

L'algoritmo di Edmonds-Karp

Luca Foschiani

Università degli studi di Udine

Advanced Algorithms

- 1 Il problema del massimo flusso
- 2 Il metodo di Ford-Fulkerson
 - Rete residua
 - Cammino aumentante
 - Taglio
- 3 L'algoritmo di Edmonds-Karp

Definizioni: rete di flusso

Una **rete di flusso** $G = (V, E)$ è un grafo orientato nel quale ad ogni arco $(u, v) \in E$ è assegnata una capacità non negativa $c(u, v) \geq 0$.

Inoltre, assumiamo che se esiste un arco $(u, v) \in E$, allora $(v, u) \notin E$.

Individuiamo due nodi s (**sorgente**) e t (**pozzo**). Per questi due nodi, abbiamo che per ogni nodo in V esiste almeno un cammino che lo contiene e che va da s a t (ogni nodo è raggiungibile da s e raggiunge t).

Definizioni: flusso

Un **flusso** in G è una funzione $f : V \times V \rightarrow \mathbb{R}$ che soddisfa le seguenti proprietà:

- Per ogni $u, v \in V$ richiediamo $0 \leq f(u, v) \leq c(u, v)$ (vincolo di capacità)
- Per ogni $u \in V \setminus \{s, t\}$ richiediamo $\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$
(conservazione del flusso)

La quantità $f(u, v)$ viene chiamata flusso dal nodo u al nodo v .

Il **valore** $|f|$ di un flusso f è definito nel seguente modo:

$$|f| = \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s)$$

Obiettivo del **problema del massimo flusso** è, data una rete di flusso, trovare il flusso di valore massimo.

- più sorgenti/più pozzi?
- ammettere archi (u, v) con $(v, u) \in E$?
- applicazioni? (es. matching bipartito di cardinalità massima)

Concetti:

- Rete residua
- Cammino aumentante
- Taglio

Per poi arrivare al **Teorema del flusso massimo e taglio minimo**
(permette di dimostrare che l'algoritmo trova sempre il flusso massimo).

L'algoritmo di Ford-Fulkerson

Rete residua

Dati una rete G e un flusso f , la rete residua G_f rappresenta in che modo è possibile cambiare il flusso in G .

Ponendo $G = (V, E)$, s sorgente e t pozzo, Possiamo definire la **capacità residua** c_f nel seguente modo:

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{se } (u, v) \in E \\ f(v, u) & \text{se } (v, u) \in E \\ 0 & \text{altrimenti} \end{cases}$$

Il primo caso della definizione corrisponde alla “capacità residua” degli archi presenti in G .

Il secondo caso permette all'algoritmo di “vedere” le quantità di flusso già assegnate agli archi, dando la possibilità di diminuire il flusso su un arco (assegnando al corrispondente arco in G_f un flusso non nullo).

In G_f avrò gli stessi nodi presenti in G , mentre gli archi saranno tutti gli (u, v) tali che $c_f(u, v) > 0$.

Il metodo di Ford-Fulkerson

Rete residua

Idea: un flusso individuato nella rete residua permette di aumentare il flusso nella rete originale.

Dati f flusso in G e f' flusso in G_f , definisco la funzione $(f \uparrow f') : V \times V \rightarrow \mathbb{R}$ nel seguente modo:

$$(f \uparrow f')(u, v) = \begin{cases} f(u, v) + f'(u, v) - f'(v, u) & \text{se } (u, v) \in E \\ 0 & \text{altrimenti} \end{cases}$$

Lemma

La funzione $f \uparrow f'$ è un flusso in G avente valore $|f \uparrow f'| = |f| + |f'|$.

Il metodo di Ford-Fulkerson

Cammino aumentante

Data una rete di flusso G e un flusso f , un **cammino aumentante** è un cammino semplice che va da s a t nella rete residua G_f .

Dato un cammino aumentante p , chiamiamo **capacità residua** di p la massima quantità di flusso che è possibile mandare su questo cammino:

$$c_f(p) = \min\{c_f(u, v) : (u, v) \text{ sta in } p\}.$$

Definiamo inoltre una funzione $f_p : V \times V \rightarrow \mathbb{R}$ nel seguente modo:

$$f_p(u, v) = \begin{cases} c_f(p) & \text{se } (u, v) \text{ sta in } p \\ 0 & \text{altrimenti} \end{cases}$$

Lemma

f_p è un flusso in G_f di valore $|f_p| = c_f(p) > 0$.

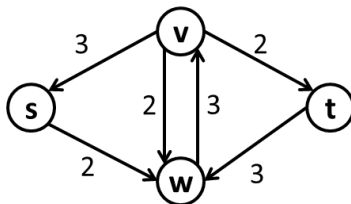
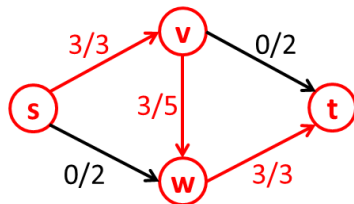
Corollario

$f \uparrow f_p$ è un flusso in G di valore $|f \uparrow f_p| = |f| + |f_p| > |f|$.

Il metodo di Ford-Fulkerson

Esempio rete residua

Rete di flusso (flusso/capacità) e corrispondente rete residua:



Il metodo di Ford-Fulkerson

Taglio

Un **taglio** (S, T) di una rete di flusso $G = (V, E)$ è una partizione di V in S e T tale che $s \in S$ e $t \in T$.

Dato f flusso, esprimiamo con $f(S, T)$ il **flusso che attraversa il taglio**:

$$f(S, T) = \sum_{u \in S} \sum_{v \in T} f(u, v) - \sum_{u \in S} \sum_{v \in T} f(v, u)$$

La **capacità di un taglio** (S, T) è:

$$c(S, T) = \sum_{u \in S} \sum_{v \in T} c(u, v)$$

Il **taglio minimo** è il taglio di capacità minima fra i tagli della rete.

Il metodo di Ford-Fulkerson

Taglio

Lemma

Dati f flusso in G con s sorgente e t pozzo, e (S, T) un qualsiasi taglio di G , il flusso attraverso (S, T) è $f(S, T) = |f|$.

Corollario

Il valore di qualsiasi flusso f su una rete G è limitato superiormente dalla capacità di un qualsiasi taglio di G .

Teorema del flusso massimo e taglio minimo

Dati f flusso in una rete $G = (V, E)$ con s sorgente e t pozzo, le seguenti condizioni sono equivalenti:

- f è un flusso massimo in G
- La rete residua G_f non ammette cammini aumentanti
- $|f| = c(S, T)$ per qualche taglio (S, T) di G

Il metodo di Ford-Fulkerson

Pseudocodice

Algorithm 1 Ford-Fulkerson(G, s, t)

```
1: for each edge  $(u, v) \in E$ 
2:    $(u, v).f = 0$ 
3: while esiste un cammino aumentante  $p$  in  $G_f$ 
4:    $c_f(p) = \min \{c_f(u, v) : (u, v) \text{ sta in } p\}$ 
5:   for each edge  $(u, v)$  in  $p$ 
6:     if  $(u, v) \in E$ 
7:        $(u, v).f = (u, v).f + c_f(p)$ 
8:     else
9:        $(v, u).f = (v, u).f - c_f(p)$ 
```

Il metodo di Ford-Fulkerson

Complessità

Una possibile implementazione del metodo consiste nello scegliere il cammino aumentante alla linea 3 in maniera arbitraria: l'algoritmo che si ottiene termina sempre a patto che le capacità degli archi siano valori razionali.

Per il calcolo della complessità, possiamo ridurci al caso in cui le capacità prendono valori interi.

Il ciclo **for** alle linee 1-2 ha complessità $O(|E|)$.

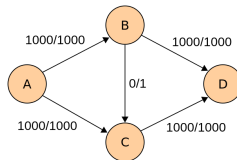
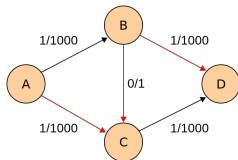
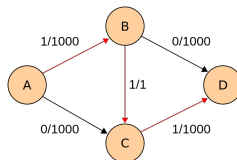
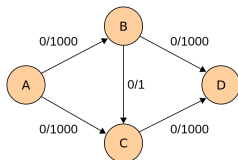
Se chiamiamo f^* il flusso massimo, abbiamo che il ciclo **while** alla linea 3 verrà eseguito al più $|f^*|$ volte (ad ogni iterazione il valore del flusso aumenterà almeno di una unità). Trovare un cammino nella rete residua costa $O(|V| + |E|) = O(|E|)$ con BFS o DFS.

La complessità dell'algoritmo è quindi $O(|E| \cdot |f^*|)$. L'algoritmo è pseudopolinomiale.

Il metodo di Ford-Fulkerson

Caso pessimo

Le prime tre immagini mostrano il primo passo dell'algoritmo e la successiva selezione del cammino aumentante. L'ultima immagine mostra lo stato al quale si arriva dopo 2000 iterazioni.



L'algoritmo di Edmonds-Karp

L'algoritmo di Edmonds-Karp è un'implementazione del metodo di Ford-Fulkerson che ci permette di risolvere il problema del massimo flusso in tempo $O(|V| \cdot |E|^2)$. L'algoritmo utilizza una ricerca in ampiezza per trovare, ad ogni iterazione, il **cammino minimo** da s a t nella rete residua.

Lemma

Dato $G = (V, E)$ con s sorgente e t pozzo, per tutti i $v \in V \setminus \{s, t\}$ abbiamo che la lunghezza del cammino minimo fra s e v nella rete residua G_f aumenta in maniera monotona durante l'esecuzione dell'algoritmo (in particolare, ad ogni aumento del flusso).

Teorema

Data una rete di flusso $G = (V, E)$, l'algoritmo di Edmonds-Karp aumenta il valore del flusso $O(|V| \cdot |E|)$ volte.

Come visto precedentemente, possiamo individuare un cammino aumentante utilizzando BFS in tempo $O(|E|)$. La complessità dell'algoritmo è quindi $O(|V| \cdot |E|^2)$.

Algoritmi basati su ford-fulkerson di complessità minore (push relabel o dinic)