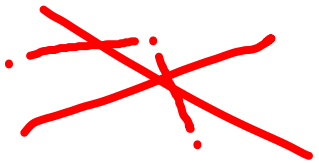
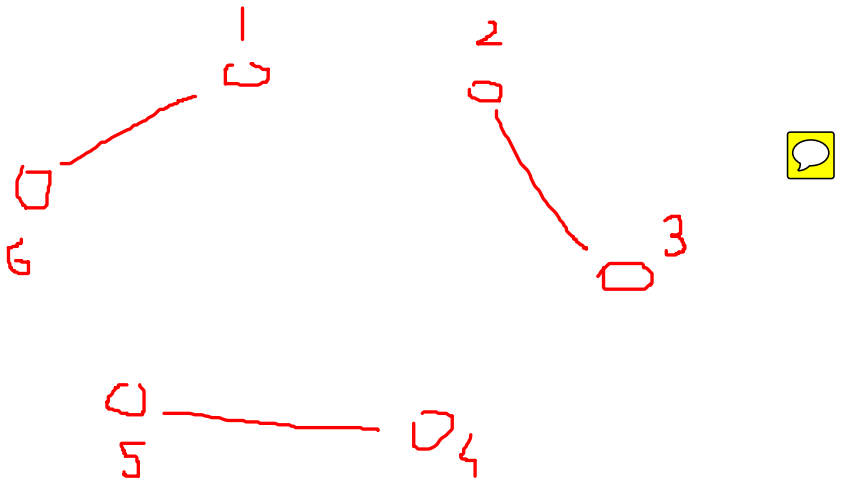
Matching

Definizione 1 *Dato un grafo non orientato $G = (V, A)$ un matching su tale grafo é un sottinsieme $M \subseteq A$ dell'insieme di archi A tale che non ci sono in M coppie di archi adiacenti, ovvero con un nodo in comune.*





Matching di peso massimo

Se associamo ad ogni arco $e \in A$ un peso $w_e > 0$ possiamo associare un peso anche a un matching M pari alla somma dei pesi degli archi in M, cioé:

$$w(M) = \sum_{e \in M} w_e.$$

Nel problema di *matching pesato* si vuole determinare tra tutti i possibili matching in un grafo quello di peso massimo, cioé si vuole risolvere il seguente problema:

$$\max_{M \subseteq A \text{ é un matching}} w(M).$$

Applicazioni

Questo problema modella tutte le applicazioni in cui abbiamo membri di un insieme (rappresentati dai nodi del grafo) alcuni dei quali accoppiabili tra loro (i nodi collegati da archi). Ad ogni potenziale accoppiamento si associa un peso (o profitto) e si vuole individuare quali coppie formare (tenendo conto che un membro dell'insieme puó far parte al massimo di una coppia) in modo da massimizzare il profitto totale.

Matching di cardinalitá massima

Caso particolare:

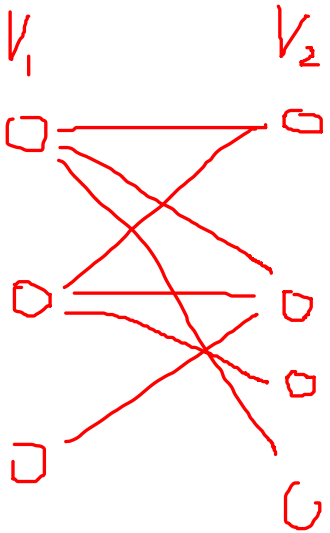
$$w_e = 1 \quad \forall\, e \in A.$$

In tal caso il peso di un matching coincide con la sua cardinalitá $\mid M \mid$ e si parla di problema di *matching di cardinalitá massima*.

Su grafi bipartiti

In tutte le applicazioni in cui gli elementi accoppiabili tra loro appartengono a due classi distinte (ad esempio, lavoratori da una parte e lavori dall'altra), il grafo corrispondente $G = (V, A)$ é un grafo bipartito con le due classi di bipartizione V_1 e V_2.

Supporremo inoltre che tutti i pesi siano uguali a 1 (matching di cardinalitá massima).



Un matching iniziale

Un matching di partenza (non necessariamente di cardinalitá massima) può essere ottenuto con la seguente semplice procedura:

Inizializzazione Si ponga $M = \emptyset$.

1. Se esiste un arco $\overline{e} \in A \setminus M$ che non sia adiacente ad alcun arco in M, allora porre $M = M \cup \{\overline{e}\}$ e ripetere il passo 1. Altrimenti: STOP.

Algoritmo di risoluzione

- **Inizializzazione** Sia M un matching di partenza (eventualmente individuato con la procedura vista).
- **Passo 0** Tutti i vertici del grafo sono non etichettati.
- **Passo 1.** Se non c'è alcun vertice del grafo in V_1 che soddisfa le seguenti proprietá
  - è non etichettato;
  - su di esso non incide alcun arco in M,

  allora: STOP. Altrimenti si seleziona un vertice del grafo in V_1 con tali proprietá e gli si assegna etichetta $(E, -)$. Si inizializzi l'insieme R dei vertici *analizzati* con $R = \emptyset$.

Continua

- **Passo 2.** Se tutti i vertici etichettati sono stati analizzati, ritornare al Passo 1.; altrimenti selezionare un vertice k etichettato ma non analizzato ed analizzarlo, cioé si ponga $R = R \cup \{k\}$. Analizzare un vertice vuol dire compiere le seguenti operazioni:

  a) Se la prima componente dell'etichetta di k è una E, allora si assegna un'etichetta (O, k) a tutti i vertici del grafo adiacenti a k e non ancora etichettati; quindi si ripete il Passo 2.

  b) Se la prima componente dell'etichetta di k è una O, allora sono possibili due casi

  Caso 1 C'è un arco marcato incidente su k: in tal caso si assegna l'etichetta (E, k) al vertice unito a k dall'arco in M e si ripete il Passo 2.;

  Caso 2 Non ci sono archi in M incidenti su k. In tal caso si va al Passo 3.

Continua

- **Passo 3.** Utilizzando la seconda componente delle etichette risalire dal vertice k in cui ci si è bloccati al Passo 2. fino al vertice s con seconda componente pari a $-$. In questo modo si è individuato un cammino da s a k che inizia con un arco non appartenente a M, prosegue alternando archi in M e archi non appartenenti a M, e termina in k con un arco non appartenente a M. A questo punto si aggiorna M invertendo l'appartenenza e non appartenenza a M per i soli archi lungo tale cammino. Quindi, tutti gli archi non appartenenti a M lungo il cammino vengono inseriti in M e viceversa. Infine si cancellano tutte le etichette ripartendo dal Passo 1.

Complessitá dell'algoritmo

Si puó dimostrare che questo algoritmo richiede un numero di operazioni $O(\min(\mid V_1 \mid, \mid V_2 \mid) \mid A \mid)$ e ha quindi complessitá polinomiale.

Va sottolineato che non è il meglio che si possa ottenere per questo problema. Esiste infatti anche un altro algoritmo, che non vedremo, la cui complessitá é pari a $O(\mid V \mid^{1/2} \mid A \mid)$.

Nota bene

L’algoritmo appena visto può essere considerato un algoritmo di raffinamento locale.

Infatti, si parte da un matching valido (eventualmente vuoto) e a ogni iterazione (esecuzuione dei Passi 1 e 2) si cerca di individuarne uno di cardinalità più elevata (quando lo si trova, esso viene calcolato al Passo 3).