Il Problema del Flusso Massimo

Algoritmi Classici, Sviluppi Recenti
e Applicazioni alla Segmentazione di Immagini Mediche

Ivan Buttignon

Scuola Ortogonale — II Ciclo

Agenda

1. Overview generale: definizioni e teorema Max-Flow Min-Cut
2. Panoramica storica: 70 anni di algoritmi
3. Algoritmi classici: Ford-Fulkerson & Edmonds-Karp
4. Chen et al. 2022: il risultato quasi-lineare
5. Applicazione: segmentazione di immagini mediche

Origini: Guerra Fredda e reti ferroviarie

Origini: Guerra Fredda e reti ferroviarie

Nel **1955**, Harris e Ross della **RAND Corporation** modellarono la rete ferroviaria sovietica come un grafo per stimare la capacità di trasporto verso l'Europa orientale.

Origini: Guerra Fredda e reti ferroviarie

Nel **1955**, Harris e Ross della **RAND Corporation** modellarono la rete ferroviaria sovietica come un grafo per stimare la capacità di trasporto verso l'Europa orientale.

Il rapporto (declassificato nel 1999) conteneva un grafo con **44 vertici** e **105 archi** e affrontava un duplice problema:

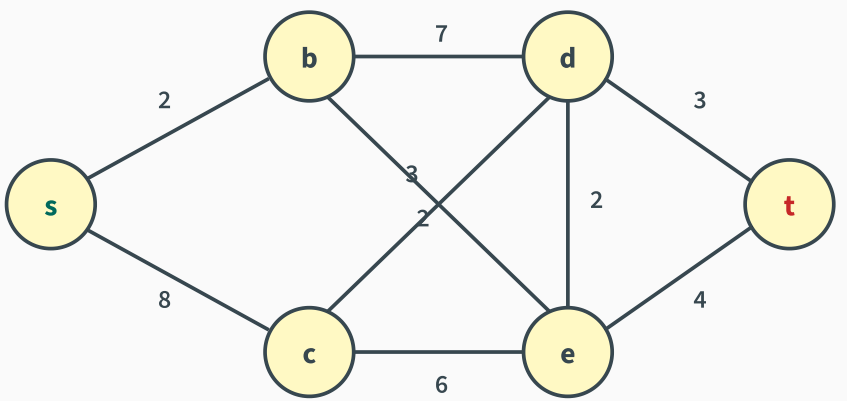
- Quale **flusso massimo** può attraversare la rete?
- Quale **taglio minimo** la interrompe con il minor costo?

STEP 1 — OVERVIEW GENERALE

Rete di flusso

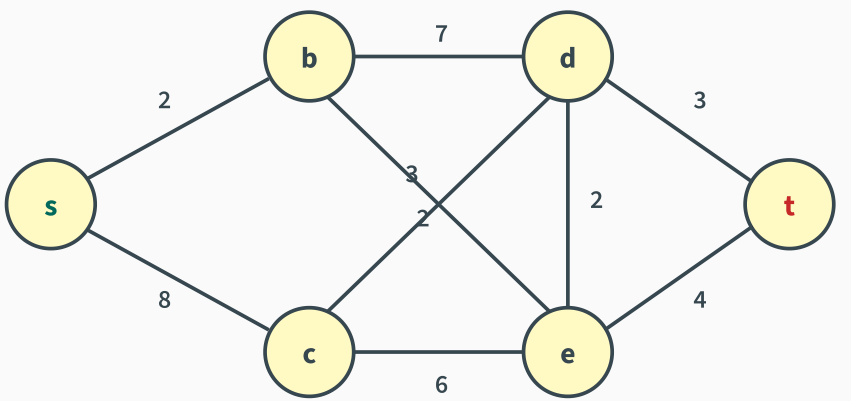
STEP 1 — OVERVIEW GENERALE

Rete di flusso



STEP 1 — OVERVIEW GENERALE

Rete di flusso



Definizione — Rete di flusso

Una rete di flusso $G = (V, E, s, t, c)$ dove (V, E) è un grafo orientato, s è la **sorgente**, t è il **pozzo**, e $c\colon V \times V \to \mathbb{R}_{\geq 0}$ è la funzione di capacità.

"Una rete di tubi con capacità massime, un rubinetto (s) e uno scarico (t)."

STEP 1 — OVERVIEW GENERALE

Flusso: le tre proprietà

STEP 1 — OVERVIEW GENERALE

Flusso: le tre proprietà

1. Vincolo di capacità

$\forall\, u, v \in V : \;\; f(u,v) \leq c(u,v)$

“Non si può mandare più acqua di quella che il tubo sopporta.”

STEP 1 — OVERVIEW GENERALE

Flusso: le tre proprietà

1. Vincolo di capacità

$\forall\, u, v \in V : \;\; f(u,v) \leq c(u,v)$

“Non si può mandare più acqua di quella che il tubo sopporta.”

2. Antisimmetria

$\forall\, u, v \in V : \;\; f(u,v) = -f(v,u)$

“5 unità da a a b significano −5 da b ad a.”

STEP 1 — OVERVIEW GENERALE

Flusso: le tre proprietà

1. Vincolo di capacità

$\forall\, u, v \in V : \;\; f(u,v) \leq c(u,v)$

“Non si può mandare più acqua di quella che il tubo sopporta.”

2. Antisimmetria

$\forall\, u, v \in V : \;\; f(u,v) = -f(v,u)$

“5 unità da a a b significano −5 da b ad a.”

3. Conservazione del flusso

$\forall\, x \in V \setminus \{s,t\} : \sum_{y \in V} f(x,y) = 0$

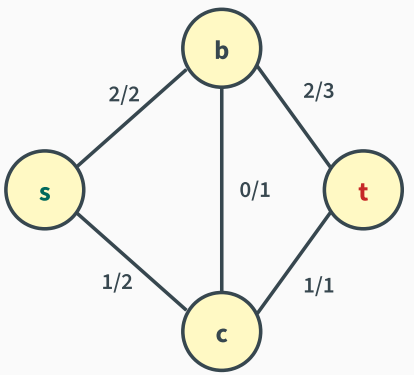
“Nessun nodo interno accumula o genera flusso — tutto ciò che entra deve uscire.”

STEP 1 — OVERVIEW GENERALE

Rete residua

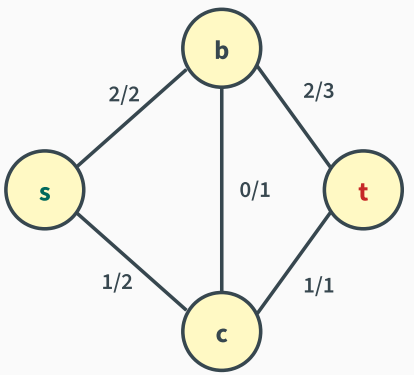
STEP 1 — OVERVIEW GENERALE

Rete residua

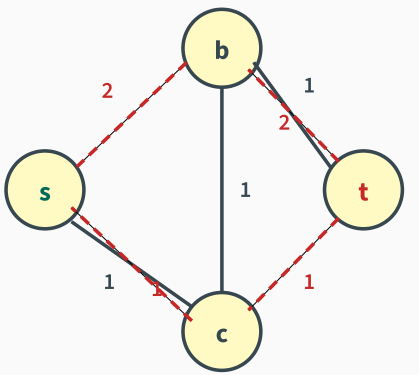


(a) Flusso $|f| = 3$

Rete residua

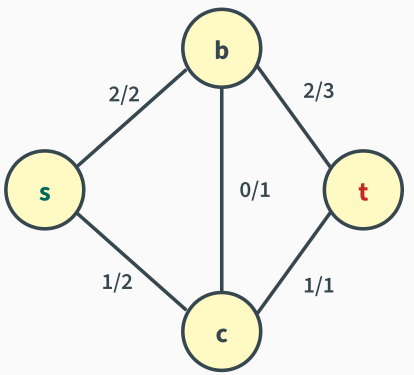


(a) Flusso $|f| = 3$

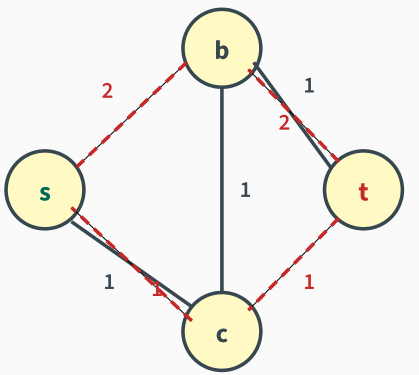


(b) Rete residua G_f

Rete residua



(a) Flusso $|f| = 3$



(b) Rete residua G_f

Capacità residua

$c_f(u, v) = c(u, v) - f(u, v)$ — "quanto spazio resta in ogni arco"
Gli **archi inversi tratteggiati** permettono di "annullare" flusso già mandato.

STEP 1 — OVERVIEW GENERALE

Cammino aumentante

STEP 1 — OVERVIEW GENERALE

Cammino aumentante

Definizione

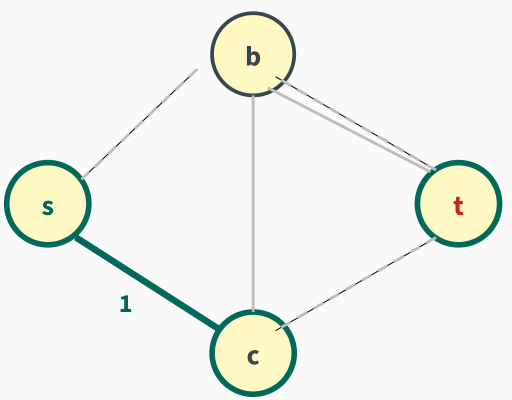
Un **cammino aumentante** è un cammino da s a t nella rete residua G_f.
Capacità (bottleneck): $c_f(P) = \min_{(u,v)\in P} c_f(u,v)$

STEP 1 — OVERVIEW GENERALE

Cammino aumentante

Definizione

Un **cammino aumentante** è un cammino da s a t nella rete residua G_f.
Capacità (bottleneck): $c_f(P) = \min_{(u,v)\in P} c_f(u,v)$

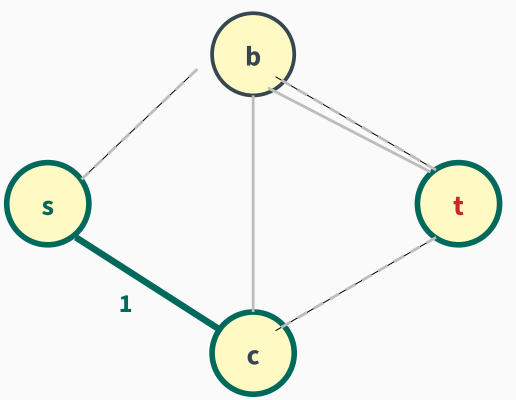


STEP 1 — OVERVIEW GENERALE

Cammino aumentante

Definizione

Un **cammino aumentante** è un cammino da s a t nella rete residua G_f.
Capacità (bottleneck): $c_f(P) = \min_{(u,v)\in P} c_f(u,v)$



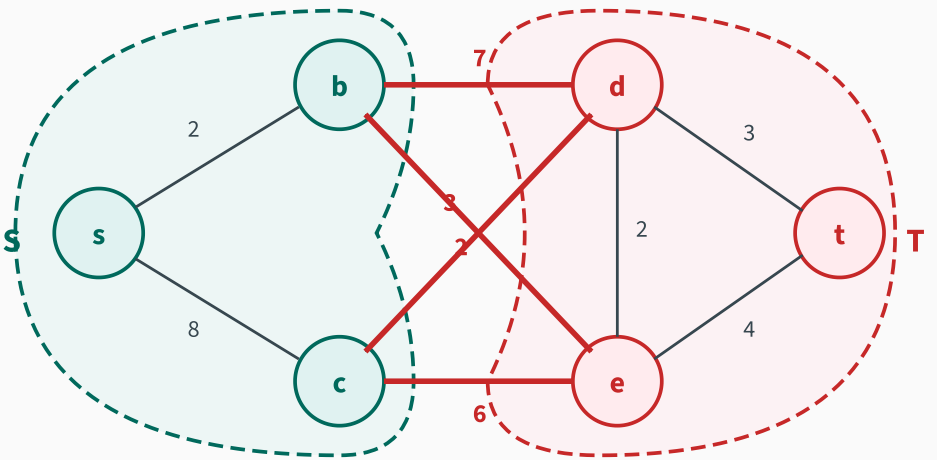
"Se esiste un cammino nella rete residua, possiamo ancora migliorare il flusso. Il cammino aumentante è il mattone fondamentale: tutti gli algoritmi che vedremo cercano questi cammini."

STEP 1 — OVERVIEW GENERALE

Taglio e capacità del taglio

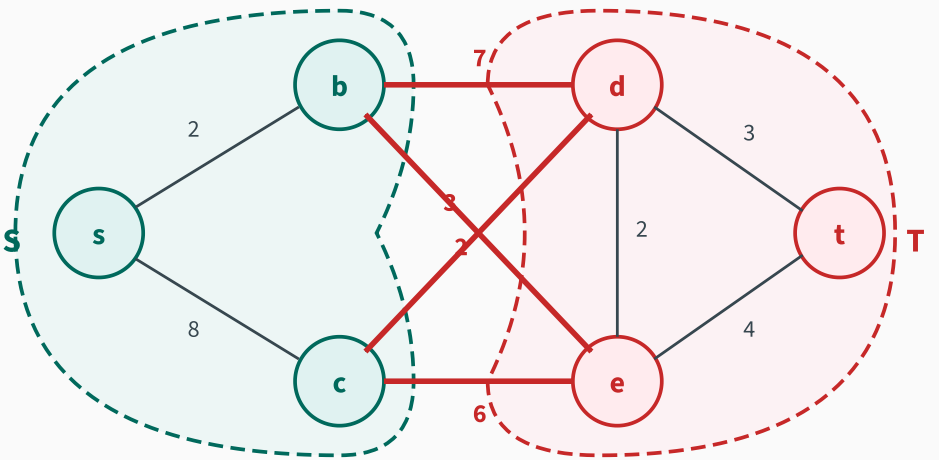
STEP 1 — OVERVIEW GENERALE

Taglio e capacità del taglio



STEP 1 — OVERVIEW GENERALE

Taglio e capacità del taglio



Taglio (S, T)

Partizione di V con $s \in S$ e $t \in T$. **Capacità:** $\|S, T\| = \sum_{u \in S} \sum_{v \in T} c(u, v) = 7 + 2 + 3 + 6 = 18$

“Si sommano solo le capacità degli archi che vanno da S a T, non viceversa.”

STEP 1 — OVERVIEW GENERALE

Teorema Max-Flow Min-Cut

Tre condizioni equivalenti:

STEP 1 — OVERVIEW GENERALE

Teorema Max-Flow Min-Cut

Teorema (Ford-Fulkerson, 1956)

$$\max_{f} |f| \;=\; \min_{(S,T)} \|S,T\|$$

"Il collo di bottiglia della rete — il taglio più debole — determina esattamente il flusso massimo."

Tre condizioni equivalenti:

STEP 1 — OVERVIEW GENERALE

Teorema Max-Flow Min-Cut

Teorema (Ford-Fulkerson, 1956)

$$\max_{f} |f| \;=\; \min_{(S,T)} \|S,T\|$$

"Il collo di bottiglia della rete — il taglio più debole — determina esattamente il flusso massimo."

Tre condizioni equivalenti:

1. f è un **flusso massimo** in G

STEP 1 — OVERVIEW GENERALE

Teorema Max-Flow Min-Cut

Teorema (Ford-Fulkerson, 1956)

$$\max_{f} |f| \;=\; \min_{(S,T)} \|S,T\|$$

"Il collo di bottiglia della rete — il taglio più debole — determina esattamente il flusso massimo."

Tre condizioni equivalenti:

1. f è un **flusso massimo** in G
2. La rete residua G_f **non contiene cammini aumentanti** da s a t

STEP 1 — OVERVIEW GENERALE

Teorema Max-Flow Min-Cut

Teorema (Ford-Fulkerson, 1956)

$$\max_{f} |f| \;=\; \min_{(S,T)} \|S,T\|$$

"Il collo di bottiglia della rete — il taglio più debole — determina esattamente il flusso massimo."

Tre condizioni equivalenti:

1. f è un **flusso massimo** in G
2. La rete residua G_f **non contiene cammini aumentanti** da s a t
3. $|f| = \|S,T\|$ per qualche taglio (S,T)

STEP 2 — PANORAMICA STORICA

Evoluzione storica degli algoritmi

Algoritmo	Anno	Complessità	Tipo
Ford-Fulkerson	1956	$O(E \lvert f^* \rvert)$	Debolmente polin.
Dinitz	1970	$O(V^2 E)$	Fortemente polin.
Edmonds-Karp	1972	$O(V E^2)$	Fortemente polin.
Push-Relabel	1988	$O(V^2 \sqrt{E})$	Fortemente polin.
Orlin	2013	$O(VE)$	Fortemente polin.
Chen et al.	**2022**	$E^{1+o(1)}$	**Debolmente polin.**

Da pseudo-polinomiale (anni ’50) a quasi-lineare (2022) — 70 anni di progresso algoritmico.

Ford-Fulkerson: il metodo

Ford-Fulkerson: il metodo

```
Ford-Fulkerson(G, s, t, c)
  f(u,v) ← 0   per ogni (u,v)
  while esiste cammino P da s a t in G_f:
    F ← min_{(u,v)∈P} c_f(u,v)    // bottleneck
    for each (u,v) ∈ P:
      f(u,v) ← f(u,v) + F
      f(v,u) ← f(v,u) − F
  return f
```

STEP 3 — ALGORITMI CLASSICI

Ford-Fulkerson: il metodo

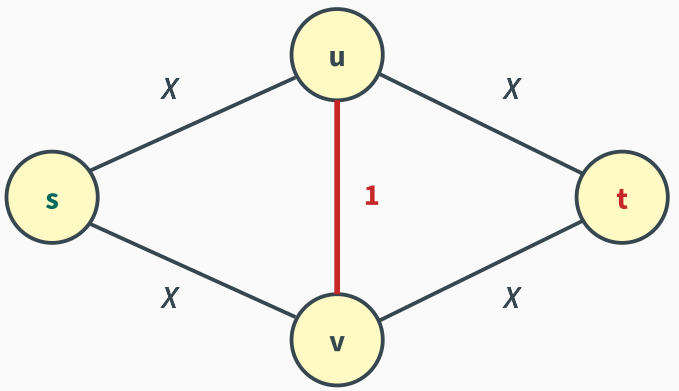
```
Ford-Fulkerson(G, s, t, c)
  f(u,v) ← 0   per ogni (u,v)
  while esiste cammino P da s a t in G_f:
    F ← min_{(u,v)∈P} c_f(u,v)    // bottleneck
    for each (u,v) ∈ P:
      f(u,v) ← f(u,v) + F
      f(v,u) ← f(v,u) − F
  return f
```

Complessità: $O(E \cdot |f^*|)$ — pseudo-polinomiale

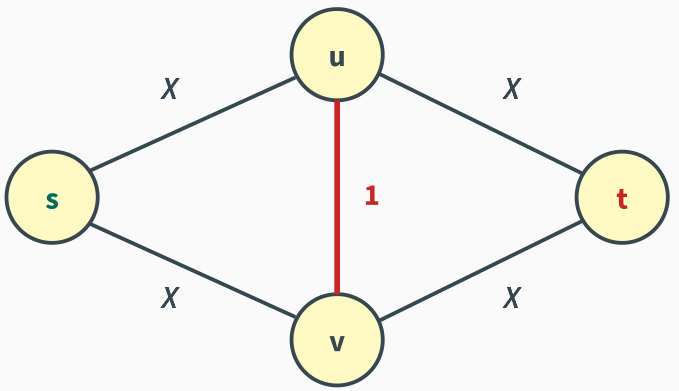
"Cerca un cammino, spingi flusso, ripeti. Semplice ma con una trappola: la complessità dipende dal VALORE del flusso, non dalla dimensione del grafo."

Ford-Fulkerson: il caso patologico

Ford-Fulkerson: il caso patologico

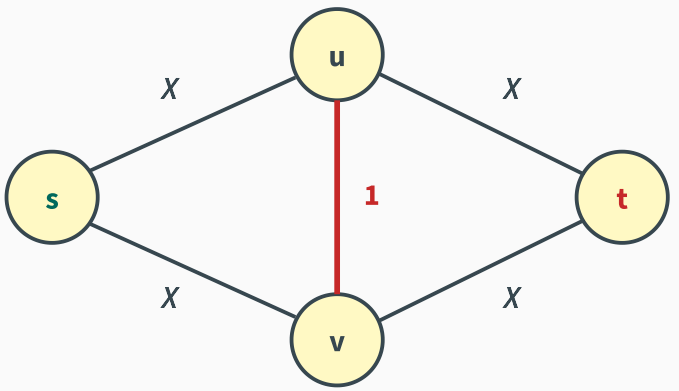


Ford-Fulkerson: il caso patologico



Flusso massimo = $2X$, ma con scelte avverse: fino a $2X$ **iterazioni**, ciascuna incrementa il flusso di sole **1 unità**.

Ford-Fulkerson: il caso patologico



Flusso massimo = $2X$, ma con scelte avverse: fino a $2X$ **iterazioni**, ciascuna incrementa il flusso di sole **1 unità**.

> ***La scelta del cammino conta!*** *Soluzione: usare **BFS** → Edmonds-Karp*

Edmonds-Karp: la correzione BFS

STEP 3 — ALGORITMI CLASSICI

Edmonds-Karp: la correzione BFS

Una sola modifica: usare **BFS** (cammino più corto in numero di archi) invece di DFS.

Edmonds-Karp: la correzione BFS

Una sola modifica: usare **BFS** (cammino più corto in numero di archi) invece di DFS.

Complessità

$$O(VE^2)$$

Fortemente polinomiale — indipendente dai valori delle capacità

Edmonds-Karp: la correzione BFS

Una sola modifica: usare **BFS** (cammino più corto in numero di archi) invece di DFS.

Complessità

$$O(VE^2)$$

Fortemente polinomiale — indipendente dai valori delle capacità

Intuizione dei lemmi chiave (senza dimostrazione):

- **Monotonia dei livelli BFS:** le distanze da s non diminuiscono mai
- **Bound sulle iterazioni:** ogni arco scompare dalla rete residua al più $V/2$ volte $\Rightarrow$ al più $VE/2$ iterazioni

Confronto: Ford-Fulkerson vs Edmonds-Karp

	Ford-Fulkerson	Edmonds-Karp
Ricerca	DFS / arbitraria	BFS (più corto)
Complessità	$O(E \lvert f^* \rvert)$	$O(VE^2)$
Tipo	Pseudo-polinomiale	Fortemente polinomiale
Terminazione	Solo cap. razionali	Sempre garantita
Dipendenza valori	Sì ($\lvert f^* \rvert$)	No

EK è un caso particolare di FF. Complessità effettiva: $O\big(\min(E \cdot \lvert f^* \rvert,\ VE^2)\big)$
Una piccola modifica, un grande impatto.

STEP 4 — CHEN ET AL. 2022

Chen et al.: il risultato rivoluzionario

STEP 4 — CHEN ET AL. 2022

Chen et al.: il risultato rivoluzionario

Risultato principale — Best Paper, FOCS 2022

$$E^{1+o(1)}$$

Flusso massimo esatto in tempo quasi-lineare nel numero di archi

STEP 4 — CHEN ET AL. 2022

Chen et al.: il risultato rivoluzionario

Risultato principale — Best Paper, FOCS 2022

$$E^{1+o(1)}$$

Flusso massimo esatto in tempo quasi-lineare nel numero di archi

Perché è rivoluzionario:

- La sola *lettura dell'input* richiede $\Omega(E) \Rightarrow$ il risultato è essenzialmente **ottimale**
- Il fattore $E^{o(1)}$ cresce più lentamente di qualsiasi potenza di E
- Barriera storica superata: da $O(VE)$ (Orlin, 2013) a quasi-lineare

STEP 4 — CHEN ET AL. 2022

Le due componenti chiave

STEP 4 — CHEN ET AL. 2022

Le due componenti chiave

IPM con barriera di potenza

- Metodo del punto interno: attraversa l'interno della regione ammissibile
- Barriera $x^{-\alpha}$ più aggressiva della logaritmica classica
- Genera sottoproblemi: trovare *cicli a rapporto minimo*

STEP 4 — CHEN ET AL. 2022

Le due componenti chiave

IPM con barriera di potenza

- Metodo del punto interno: attraversa l'interno della regione ammissibile
- Barriera $x^{-\alpha}$ più aggressiva della logaritmica classica
- Genera sottoproblemi: trovare *cicli a rapporto minimo*

Struttura dati dinamica

- Alberi ricoprenti a basso stretch
- I cicli fondamentali approssimano la soluzione
- Aggiornamento efficiente: solo le porzioni modificate

STEP 4 — CHEN ET AL. 2022

Le due componenti chiave

IPM con barriera di potenza

- Metodo del punto interno: attraversa l'interno della regione ammissibile
- Barriera $x^{-\alpha}$ più aggressiva della logaritmica classica
- Genera sottoproblemi: trovare *cicli a rapporto minimo*

Struttura dati dinamica

- Alberi ricoprenti a basso stretch
- I cicli fondamentali approssimano la soluzione
- Aggiornamento efficiente: solo le porzioni modificate

$$E^{1+o(1)} \text{ iterazioni} \times E^{o(1)} \text{ per iterazione} = E^{1+o(1)}$$

STEP 5 — APPLICAZIONE

Segmentazione: dal pixel al grafo

STEP 5 — APPLICAZIONE

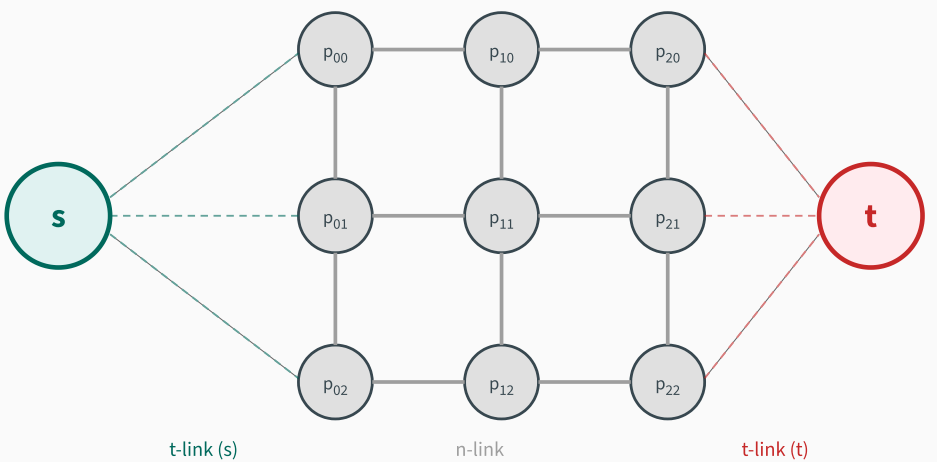
Segmentazione: dal pixel al grafo

Problema: separare oggetto (es. tumore) da sfondo in un'immagine medica (TAC, RM).

STEP 5 — APPLICAZIONE

Segmentazione: dal pixel al grafo

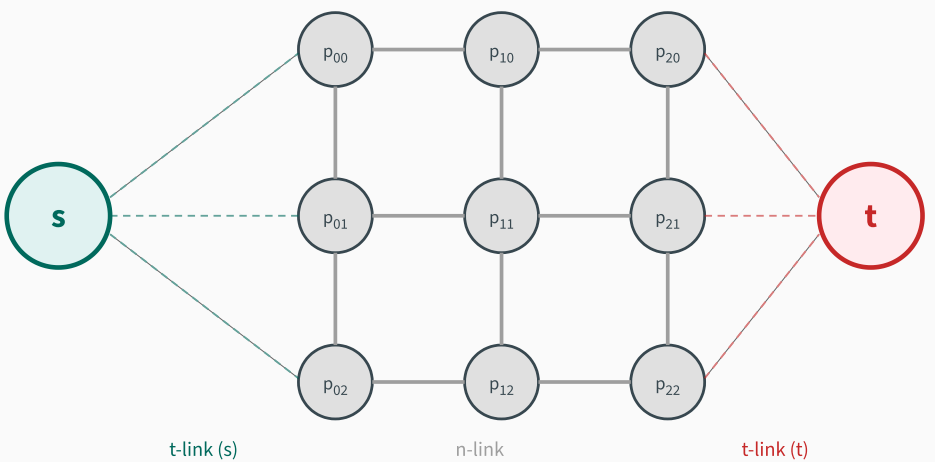
Problema: separare oggetto (es. tumore) da sfondo in un'immagine medica (TAC, RM).



STEP 5 — APPLICAZIONE

Segmentazione: dal pixel al grafo

Problema: separare oggetto (es. tumore) da sfondo in un'immagine medica (TAC, RM).

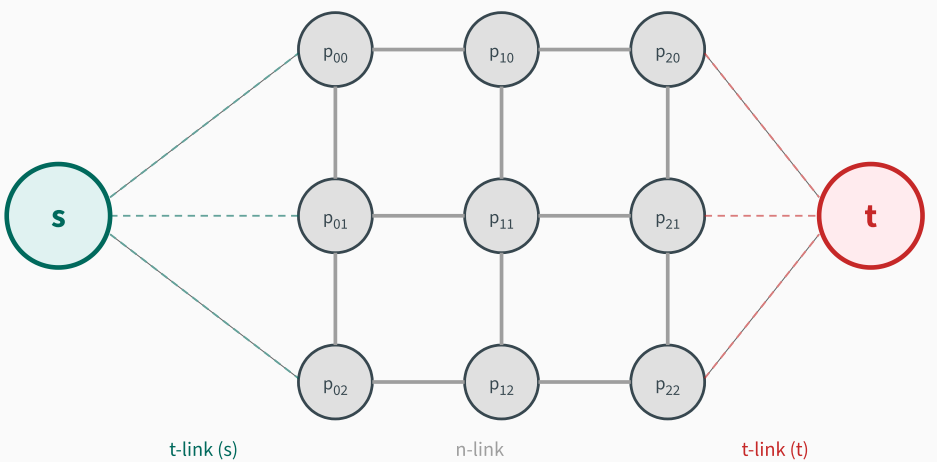


Energia: $E(A) = \lambda \cdot R(A) + B(A)$ — bilancia aderenza ai dati e regolarità

STEP 5 — APPLICAZIONE

Segmentazione: dal pixel al grafo

Problema: separare oggetto (es. tumore) da sfondo in un'immagine medica (TAC, RM).



Energia: $E(A) = \lambda \cdot R(A) + B(A)$ — bilancia aderenza ai dati e regolarità

> ***Taglio minimo nel grafo*** *= segmentazione ottima dell'immagine. Pixel simili = arco pesante = difficile da tagliare.*

STEP 5 — APPLICAZIONE

Applicazioni cliniche e risultati

STEP 5 — APPLICAZIONE

Applicazioni cliniche e risultati

Algoritmo Boykov-Kolmogorov: 3 fasi cicliche

Grow

Espandi due alberi (da s e da t) lungo archi non saturi

→

Augment

Spingi flusso lungo il cammino trovato, satura almeno un arco

→

Adopt

Ripara gli alberi: adotta nodi orfani o liberali

Riutilizza gli alberi tra iterazioni ⇒ **2–5× più veloce** su grafi griglia rispetto a Dinitz e push-relabel

STEP 5 — APPLICAZIONE

Applicazioni cliniche e risultati

Algoritmo Boykov-Kolmogorov: 3 fasi cicliche

Grow

Espandi due alberi (da s e da t) lungo archi non saturi

→

Augment

Spingi flusso lungo il cammino trovato, satura almeno un arco

→

Adopt

Ripara gli alberi: adotta nodi orfani o liberali

Riutilizza gli alberi tra iterazioni ⇒ **2–5× più veloce** su grafi griglia rispetto a Dinitz e push-relabel

Vantaggio chiave: garanzia di ottimalità globale (vs minimi locali di level sets / active contours)

- **Fegato** in TAC • **Polmoni** in PET-CT • **Cervello** in RM • **Mammella** in RM

Workflow: il medico segna pochi pixel seme → l'algoritmo calcola il confine ottimale. Funziona anche in 3D.

Riepilogo e conclusioni

1

Il flusso massimo ha una **teoria elegante** (Max-Flow Min-Cut) con 70 anni di evoluzione algoritmica

2

Da $O(E|f^*|)$ a $E^{1+o(1)}$: **quasi-ottimale**, essenzialmente non migliorabile

3

Le applicazioni (imaging medico) dimostrano l'**impatto concreto** della teoria

Problemi aperti

- Algoritmo fortemente polinomiale migliore di $O(VE)$?
- Versione deterministica del risultato quasi-lineare
- Integrazione deep learning + graph cuts per la segmentazione

Grazie per l'attenzione

Domande?

Riferimenti principali:

Ford, L. R. & Fulkerson, D. R. (1956). “Maximal Flow Through a Network.” *Canadian J. Math.*, 8, 399–404.

Edmonds, J. & Karp, R. M. (1972). “Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems.” *J. ACM*, 19(2), 248–264.

Chen, L. et al. (2022). “Maximum Flow and Minimum-Cost Flow in Almost-Linear Time.” *FOCS ’22*, 612–623.

Boykov, Y. & Kolmogorov, V. (2004). “An Experimental Comparison of Min-Cut/Max-Flow Algorithms.” *IEEE TPAMI*, 26(9), 1124–1137.