Simone Remoli

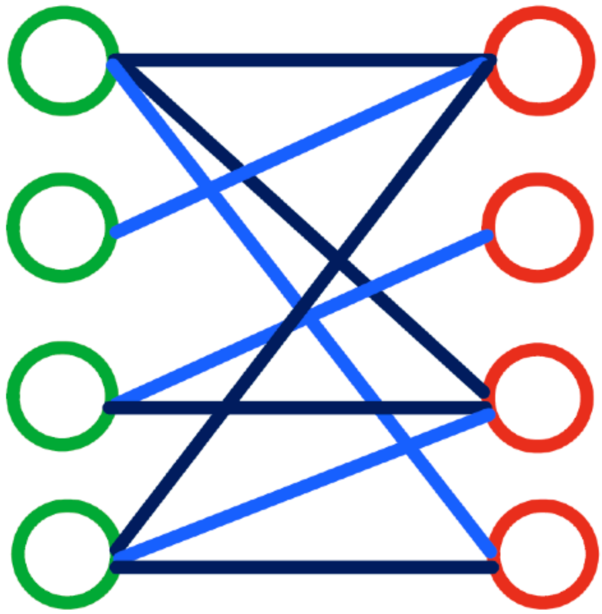
MATCHING SU GRAFI BIPARTITI

Cos'è un grafo bipartito?

Un grafo bipartito è un tipo di grafo i cui nodi possono essere divisi in due insiemi distinti, in modo che ogni arco colleghi un nodo di un insieme a un nodo dell'altro.

Più precisamente?

Un grafo bipartito è un tipo di grafo i cui vertici possono essere suddivisi in due insiemi distinti (chiamati qui U e V), con la caratteristica che ogni **arco collega un vertice in U a un vertice in V.** In altre parole, non ci sono archi tra i vertici all'interno dello stesso insieme; tutti gli archi collegano vertici di insiemi opposti.



L'immagine raffigura un semplice grafo bipartito. L'insieme U è rappresentato dai nodi verdi mentre l'insieme V è rappresentato dai nodi rossi.

Questo è facile, facciamo un passo avanti.

Un grafo bipartito è una struttura che si presta a problemi di **Matching** (abbinamenti ottimali) e ad algoritmi di **flusso massimo.**

Inquadriamo il problema.

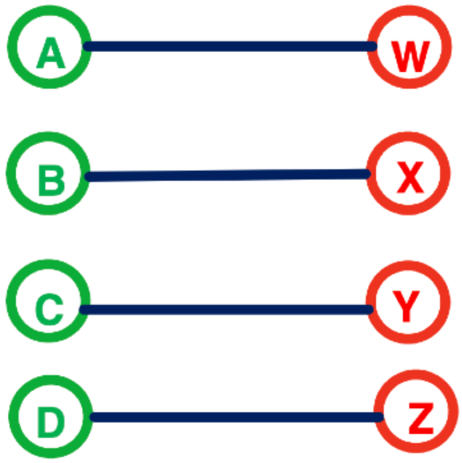
Un **Matching** è un sottoinsieme di archi M ⊆ E, tale che nessun nodo in U o V è **incidente** con più di un arco in M. <u>In altre parole, ogni nodo può essere collegato con al massimo un nodo dell'altro insieme</u>.

Tipologie di matching.

Matching massimo: un matching che contiene il maggiore numero possibile di archi, **abbinando il maggior numero di nodi. Non è detto che tutti i nodi debbano essere abbinati;** semplicemente, <u>non si può aggiungere un altro arco al matching senza violare la regola</u>.

Entrambi i grafi presentano un <u>*matching massimo*</u>.
Il grafo a sinistra ha un matching massimo di valore tre composto dall'insieme: **M = {(A,X),(B,Y),(C,Z)}.**
Il grafo a destra presenta anche lui un matching massimo nonostante il il nodo C di U non sia abbinato a nessun nodo di V, quindi non tutti i nodi sono coinvolti nel matching, eppure per quel grafo il matching massimo ha valore due, ed è composto dall'insieme:
M = {(A,X),(B,Y)}.

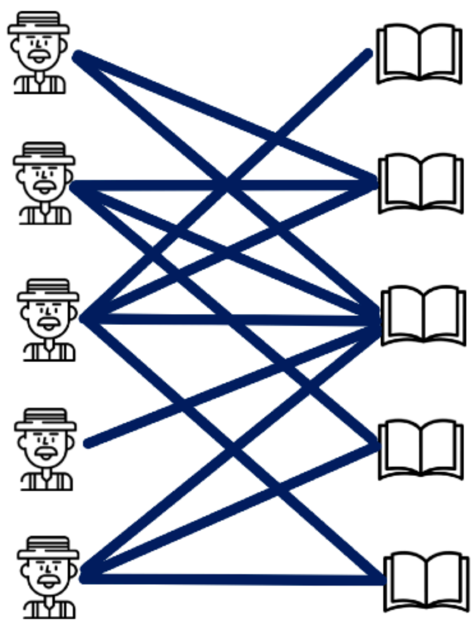
Matching perfetto: un matching in cui ogni nodo di U è abbinato ad esattamente ed unicamente un nodo di V e viceversa. Questo è possibile solo se |U| = |V|.



Ogni matching perfetto è completo/massimo, ma non tutti i matching completi/massimi sono perfetti.

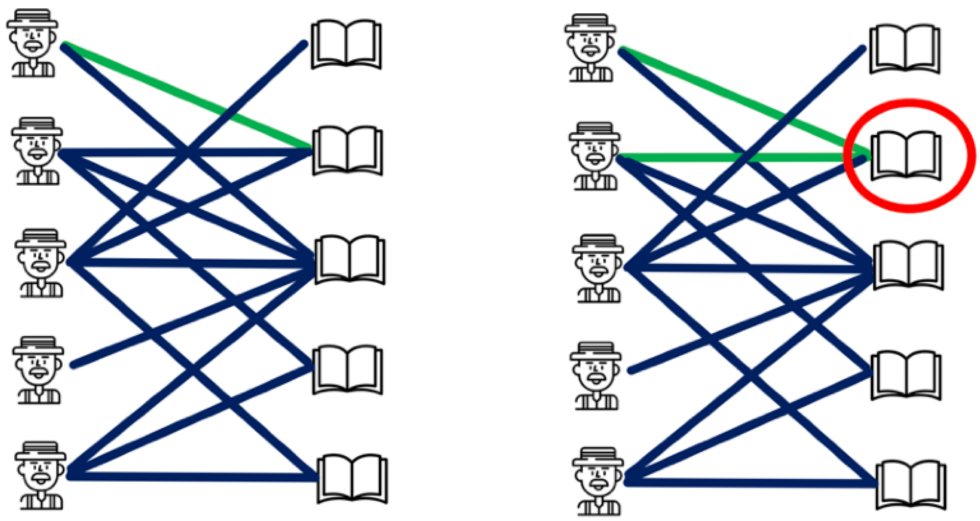
Corrispondenza tra il massimo flusso S-T in un grafo e il problema del matching.

Esprimiamo gli insiemi nel seguente modo:
Ogni prenotato presso la libreria si lega al libro riservato in una perfetta simbiosi, rendendo l'insieme delle corrispondenze un'espressione sublime dell'incontro tra pensiero e sapere, una danza armoniosa di intelletti e pergamene.



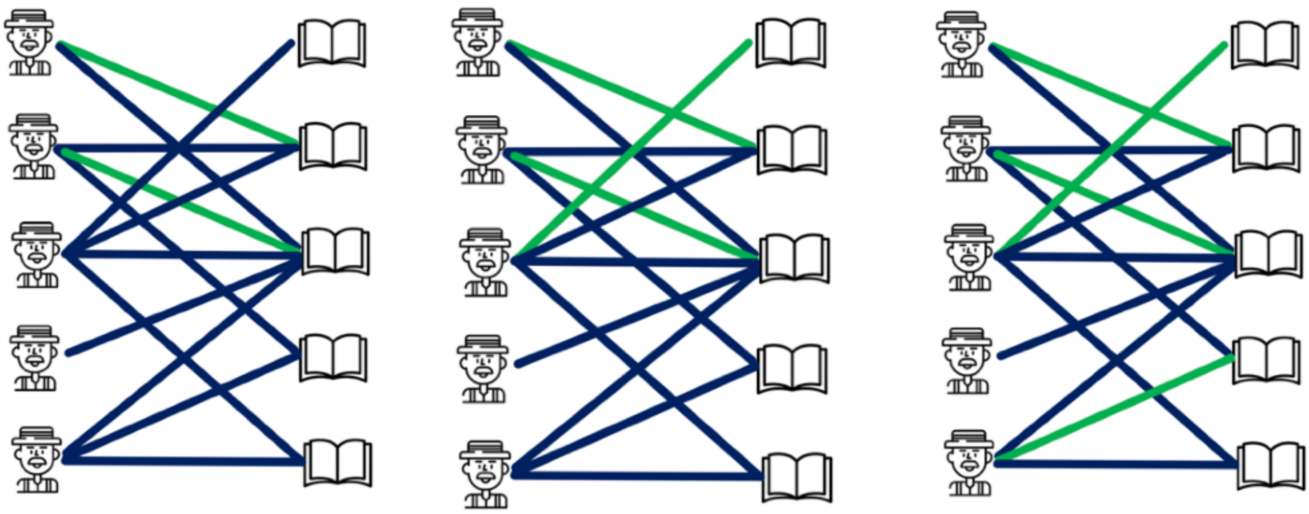
Supponiamo di voler individuare un **matching massimo** in questa rete, applicando un procedimento di risoluzione ispirato all'algoritmo **greedy**, o "goloso".
Tale metodo, per sua natura, è incline a selezionare in ciascun passaggio quella che appare, nell'immediato, come la scelta più favorevole, in un processo iterativo che mira a comporre una soluzione gradualmente, tassello dopo tassello, senza badare agli effetti a lungo termine.

Tuttavia, l'assenza di una visione prospettica e la deliberata noncuranza per le implicazioni future possono condurci all'errore di **non raggiungere il matching massimo desiderato.** Così facendo, il nostro cammino di ricerca potrebbe arrestarsi in una soluzione subottimale, poiché il metodo greedy, nella sua ricerca rapida e immediata, è suscettibile di perdere di vista la soluzione globale ottimale che, sebbene raggiungibile, sfugge alla morsa dell'avidità del momento.



Okay, capisco che il secondo "omino" non può matcharsi con il secondo libro perché già il primo "omino" ha preso quest'ultimo (probabilmente ispirato dalla sua voglia di leggere).

L'approccio di tipo greedy ha condotto il risultato ad un matching massimo di valore quattro.
È evidente che in questa rete il matching di valore massimo non è quattro. Possiamo fare di meglio.



Per trovare un matching su grafo bipartito possiamo ridurci ad un problema di massimo flusso.

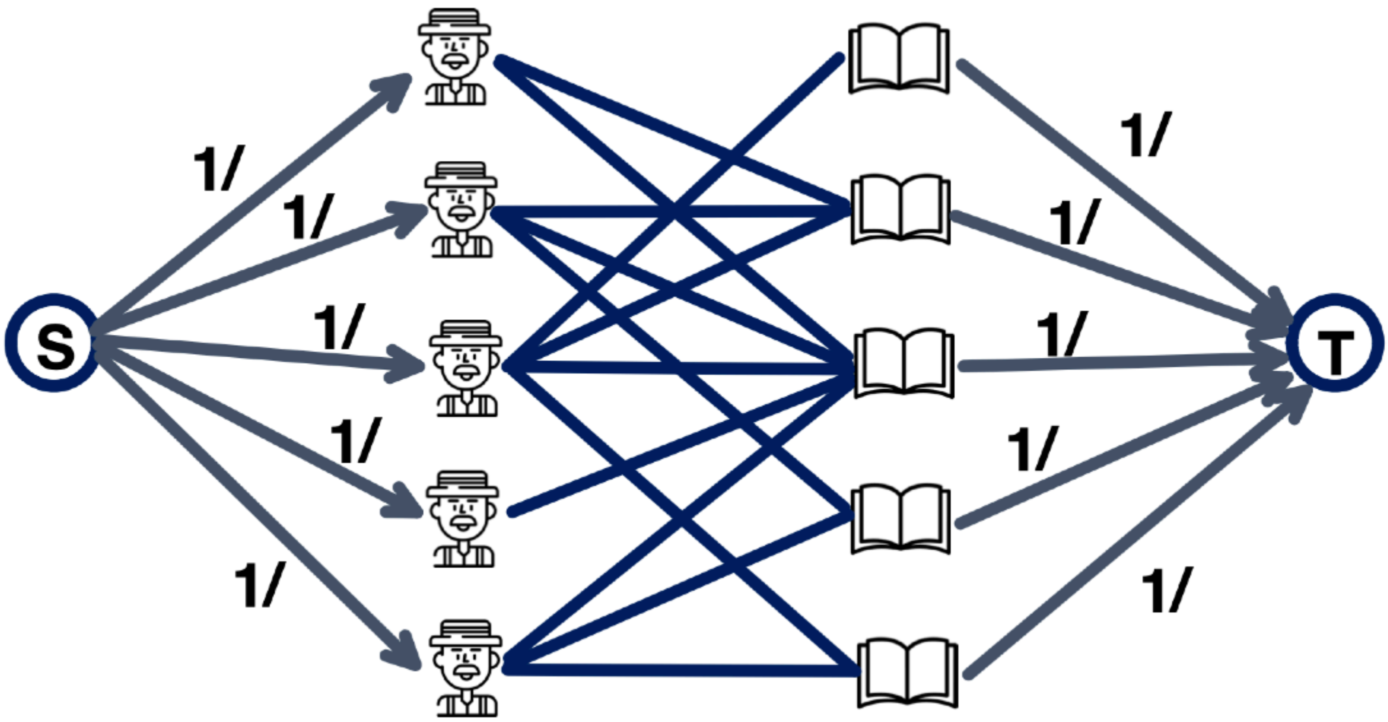
Come procedere?

La corrispondenza tra i due problemi può essere osservata se il grafo bipartito è convertito in un **grafo di flusso.** Si può procedere così:

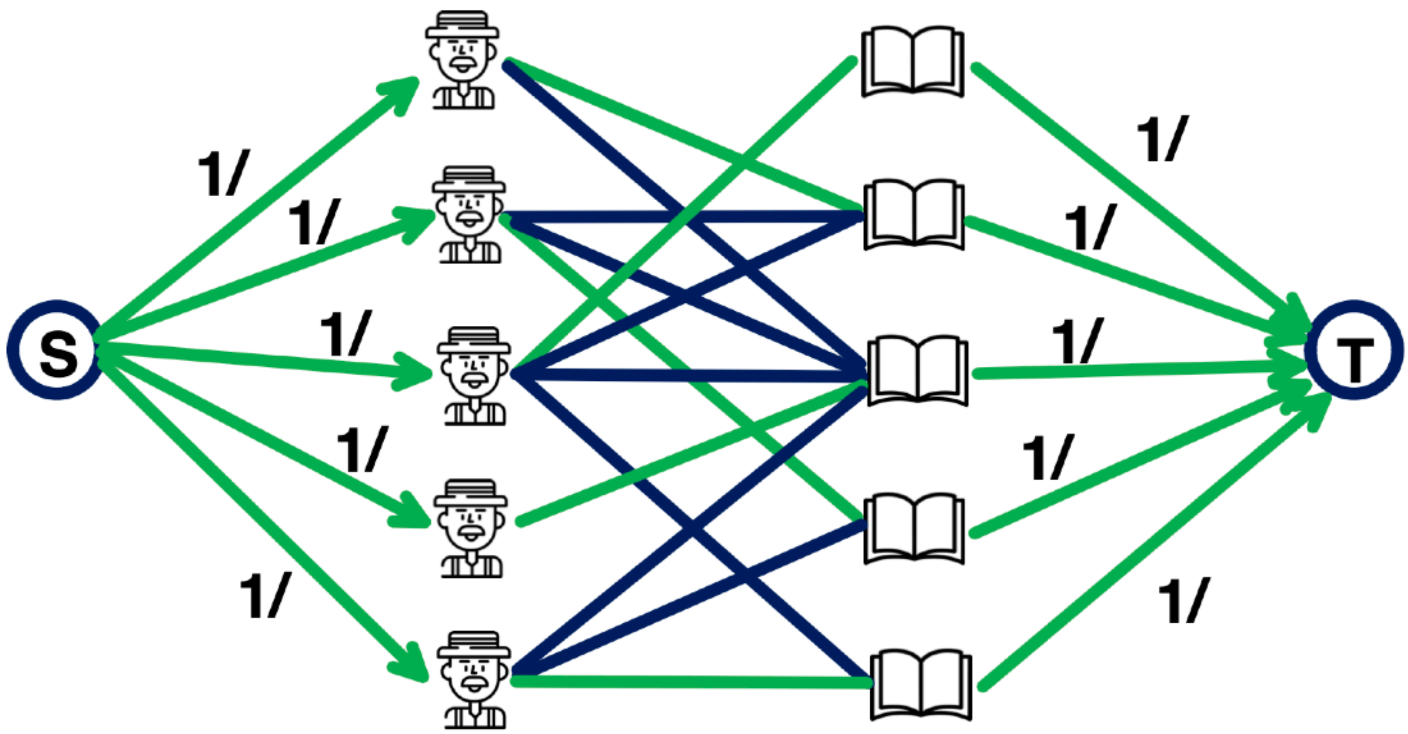
1. **Aggiunta di una sorgente e di un pozzo**: si aggiunge un nodo sorgente S che è connesso a tutti i nodi di uno dei due insiemi e un nodo destinazione T che è connesso a tutti i nodi dell'altro insieme.

2. **Impostazione delle capacità**: si assegna una capacità di 1 a ogni arco, inclusi quelli tra S e il primo insieme e tra il secondo insieme e T.
Questo limita il flusso massimo su ogni arco a 1 unità, il che rappresenta il vincolo del matching (ogni nodo può essere abbinato al più una volta).

3. **Risultato del flusso massimo**: il valore del flusso massimo S-T rappresenta il numero massimo di abbinamenti possibili tra i due insiemi nel grafo bipartito, poiché ogni unità di flusso corrisponde a un arco selezionato nel matching.
È stata aggiunta una capacità unitaria su ogni arco. Il flusso è zero.
Una volta che impongo che esiste una sola unità di flusso che può transitare su quell'arco, sto imponendo il vincolo che,



dal momento in cui può entrare al più uno, può uscire al più uno. Quindi quel nodo potrà matchare una sola volta.
Ora sappiamo perfettamente ricostruire il matching massimo.



Teorema: Esiste una corrispondenza 1:1 tra i flussi interi di valore K e i matching di valore K.

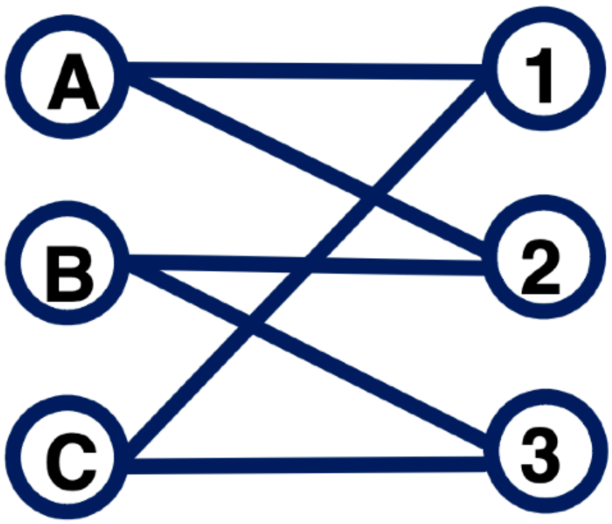
Corollario: ***Ogni volta che possiedo un matching di valore K posso inferire un flusso di valore K*.**
***Ma vale anche il viceversa*.**
Nota che il flusso di questo grafo è pari a 5.
Il matching massimo di questo grafo è pari a 5.
Se esiste un flusso di valore 5 deve esistere necessariamente un matching di valore 5.

Condizione di Hall.

Definizione 1: **Un matching di cardinalità |X| è detto X-completo.**
La **Condizione di Hall** specifica un criterio per determinare se un matching X-completo è possibile.

Affermazione condizione di Hall: **Se esiste un sottoinsieme Q ⊆ X tale che il numero di nodi adiacenti a Q (indicato come N(Q)) è inferiore alla cardinalità di Q, allora non può esistere un matching X-completo.**
In altre parole, per ogni sottoinsieme di X, l'insieme dei nodi adiacenti deve essere almeno grande quanto il sottoinsieme Q stesso, altrimenti non è possibile costruire un matching che copra tutti i nodi di X.

Facciamo un esempio:



X = {A,B,C}
Y = {1,2,3}
Troviamo i sottoinsiemi di X.

Teorema: **Il numero di sotto-insiemi di un insieme è pari all'insieme delle parti meno uno** (vedi corso algebra e logica).

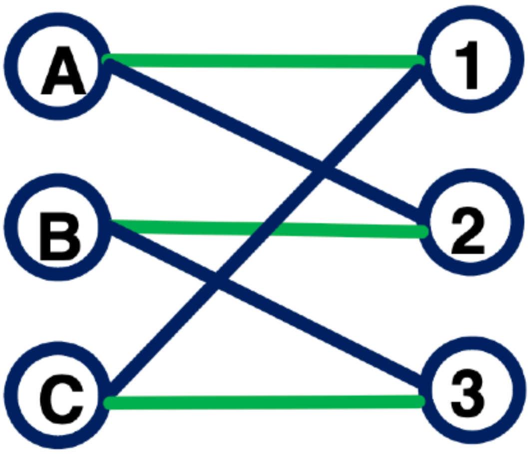
Ci sono 7 sottoinsiemi di X.

1. Q = {A} e N(Q) = {1,2};
2. Q = {B} e N(Q) = {2,3};
3. Q = {C} e N(Q) = {1,3};
4. Q = {A,B} e N(Q) = {1,2,3};
5. Q = {A,C} e N(Q) = {1,2,3};
6. Q = {B,C} e N(Q) = {1,2,3};
7. Q = {A,B,C} e N(Q) = {1,2,3};

1. $2 > 1$
2. $2 > 1$
3. $2 > 1$
4. $3 > 2$
5. $3 > 2$
6. $3 > 2$
7. $3 = 3$

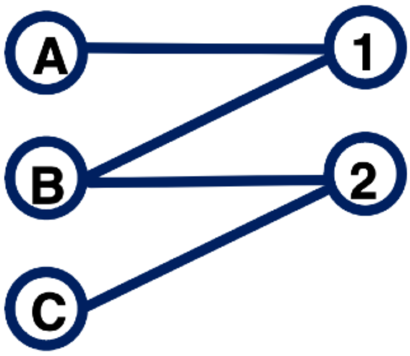
NON ESISTE UN SOTTOINSIEME Q TALE CHE IL NUMERO DI NODI ADIACENTI SIA INFERIORE ALLA CARDINALITÀ DI Q STESSO.

Pertanto in questa rete ESISTE un match X- Completo.



Il match è di valore pari alla cardinalità di X.

Vediamo un esempio in cui la condizione di Hall non è soddisfatta.



In questo esempio possiamo notare anche ad occhio che non esiste un matching X-Completo.

Esiste il sottoinsieme massimo Q = {A,B,C} dove il numero dei nodi adiacenti {1,2} è inferiore alla cardinalità di Q, pertanto 2 < 3, ciò vuol dire che non esiste un matching X-Completo.