Accoppiamento Massimo Bip. (18)

Segue dim. Del Th. *M è un accoppiamento massimo se e solo se non esiste alcun cammino aumentante rispetto ad M.*

- Oss. Ricordiamo che $|M|<|M'|$ per hp.
- Di tutte le componenti ora definite, solo 5 e 6 hanno un diverso num. di archi, e solo 6 ha più archi di *M'* che di *M*.
- Segue che deve esistere almeno una componente di tipo 6 
- Tale comp. è un cammino aumentante per M. Assurdo. CVD

Accoppiamento Massimo Bip. (19)

- Utilizziamo il teorema del Cammino Aumentante per produrre un algoritmo iterativo che, ad ogni iterazione, cerca un nuovo cammino aumentante tramite una modifica di una visita in ampiezza partendo dai nodi che l'accoppiamento non tocca. In questo modo, i nodi sono strutturati in livelli (toccati e non toccati dall'accoppiamento).

Accoppiamento Massimo Bip. (20)

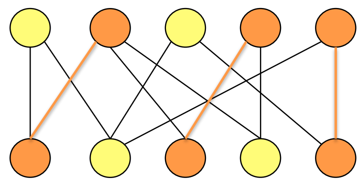
Idea dell'algoritmo:

- Parti da un accoppiamento arbitrario (anche vuoto)
- Finché esistono cammini aumentanti:
  - Trova il cammino aumentante P
  - Scambia in P gli archi dell'accoppiamento con gli altri

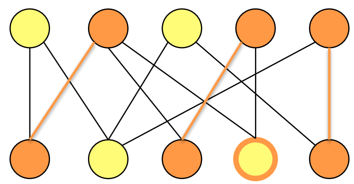
Complessità: dipende dalla complessità di cercare il cammino aumentante.

Accoppiamento Massimo Bip. (21)

- Parti da un accoppiamento arbitrario

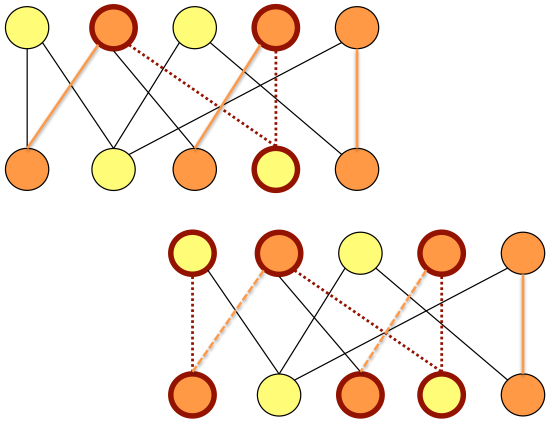


- Scegli un nodo libero…



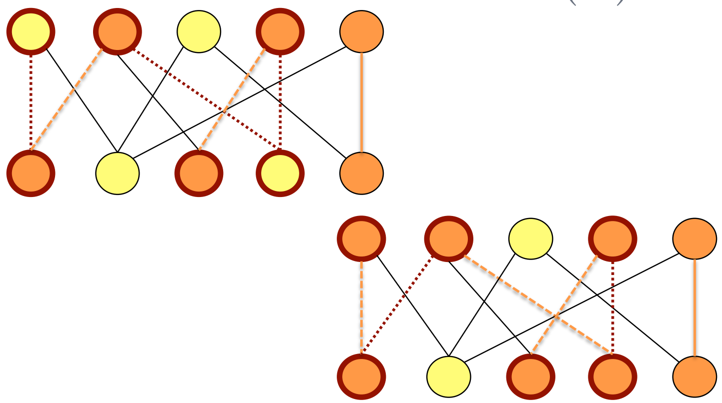
- Ed esegui una BFS modificata…

Accoppiamento Massimo Bip. (22)



… finché non viene raggiunto un altro nodo libero, e quindi si trova un cammino aumentante

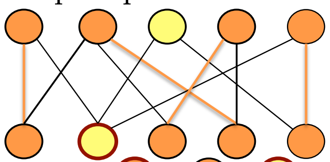
Accoppiamento Massimo Bip. (23)

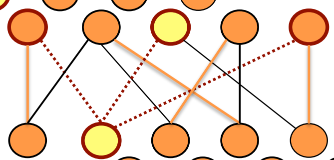


Scambia il ruolo degli archi dell'accoppiamento e degli altri

Accoppiamento Massimo Bip. (24)

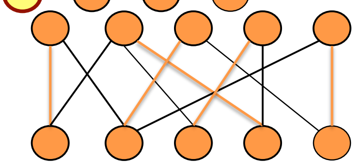
Ripeti: prendi un altro nodo libero…





…Esegui la BFS modificata…

… e scambia



Non ci sono altri cammini aumentanti: Fine

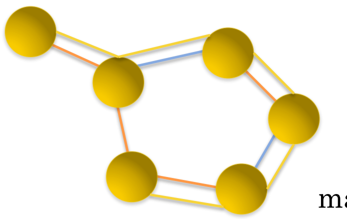
Accoppiamento Massimo Bip. (25)

- Problema: come si trova un cammino aumentante per M?
- Idea:
  - Parti da un nodo libero
  - Esegui una DFS modificata come segue:
    - tieni traccia del livello corrente
    - se il livello è dispari, usa un arco di M
    - se il livello è dispari, usa un arco di E-M
    - Appena trovi un nodo libero hai trovato un cammino aumentante

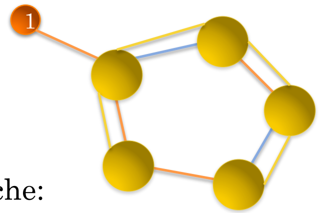
Accoppiamento Massimo Bip. (26)

Esempio:

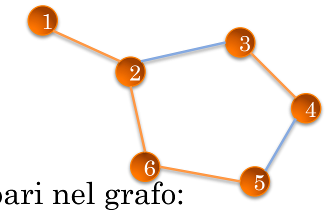
- Parti da un nodo libero
- Esegui una DFS modificata come segue:
  - tieni traccia del livello corrente
  - se il livello è dispari, usa un arco di M
  - se il livello è dispari, usa un arco di E-M
  - Appena trovi un nodo libero hai trovato un cammino aumentante



ma anche:



Accoppiamento Massimo Bip. (27)



- Problema: presenza di cicli dispari nel grafo:
  - in un ciclo dispari c'è sempre un nodo libero con due archi non in *M* che contribuiscono al ciclo
  - se la DFS percorre il ciclo nella direzione "sbagliata" il cammino aumentante non viene trovato
- Grafi senza cicli dispari: grafi bipartiti

Accoppiamento Massimo Bip. (28)

Algoritmo TrovaCamminoAumentante $(G=(U \cup W,E), M)$

- parti da un nodo libero di U
- se il nodo corrente è in U segui un arco non in M
- altrimenti segui un arco in M
- appena trovi un nodo di W libero hai trovato un cammino aumentante

Complessità: $O(n+m)$

Complessità dell'algoritmo che trova l'accoppiamento massimo: $n/2[O(n+m)+O(n)]=O(nm)$

max n. di iterazioni

inversione archi del cammmino aum.

Accoppiamento Massimo Bip. (29)

- L'algoritmo di Hopcroft–Karp (1973) trova un accoppiamento massimo di un grafo bipartito in tempo $O(m\sqrt{n})$.
- L'idea è analoga alla precedente, e consiste nell'accrescere ripetutamente la cardinalità dell'accoppiamento parziale cercando cammini aumentanti.
- Invece di trovare un cammino aumentante per ogni iterazione, l'algoritmo cerca un insieme massimale di cammini aumentanti.
- In questo modo sono necessarie solo $O(\sqrt{n})$ iterazioni.

Accoppiamento Massimo Bip. (30)

Algoritmo di Hopcroft–Karp

Passi della k-esima fase:

- breadth first search modificata partendo da tutti i nodi di V_1. La visita termina quando vengono raggiunti nodi liberi di V_2 (al livello k)
- Tutti i nodi liberi di liv. k di V_2 sono messi in un insieme F.
  N.B. v è in F sse è la fine di un cammino aumentante
- Trova un insieme massimale di cammini aumentanti *vertex disjoint* di lungh. k usando una depth first search da F verso i nodi di partenza di V_1 (risalita di padre in padre).
- Ogni cammino trovato è un cammino aumentante usato per aumentare M.
- L'alg. termina quando non ci sono più cammini aumentanti trovati dal primo passo.

Accoppiamento Massimo Bip. (31)

Analisi dell'Algoritmo di Hopcroft–Karp (sketch)

- Ogni passo consiste di una breadth first search ed una depth first search. Perciò può essere implementato in $O(n+m)$.
- I primi $\sqrt{n}$ passi prendono tempo $O(m \sqrt{n})$.
- N.B. ad ogni passo la lunghezza dei cammini aumentanti trovati è sempre maggiore poiché ad ogni passo k vengono trovati tutti i cammini di lungh. k e i rimanenti hanno lungh. maggiore.
- Dopo i primi $\sqrt{n}$ passi, il più corto cammino aumentante è lungo almeno $\sqrt{n}$.
- …

Accoppiamento Massimo Bip. (32)

Segue Analisi dell'Algoritmo di Hopcroft–Karp (sketch)

- La differenza simmetrica tra un possibile accoppiamento ottimo e l'accoppiamento parziale M trovato dai primi $\sqrt{n}$ passi è un insieme di cicli alternanti e di cammini aumentanti *vertex-disjoint*.
- Ciascuno di questi cammini ha lunghezza almeno $\sqrt{n}$, quindi ce ne possono essere al più $\sqrt{n}$ e la dim. dell'accoppiamento massimo è più grande di al più $\sqrt{n}$ archi rispetto ad M.
- Ogni passo dell'algoritmo aumenta M di almeno uno, quindi al più $\sqrt{n}$ ulteriori passi sono sufficienti.
- L'algoritmo esegue quindi al più $2\sqrt{n}$ passi, quindi prende tempo $O(m \sqrt{n})$ nel caso peggiore.

Accoppiamento Massimo Bip. (33)

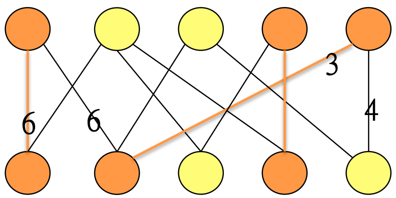
- In molti casi questa complessità può essere migliorata.
- Per esempio, nel caso medio di grafi sparsi bipartiti random, nel 2006 [Bast et al.] è stato provato che i cammini aumentanti hanno lunghezza logaritmica.
- Come conseguenza, l'alg. di Hopcroft–Karp necessita di O(log n) passi e quindi O(m log n) tempo totale.

L'Accoppiamento Perfetto di Peso Minimo in Grafi Bipartiti

Accoppiamento Pesato (1)

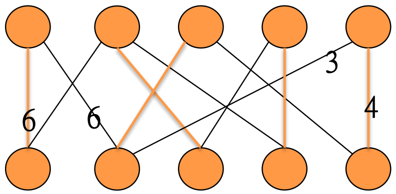
- Definizione di accoppiamento: come prima
- Ogni arco ha un costo
- Noi cerchiamo un accoppiamento perfetto di costo minimo
- N.B. Questo è equivalente a cercare un accoppiamento con costo massimo, in cui i pesi sono negativi

Accoppiamento Pesato (2)



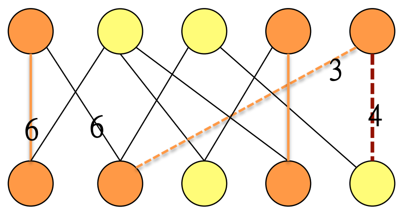
Costo dell'accoppiamento: 6+3+1=10

Accoppiamento di costo massimo: 6+4+1+1+1=13



Accoppiamento Pesato (3)

Def. Cammino aumentante (diversa da prima!) Ogni cammino alternante tale che il costo totale degli archi non accoppiati > del costo degli archi accoppiati.
Costo del cammino aumentante= Costo degli archi non accoppiati-costo degli archi accoppiati



N.B. Ora i cammini aumentanti non devono finire necessariamente con un arco fuori dell'accoppiamento.

Accoppiamento Pesato (4)

Algoritmo:

- Inizia con un accoppiamento vuoto
- Ripeti
  - Trova un cammino aumentante P con costo massimo
  - Se il costo > 0, scambia il ruolo degli archi
  - Altrimenti ritorna l'accoppiamento di massimo peso.

- Complessità: almeno *O(nm)*.

Accoppiamento Pesato (5)

- E' possibile definire il problema dell'accoppiamento di minimo peso come un problema di programmazione lineare (Hungarian method):
  - Dato un accoppiamento *M*, sia *x* il suo vettore di incidenza, dove $x_{ij} = 1$ se (i, j) è in *M* e 0 altrimenti.
  - Il problema diviene:

    minimizzare $\sum_{i,j} c_{ij} x_{ij}$ soggetto a

$$\sum_j x_{ij} = 1, i \in A$$

$$\sum_i x_{ij} = 1, j \in B$$

$$x_{ij} \geq 0, i \in A, j \in B$$

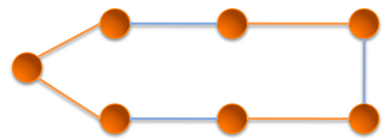
$$x_{ij} \text{ integer}, i \in A, j \in B$$

- Complessità: $O(n^3)$.

L'Accoppiamento Massimo in Grafi Qualunque

Boccioli (Blossoms) (1)

- Abbiamo detto che il problema dei grafi qualunque risiede nei cicli dispari contenenti un numero massimale di archi dell'accoppiamento



- Tali cicli sono detti boccioli (blossoms)

Boccioli (Blossoms) (2)

- Lemma (della contrazione dei cicli). Sia M un accoppiamento di G e B un bocciolo. Sia B nodo-disgiunto dal resto di M. Sia G' il grafo ottenuto da G contraendo B in un singolo nodo. Allora M' di G' indotto da M è massimo in G' sse M è massimo in G.
- Dim. M max in G => M' max in G'

  P.A. M' non è max. Quindi esiste un cammino aumentante P in G' rispetto ad M'. Sia b il nodo che rappresenta B.

  Due casi:

1. il cammino non passa per b => P aumentante anche per M. ASSURDO

Boccioli (Blossoms) (3)

Segue dim. del Lemma della contrazione dei cicli

2. il cammino passa per b => b è un estremo di P poiché gli archi di G' incidenti a b non sono in M per hp.

   Sia v il nodo libero di B

   Definisci $P' = P \cup P''$ dove P'' è dentro B e congiung b con v.

   P' aumentante per G. ASSURDO.

Boccioli (Blossoms) (4)

Segue dim. del Lemma della contrazione dei cicli

- M' max in G' => M max in G

  P.A. M non è max. Sia P un cammino aumentante in G per M.

  Due casi:

1. P non passa per b => P aumentante per G'. ASSURDO
2. P passa per b. Poiché B contiene un solo nodo libero, almeno un estremo di P è fuori di B. Sia w.

   Sia P' il sottocammino di P che congiunge w con b.

   P' è un cammino aumentante per G'.

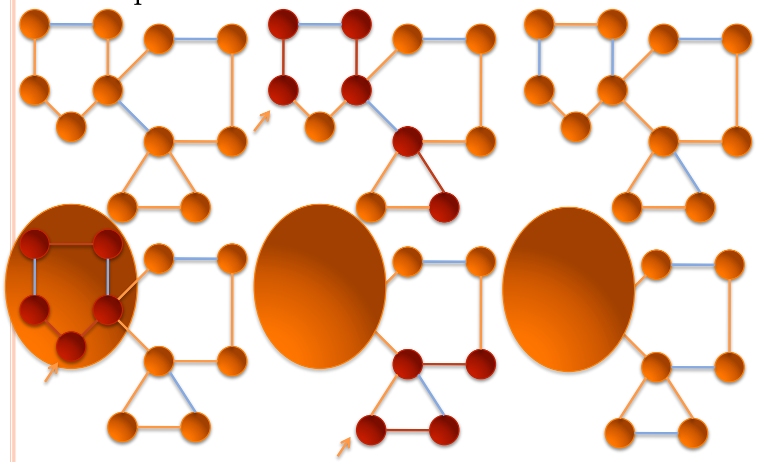
   ASSURDO. CVD

Accoppiamento Max in Grafi Qualunque (1)

- Per trovare un cammino aumentante in un grafo generale, basta modificare l'algoritmo per i bipartiti in modo che trovi anche i boccioli.
- Per ogni bocciolo trovato, questo viene contratto in un nodo e generato un nuovo grafo.
- Ogni cammino aumentante trovato sul nuovo grafo si traduce facilmente in un cammino aumentante in G.
- Per il lemma precedente, se M è massimo nel nuovo grafo, esso è massimo anche in G.

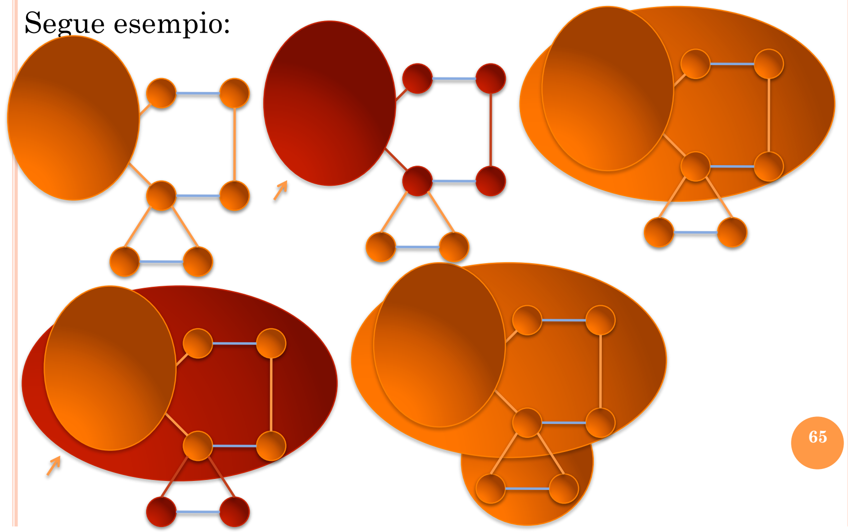
Accoppiamento Max in Grafi Qualunque (2)

Esempio:



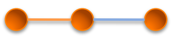
Accoppiamento Max in Grafi Qualunque (3)

Segue esempio:



Accoppiamento Max in Grafi Qualunque (4)

Algoritmo di Edmonds ['65]

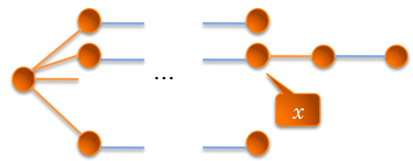
- M accoppiamento di G
- L sottinsieme dei nodi liberi (se L è vuoto => M max)
- F foresta t.c. ogni nodo di L corrisponde ad una componente di F
- Estendi F aggiungendo 
- Quindi: nodi a dist. dispari da elementi di L hanno grado 2 (1 in M e 1in E-M): siano interni
- Gli altri nodi: esterni

(segue)

Accoppiamento Max in Grafi Qualunque (5)

Algoritmo di Edmonds – segue

- Considera i vicini dei nodi esterni.
- 4 possibilità:
  1. esiste x esterno incidente ad un y non in F: aggiungi ad F gli archi (x,y) ed (y,z), con (y,z) arco di M.



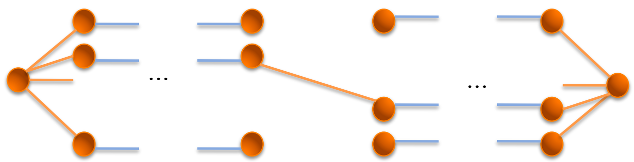
(segue)

Accoppiamento Max in Grafi Qualunque (6)

Algoritmo di Edmonds – segue

2. due nodi esterni in due diverse componenti di F sono adiacenti:

   cammino aumentante



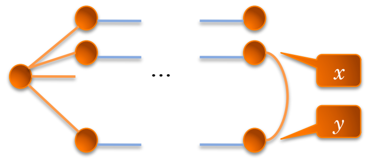
(segue)

Accoppiamento Max in Grafi Qualunque (7)

Algoritmo di Edmonds – segue

3. 2 nodi esterni x, y nella stessa componente di F sono adiacenti:

   sia C il ciclo che si forma. E' possibile spostare gli archi di M in C in modo che soddisfi la condizione del lemma della contrazione dei cicli => grafo ridotto G'



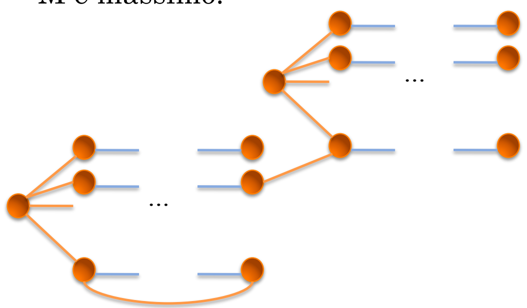
(segue)

Accoppiamento Max in Grafi Qualunque (8)

Algoritmo di Edmonds – segue

4. tutti i nodi esterni sono adiacenti a soli nodi interni:

   M è massimo.



Accoppiamento Max in Grafi Qualunque (9)

Lemma. Ad ogni passo dell'algoritmo di Edmonds, o cresce la dimensione di F, o decresce la dimensione di G, o si trova un cammino aumentante, o M è massimo.

Complessità. Num. di terazioni ≤

num. delle crescite di F (al più n)+

num. delle contrazioni di boccioli (al più n)+

num. dei cammini aumentanti (al più $n/2$).

La complessità dipende dalla gestione dei boccioli. A seconda delle versioni: $O(n^3)$ o $O(mn^2)$.

Migliore complessità: $O(m\sqrt{n})$ [Micali & Vazirani '80]

Un Problema di Scheduling (per l'anno prossimo)

Un Problema di Scheduling

- prendi la problematica dello scheduling su circuiti VLSI dall'articolo:
- A.H. Timmer and J. A. G. Jess. Exact Scheduling Strategies based on Bipartite Graph Matching. Proc. European Design & Test Conference 42-47, 1995.
- citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.30.8394&rep=rep1&type=pdf