \text{se} \quad e \in Q^-(S)$$

e fornisce una dimostrazione alternativa del teorema del massimo flusso-minima capacità.

Se la condizione di ottimalità non è verificata ci deve essere almeno un arco del taglio che sia 'saturo' ($x_j = c_j^+$) e diretto dal pozzo alla sorgente oppure 'svuotato' ($x_j = c_j^-$) e diretto dalla sorgente al pozzo. Inserendo tale arco in $G(B)\backslash e_0$ si crea un cammino orientato dalla sorgente al pozzo sul quale si può aumentare il flusso, a meno di degenerazione, fino a che un qualsiasi arco del cammino viene o saturato o svuotato. Questo arco viene rimosso dalla base e un nuovo taglio viene così generato per la successiva iterazione.

È abbastanza frequente il caso in cui $c_j^- = 0$ oppure $c_j^+ = 0$. In questo caso è immediatamente disponibile una soluzione iniziale di base data dal flusso nullo. Se ad esempio $c_j^+ > c_j^- = 0$ una base iniziale fortemente ammissibile è data da $S = \{s\}$ e l'albero collegato al pozzo con gli archi tutti diretti verso il pozzo.

Come si vede, l'algoritmo del simplesso è in questo caso un particolare tipo di algoritmo aumentante in quanto il ciclo su cui far circolare del flusso si riduce di fatto ad un cammino dalla sorgente al pozzo. La differenza è che le iterazioni possono essere degeneri per cui si possono scegliere dei cammini aumentanti senza un reale aumento di flusso. Si noti che in base alla discussione della sezione precedente vi possono essere al più n iterazioni degeneri consecutive. Si può provare questo fatto anche semplicemente osservando che la regola anticiclaggio obbliga a scegliere l'arco bloccante più vicino al pozzo e quindi l'insieme S cresce ad ogni iterazione degenere. Se non si specifica nessuna regola particolare per la scelta dell'arco entrante in base, una limitazione superiore al numero di iterazioni non degeneri è $n\,\hat{c}$ con $\hat{c}$ il massimo valore assoluto fra tutti i valori di capacità (si pensi ad esempio al taglio indotto dalla sola sorgente). Quindi il metodo ha una complessità soltanto pseudopolinomiale $O(n^2\,\hat{c})$. Goldfarb e Hao [1990] hanno dimostrato che una particolare regola di selezione dell'arco entrante in base richiede $O(n\,m)$ iterazioni e l'algoritmo richiede globalmente un tempo $O(n^2\,m)$. Un'implementazione dovuta a Goldberg et al. [1988] richiede addirittura tempo $O(n\,m\,\log n)$

Per formulare il problema del cammino minimo mediante la programmazione lineare è più conveniente, per motivi di degenerazione, affrontare il problema di minimizzare tutti i cammini da un nodo sorgente s a tutti gli altri nodi, piuttosto che il problema di minimizzare un particolare cammino $s \to t$. Quindi $b_s := n - 1$ e $b_i := -1\ \forall i \neq s$. Se l'intervallo di espansione di un arco $e = (i, j)$ è $[d_e^-, d_e^+]$, si deve sostituire all'arco una coppia di archi (i, j) e (j, i) di lunghezze $d_{ij} := d_e^+$ e $d_{ji} := -d_e^-$ rispettivamente. Sul nuovo grafo si cerca allora un cammino orientato.

Data la struttura del problema conviene porre la radice dell'albero di supporto di base nella sorgente. È immediato verificare che le variabili duali sull'albero rappresentano la lunghezza del percorso dalla sorgente al nodo in questione lungo i rami dell'albero e che quindi l'albero stesso rappresenta l'insieme di tutti i percorsi.

La condizione di ottimalità, $v_e \leq d_e \quad \forall e \in E$, ha un'interpretazione molto semplice: se $e = (h, k)$ e $v_e = u_k - u_h$, con u_k uguale alla distanza da s a k lungo l'albero di base e analogamente per u_h, allora la condizione di ottimalità esprime il fatto che non è più conveniente raggiungere il nodo k andando dapprima fino al nodo h lungo l'albero e poi al nodo k lungo l'arco $e = (h, k)$. Se invece la condizione di ottimalità non è verificata un qualsiasi arco per cui $v_e > d_e$ può entrare in base a spese di un altro arco, che, come è facile verificare, risulta essere l'ultimo arco nel percorso dalla sorgente a k sull'albero.

La scelta dell'arco da far entrare in base corrispondente al primo arco di costo ridotto negativo in una scansione circolare degli archi porta ad un algoritmo fortemente polinomiale.

Infatti dopo la k-esima scansione sono stati trovati tutti i cammini minimi di al più k archi. Quindi n scansioni sono sufficienti. Ogni scansione comporta l'esame di al più m archi e ogni iterazione ha un costo di $O(n)$. Quindi in totale la complessità è $O(n^2\, m)$. Questo valore può essere abbassato a $O(n^3)$.

8.14. Reti multiflusso

In tutti i problemi considerati finora il flusso circolante nella rete è stato di un unico tipo. Quindi il flusso in arrivo da due sorgenti si può miscelare e poi suddividere a seconda delle esigenze verso destinazioni diverse. Si consideri ad esempio la rete in figura 8.46 (a). Del flusso deve essere inviato dalle due sorgenti s_1 e s_2 alle destinazioni t_1 e t_2. Se il flusso in partenza dalle due sorgenti è dello stesso tipo, allora non ha importanza se il flusso in partenza da s_1 finisce in t_1 o in t_2, e quindi una soluzione del tipo di quella indicata in figura 8.46 (b) è ammissibile. Se però i flussi dalle due sorgenti rappresentano beni diversi, è essenziale poter identificare e distinguere ciò che ha origine in s_1 da ciò che ha origine in s_2. Pertanto se un flusso deve partire da s_1 e arrivare a t_1 ed un flusso diverso deve partire da s_2 e arrivare a t_2, la soluzione è come quella in figura 8.46 (c), dove i due flussi condividono un arco senza annullarsi.

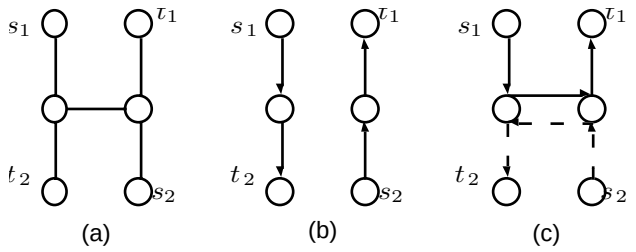


FIGURA 8.46

Reti in cui sono presenti flussi di tipo diverso vengono dette *reti multiflusso* oppure *reti di flusso a molti beni*, (*Multi-commodity flows*). Il modo in cui flussi diversi interagiscono fra di loro avviene tramite le capacità degli archi, dato che sfruttano tutti la stessa capacità d'arco. Tuttavia va precisato meglio cosa si intenda per uso comune della medesima capacità. Infatti, se è presente un unico flusso, il fatto che il valore assoluto del flusso non ecceda il valore c qualunque sia la direzione del flusso lungo l'arco si può modellare semplicemente con un intervallo di capacità $[-c, c]$.

Se sono presenti $p > 1$ tipi di flusso e solo flusso non negativo può circolare negli archi, non ci sono particolari problemi. Basta porre il vincolo $\sum_k x_e(k) \leq c_e$, dove $x_e(k)$ è il flusso di tipo k circolante nell'arco e di capacità c_e.

Se però sono presenti più flussi, che possono circolare negli archi in modo arbitrario, allora il vincolo che si deve considerare è $\sum_k |x_e(k)| \leq c_e$. Questo vincolo può essere facilmente espresso senza il valore assoluto definendo $x_e(k) = x_e^+(k) - x_e^-(k)$ con $x_e^+(k) \geq 0$ e $x_e^-(k) \geq 0$ e vincolando $\sum_k x_e^+(k) + x_e^-(k) \leq c_e$. Questo è equivalente a duplicare ogni arco nella rete con una coppia di archi antiparalleli con vincolo di flusso non negativo. Tuttavia il nuovo vincolo di capacità $\sum_k x_e^+(k) + x_e^-(k) \leq c_e$ è di tipo diverso da quelli finora considerati dato che vincola fra loro flussi di archi diversi (i due archi antiparalleli). Il problema comunque

rientra nell'ambito della programmazione lineare e si può esprimere nel seguente modo.

$$\begin{aligned} \min \ & \sum_k \sum_e d_e(k)\, x_e(k) \\ & A\,(x^+(k) - x^-(k)) = b(k) \qquad \forall k \\ & \sum_k x_e^+(k) + x_e^-(k) \le c_e \qquad \forall e \\ & x_e^+(k) \ge 0,\ x_e^-(k) \ge 0 \qquad \forall k,\, \forall e \end{aligned} \tag{8.17}$$

dove A è, al solito, la matrice d'incidenza nodi-archi della rete (la medesima per ogni flusso). Il problema (8.17) ha però un numero eccessivo di vincoli ($p\,m + m$) e variabili ($2\,n\,m$) per essere affrontato direttamente.

Risulta più agevole affrontare (8.17) decomponendo il problema come nell'esempio 7.38 e quindi risolvendo un problema di programmazione lineare con m righe. Per generare una colonna bisogna risolvere un problema di flusso a costo minimo per ogni tipo di flusso.

Un diverso approccio, sempre a generazione di colonne, si basa sulle idee dell'esercizio 7.40. Sia $\mathcal{P}^k$ l'insieme dei cammini orientati dalla sorgente s_k alla destinazione t_k (supponiamo per semplicità di notazione che per ogni tipo di flusso vi sia una sola sorgente ed una sola destinazione e la quantità richiesta di flusso da s_k sia b_k). Sia $P \in \mathcal{P}^k$ un cammino e sia $x_P \ge 0$ un flusso costante lungo il cammino P. Per ogni P si può definire il costo $d_P := \sum_{e \in P} d_e$. Allora (8.17) può essere risolto da

$$\begin{aligned} \min \ & \sum_k \sum_{P \in \mathcal{P}^k} d_P\, x_P \\ & \sum_{P \in \mathcal{P}^k} x_P = b_k \qquad \forall k \\ - & \sum_k \sum_{\substack{P \in \mathcal{P}^k \\ P \ni e^+ \cup e^-}} x_P \ge -c_e \qquad \forall e \\ & x_P \ge 0 \end{aligned} \tag{8.18}$$

dove una colonna viene generata se $d_P - w_k + \sum_{e \in P} u_e < 0$ dove w_k sono le variabili duali dei vincoli sulle sorgenti e $u_e \ge 0$ sono le variabili duali sui vincoli di capacità. Quindi bisogna risolvere un problema di cammino minimo con lunghezze sugli archi $d_e + u_e$. Se la lunghezza del cammino minimo è inferiore a w_k, la colonna relativa a tale cammino viene generata.

La formulazione (8.18) richiede $m + p$ righe. Quindi il calcolo delle variabili duali e l'aggiornamento dell'inversa della matrice di base è più oneroso che con il metodo di decomposizione (esempio 7.38). Però la generazione di una colonna richiede il calcolo di un cammino minimo, cosa che è molto più rapida del calcolo di un flusso a costo minimo.

È importante tener presente che in una rete multiflusso la proprietà di totale unimodularità viene persa e aggiungere il vincolo d'interezza rende un problema **NP**-difficile.