

MARVEL SNA

Greta Rausa-150634

Introduzione e risultati attesi	2
Il dataset	3
Creazione di una rete pesata di eroi	3
Esportazione ed analisi del Grafo	5
Caratterizzazione della rete	6
Centralità dei nodi	8
Degree Centrality (nodi influenti)	8
Betweenness centrality	8
Closeness centrality (broadcasters)	9
Eigenvector centrality	9
I nodi più importanti per ogni centralità	10
Visualizzazione tramite Gephi	12
Community Detection	14
Le metriche	14
Silhouette index - max[-1, +1]	14
Calinski-Harabasz Index - max [0, +∞]	14
Davies Bouldin Score - min [0, +∞]	14
Dunn Index - max[0, +∞]	14
Algoritmi di clustering	15
K-means clustering	15
Agglomerative clustering	17
Spectral Clustering	19
Reclustering	20
Affinity propagation	20
Clustering Ensemble	22
Community Detection	24
Label Propagation Method	25
Clauset-Newman-Moore (Greedy)	26
Minimum Cut	27
Tecnologie utilizzate	28
Conclusioni	28

Introduzione e risultati attesi

In questo progetto ho esplorato la social network dei personaggi dell'Universo Marvel utilizzando gli stessi dati usati nello studio "Marvel Universe looks almost like a real social network". (Cesc Rosselló, Ricardo Alberich, and Joe Miro from the University of the Balearic Islands, 2002).

In particolare, spero di scoprire intuizioni su eroi e fumetti, nonché la relazione tra loro. Indaghiamo sulla struttura della rete di collaborazione dell'Universo Marvel, in cui due personaggi Marvel sono considerati collegati se compaiono insieme nello stesso fumetto. Mostriamo che questa rete non è chiaramente una rete casuale e che ha la maggior parte delle caratteristiche delle reti di collaborazione "reali", come attori cinematografici o reti di collaborazione scientifica.

Una delle caratteristiche principali dei fumetti Marvel dagli anni Sessanta ai giorni nostri è stata la creazione e lo sviluppo, sotto la guida di Stan Lee, del cosiddetto Universo Marvel. La natura e la durata dei crossover hanno portato alla percezione che tutti i personaggi Marvel vivessero le loro avventure nello stesso cosmo immaginario, in cui interagivano come veri attori. Questo concetto è stato aiutato dall'interrelazione di tutti i titoli che venivano creati, che facevano incrociare personaggi e persino trame su base regolare, dall'apparizione degli stessi cattivi e personaggi secondari in fumetti di titoli diversi e dai continui riferimenti agli eventi che stavano accadendo simultaneamente, o erano accaduti, in altri libri.

Trattandosi di una rete modellata su persone (sebbene non reali) con interazioni e diverse dinamiche che si intrecciano tra loro, ci si aspetta che dall'analisi ne emerga una rete di tipo Small World. Sono attesi diversi cluster locali, plausibilmente piccoli: diversi eroi hanno ruoli fondamentali e diametralmente opposti all'interno delle storie, per cui ci si aspetta che i gruppi formati siano piccoli e numerosi. Per lo stesso motivo è attesa una rete con una distribuzione di nodi che segua una power-law (pochi nodi molto connessi, molti nodi poco connessi).

Dopo una classificazione della rete l'analisi verterà su una profilazione delle relazioni e dei nodi, andando ad identificare i key players e le relazioni tra questi. Per comprendere più a fondo le dinamiche della rete verrà effettuata una ricerca delle community e di collegamento tra queste.

Il dataset

L'analisi è stata effettuata su un dataset già esistente, recuperata dal sito [Kaggle](#). Questa rete riguarda le interazioni tra i supereroi apparsi negli episodi della collana di fumetti MARVEL, dai più famosi ai meno noti.

Il dataset mette a disposizione 3 file csv: nodes, edges e network.

nodes.csv - 38179 - due colonne, nodo e tipo, indicano nome e tipo (eroe e comic) del nodo.

edges.csv - 96103 - due colonne, hero e comic, collegano un hero ai comic in cui è apparso.

hero-edge.csv - link tra eroi apparsi nella stessa rete, non pesati

Tra questi tre file, sarà necessaria l'elaborazione del file edges.csv, perché sapendo in quale episodio è apparso ogni eroe, possiamo già costruire una grafo che ne rappresenta le interazioni in una rete pesata, che andremo a creare con l'aiuto delle funzioni cypher and apoc di neo4j.

Creazione di una rete pesata di eroi

Il software utilizzato per estrarre il grafo dal file edges.csv è Neo4j, cypher (cypher query language) è il linguaggio utilizzato per la somministrazione di query su grafi. Il modello di rete estrapolato dal file edges.csv avrà una struttura estremamente semplice: i nodi saranno di due tipi (hero e comic) e saranno collegati dalla relazione APPEARED_IN.

Caricando il file csv online, è possibile importarlo attraverso la query in un unico passaggio. La query è molto semplice: si applica una MERGE a due nodi e alla relazione tra questi.

```
LOAD CSV WITH HEADERS FROM
"https://raw.githubusercontent.com/tomasonjo/neo4j-marvel/master/data/edges.csv" as row
MERGE (h:Hero{name:row.hero})
MERGE (c:Comic{name:row.comic})
MERGE (h)-[:APPEARED_IN]->(c)
```

Lo step successivo è quello di creare una rete pesata. Si da per assunto che più due eroi si incontrano nell'arco dei vari fumetti, più questi saranno uniti. Si salva quindi il numero di eroi apparsi insieme nello stesso fumetto come "weight" per la proprietà "KNOWS".

Viene usato apoc.periodic.iterate per la gestione del batching: mano a mano che il grafo cresce, vogliamo evitare che il refactoring globale avvenga in una sola iterazione, per cui ne eseguiamo il batch tramite Apoc.

Dalla documentazione: Con la funzione apoc.periodic.iterate vengono fornite 2 dichiarazioni: nella prima si forniscono i valori da processare; nella seconda, viene processato un elemento per volta (oppure, con il flag iterateList:true l'intero set).

```

CALL apoc.periodic.iterate(
"MATCH (p1:Hero)-->(:Comic)<--(p2:Hero) where id(p1) < id(p2) RETURN p1,p2",
"MERGE (p1)-[r:KNOWS]-(p2) ON CREATE SET r.weight = 1 ON MATCH SET r.weight = r.weight + 1"
,{batchSize:5000, parallel:false,iterateList:true})

```

Prima dichiarazione: è una semplice query cypher che matcha due eroi apparsi nello stesso episodio. Viene specificato WHERE id(p1) < id(p2) per evitare ridondanze.

Seconda dichiarazione: dal momento in cui il flag iterateList è settato a True, i dati saranno processati interamente in una iterazione. Utilizzando la funzione MERGE, si cercano relazioni tra i nodi, e se non si trovano, queste vengono create. Cypher permette di definire due comportamenti diversi a seconda che la relazione sia stata trovata (usando ON MATCH SET) o creata (usando ON CREATE SET). Inoltre la MERGE permette di non specificare la direzione della relazione, per cui cerca una relazione tra i nodi indicati prescindendo dalla direzione, e se non trovata, questa viene creata con una direzione randomica.

Anche se trattiamo la nostra rete come non orientata, Neo4j memorizza le relazioni come orientate. Questo non è un problema in quanto possiamo eseguire l'istruzione MATCH anche senza specificare la direzione della relazione. In questo modo trattiamo il grafo come non orientato.

Fatto questo abbiamo ottenuto una rete pesata non orientata di eroi, connessi tra loro a coppie tramite una relazione KNOWS, con una proprietà weight che rappresenta il numero di episodi in cui sono apparsi insieme.

Da qui possiamo applicare una serie di misure, ad esempio è possibile controllare quale eroe ha il weight maggiore, quindi che appare in più episodi, tramite la seguente query:

```

MATCH (h:Hero)-[t:KNOWS]-()
RETURN h.name as hero,sum(t.weight) as weighted_degree
ORDER BY weighted_degree DESC LIMIT 10

```

"CAPTAIN AMERICA"	55449
"SPIDER-MAN/PETER PARKER"	43774
"IRON MAN/TONY STARK"	41189
"THOR/DR. DONALD BLAK" "	39763
"WOLVERINE/LOGAN"	36282
"THING/BENJAMIN J. GR"	36247
"SCARLET WITCH/WANDA"	36008
"BEAST/HENRY & HANK& P"	35829
"HUMAN TORCH/JOHNNY S"	34855
"VISION"	34679

Esportazione ed analisi del Grafo

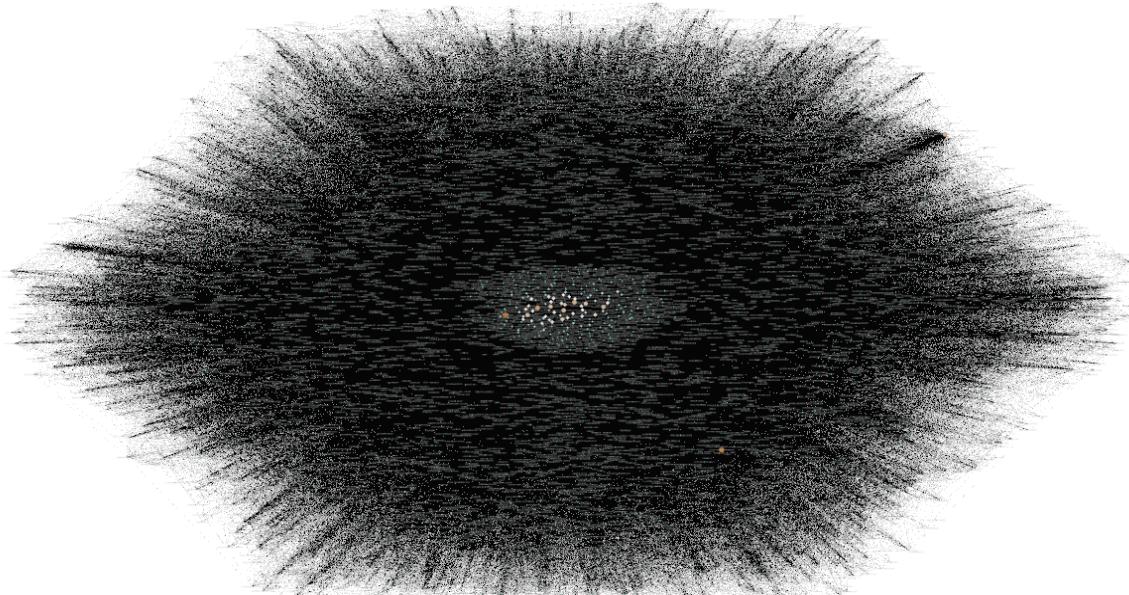
Per visualizzare tutte le connessioni tra eroi, pesate a seconda dei loro incontri:

```
MATCH (h1:Hero)-[t:KNOWS]-(h2:Hero)
```

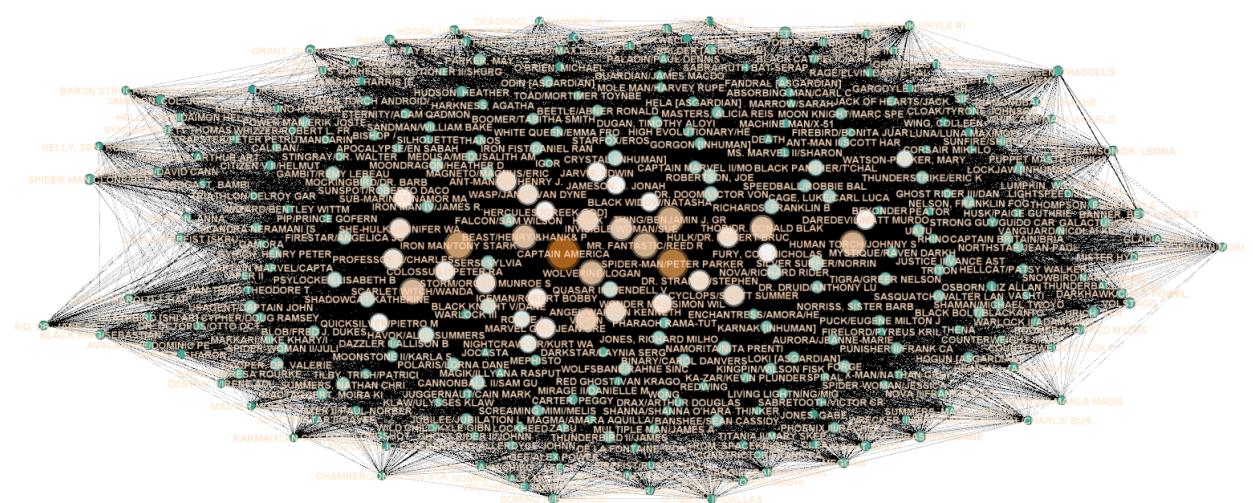
```
RETURN h1.name as Source , h2.name as Target, t.weight as Weight
```

Il client Neo4j offre una funzione di esportazione in formato .csv così da poter salvare il grafo ed importarlo su Gephi. Formattiamo il file csv creando le seguenti colonne: Source, Target, Weight in modo che Gephi lo interpreti correttamente (i nomi delle colonne sono mandatori). All'interno di Gephi, il file viene importato come "Edges table" dalla scheda "Data laboratory", specificando che il Weight è di tipo intero, e che il grafo è non orientato (è necessario effettuare tutti gli aggiornamenti necessari, o Gephi non importerà correttamente gli archi).

Ora possiamo dare una prima occhiata alla nostra rete:



un po' più zoomata:



Caratterizzazione della rete

L'analisi continua tramite la libreria networkx, richiamata comodamente dal notebook jupyter. L'importazione avviene in modo immediato tramite l'utilizzo dei pandas su csv file.

La rete importata ha un totale di 6421 nodi e 171644 archi, non è orientata. Il grafo risultante non è connesso in quanto presenta 2 componenti non connesse, con un totale di 18 nodi, con dimensioni quindi trascurabili rispetto a quelle del giant component, contenente 6403 nodi. Dal momento in cui, per applicare diverse misurazioni, tra cui la APL (necessaria per la caratterizzazione della rete) il Grafo deve essere connesso, tutte le misure sono state effettuate sul giant component.

Per verificare la tesi per cui la rete rispetti le caratteristiche Small World, si analizzano due parametri, che vengono paragonati con i corrispettivi di una rete random equivalente. L'Average Path Length è definito come il numero medio di passi necessari sul cammino più breve per tutte le possibili coppie di nodi della rete. È una misura di efficienza del trasporto di massa su una rete.

Il Clustering Coefficient definisce la misura in cui i nodi di un grafo tendono ad essere connessi fra loro. Può essere locale su un nodo (rapporto tra numero dei collegamenti tra i suoi vicini e il numero di collegamenti potenziali tra loro) oppure globale sulla totalità della rete (numero di triple chiuse fratto il numero totale di triple, somma di quelle aperte e chiuse).

Paragonando i parametri della rete in analisi con una rete randomica, si definisce small world se: Il Clustering Coefficient è molto maggiore sulla rete analizzata (anche di diversi ordini di grandezza); l'Average Path Length deve essere maggiore, ma rimanendo nello stesso ordine di grandezza.

Di seguito il risultato delle misurazioni:

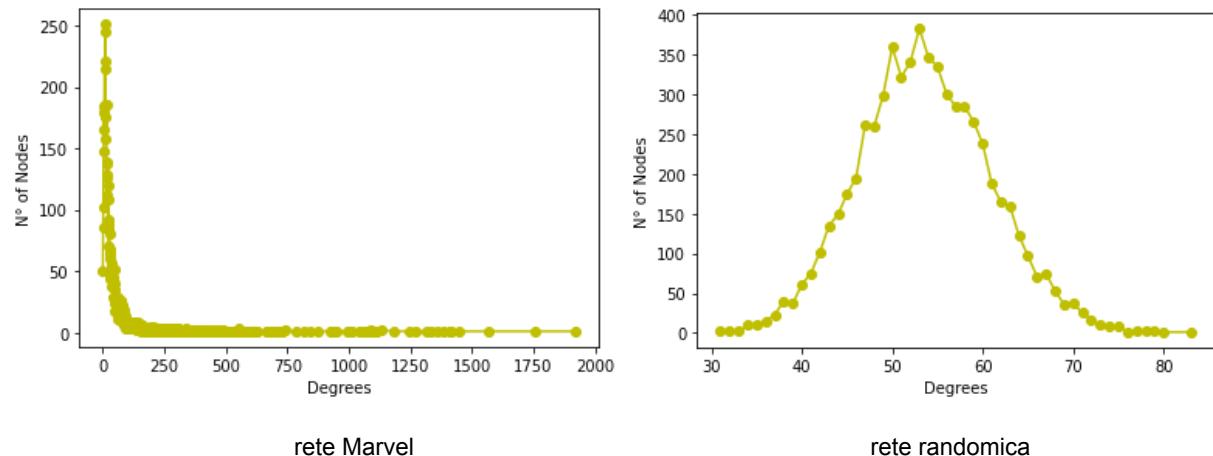
	Rete originale	Rete random equivalente
CC	0.7739	0.0083
APL	2.630	2.627
Densità	0.008371778634107343	0.008371778634107343
Diametro	5	3

Nella tabella possiamo osservare le misurazioni di diversi parametri, di seguito analizzati.

Densità: definito come rapporto tra il numero di archi E e il numero di possibili archi in una rete con N nodi. Per definizione, le due densità rilevate coincidono.

Diametro: possiamo definire il diametro di una rete come il più lungo di tutti i cammini minimi. È la distanza più breve tra i due nodi più distanti della rete. È una misura importante in quanto esplicita quanto velocemente un'informazione si diffonderà attraverso la rete e anche quanto è probabile che siano integrati diversi componenti nella rete.

Analizziamo la distribuzione del degree dei nodi delle due reti:



I collegamenti tra i nodi della rete randomica vengono stabiliti in modo casuale per cui la maggior parte dei nodi avrà lo stesso numero di collegamenti. La distribuzione di questi collegamenti può essere descritta tramite una distribuzione di Poisson per cui la probabilità che un nodo sia connesso d'altri K nodi decresce esponenzialmente al crescere di K , Notiamo quindi l'assenza di nodi altamente connessi.

La distribuzione dei collegamenti della rete Small World segue una power law $P(k) \sim k^{-z}$. La probabilità diminuisce al crescere del valore di k , ma rimane diversa da 0 anche per valori molto grandi. Si creano quindi nodi molto connessi (detti anche hub) e molti nodi poco connessi. La rete in analisi è una Scale-Free Network, con i suoi vantaggi e svantaggi: robusta rispetto ai guasti ma vulnerabile ad attacchi mirati. Riadattato al nostro contesto, sicuramente un super-cattivo potrebbe trovare più interessante "far sparire" il gruppo di eroi con degree più alto. O magari un personaggio in difficoltà potrebbe rivolgersi agli stessi per ottenere un soccorso più corposo ed immediato.

Centralità dei nodi

Utilizzeremo una serie di indicatori di centralità, per evidenziare i nodi di maggiore "importanza", che vedremo essere una definizione molto relativa. In queste misurazioni non è stata considerata la misura PageRank in quanto applicabile solo su grafi orientati.

Degree Centrality (nodi influenti)

Questa è la misura di centralità più semplice da calcolare. Calcola localmente il rapporto tra il numero di vicini di ogni nodo e il numero massimo di vicini che potrebbe avere. In un grafo non orientato, questo è $n-1$ dove n è il numero totale di nodi del grafo. Può essere usata per trovare individui molto connessi, popolari, che probabilmente detengono la maggior parte delle informazioni o che possono connettersi rapidamente con la rete più ampia.

Di seguito i 10 nodi con punteggio più alto.

CAPTAIN AMERICA	0.29975007810059356
SPIDER-MAN/PETER PARKER	0.2739768822243049
IRON MAN/TONY STARK	0.24461105904404873
THING/BENJAMIN J. GR	0.22617931896282412
MR. FANTASTIC/REED R	0.2211808809746954
HUMAN TORCH/JOHNNY S	0.22055607622617932
WOLVERINE/LOGAN	0.21587004061230863
SCARLET WITCH/WANDA	0.21274601686972822
BEAST/HENRY HANK P	0.2085285848172446
THOR/DR. DONALD BLAK	0.20540456107466415

Betweenness centrality

La betweenness Centrality di un nodo v è la somma della frazione dei cammini minimi di tutte le coppie che passano per v , definita da questa formula:

$$c_B(v) = \sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)}$$

Dove V è l'insieme dei nodi, $\sigma(s,t)$ è il numero di cammini minimi(s, t) e $\sigma(s,t|v)$ è il numero di cammini minimi che passano attraverso v .

Indica quali nodi hanno maggior probabilità di trovarsi sui path di comunicazione tra altri nodi, e la capacità di influenzare il flusso delle comunicazioni.

Di seguito i 10 nodi con punteggio più alto.

SPIDER-MAN/PETER PARKER	0.07204460717982197
CAPTAIN AMERICA	0.05596671589099814
IRON MAN/TONY STARK	0.03770633619637516
WOLVERINE/LOGAN	0.03519963458219211
HAVOK/ALEX SUMMERS	0.0351728898799486
DR. STRANGE/STEPHEN	0.029305009293613304
THING/BENJAMIN J. GR	0.025048027049570366
HAWK	0.024556061663952355
HULK/DR. ROBERT BRUC	0.023833890997933878
MR. FANTASTIC/REED R	0.02365612173366362

Closeness centrality (broadcasters)

La Closeness centrality di un nodo u è il reciproco della somma delle distanze di cammino più brevi da u a tutti gli altri $n-1$ nodi. La somma delle distanze dipende dal numero di nodi nel grafico, la closeness è normalizzata dalla somma delle distanze minime possibili $n-1$.

$$C(u) = \frac{n - 1}{\sum_{v=1}^{n-1} d(v, u)}$$

Dove $d(v, u)$ è la distanza del cammino minimo tra v ed u , ed n è il numero di nodi nel grafico. Notare che valori di prossimità più elevati indicano una centralità maggiore. Può essere usata per trovare gli individui che sono nella posizione migliore per influenzare l'intera rete più rapidamente. Può aiutare a trovare buoni "broadcasters", ma in una rete altamente connessa, tutti i nodi hanno un punteggio simile (all'interno di un cluster è meglio usare la Closeness).

CAPTAIN AMERICA	0.5861027190332326
SPIDER-MAN/PETER PARKER	0.5767048013692461
IRON MAN/TONY STARK	0.565248101712873
THING/BENJAMIN J. GR	0.5613327487943884
MR. FANTASTIC/REED R	0.5595175668589407
HUMAN TORCH/JOHNNY S	0.5590778097982709
WOLVERINE/LOGAN	0.5570831883049078
SCARLET WITCH/WANDA	0.5561636695334897
BEAST/HENRY & HANK & P	0.5545738045738046
THOR/DR. DONALD BLAK	0.5537104307213285

Eigenvector centrality

Assegna punteggi in base a quanto sono connessi i nodi con cui un nodo è connesso. Un nodo con una alta Eigenvector Centrality è connesso con nodi aventi a loro volta un alto punteggio di centralità. Calcolando le connessioni estese di un nodo, la Eigenvector Centrality è in grado di identificare i nodi con influenza su tutta la rete, non solo quelli ad essa direttamente collegati.

CAPTAIN AMERICA	0.11423433119544253
IRON MAN/TONY STARK	0.10189063956949355
THING/BENJAMIN J. GR	0.10016271122458334
SCARLET WITCH/WANDA	0.09990996942577986
MR. FANTASTIC/REED R	0.09930617262050313
HUMAN TORCH/JOHNNY S	0.09895302348665509
SPIDER-MAN/PETER PARKER	0.09891423307668402
BEAST/HENRY HANK P	0.09744838912431246
WOLVERINE/LOGAN	0.09682293862280675
VISION	0.09603059176613907

I nodi più importanti per ogni centralità

Possiamo notare come alcuni nomi siano ricorrenti in tutte e quattro le misure di centralità qui analizzate. È utile tracciare le relazioni tra i punteggi di centralità per vedere la loro correlazione e capire fino a che punto forniscono informazioni ridondanti o meno. Possiamo farlo facilmente usando Panda per la creazione delle tabelle e Seaborn per il plotting.

Ordinate secondo degree

	degree	betweenness	closeness	eigenvector
CAPTAIN AMERICA	0.299750	0.055967	0.586103	0.114234
SPIDER-MAN/PETER PARKER	0.273977	0.072045	0.576705	0.098914
IRON MAN/TONY STARK	0.244611	0.037706	0.565248	0.101891
THING/BENJAMIN J. GR	0.226179	0.025048	0.561333	0.100163
MR. FANTASTIC/REED R	0.221181	0.023656	0.559518	0.099306
HUMAN TORCH/JOHNNY S	0.220556	0.021661	0.559078	0.098953
WOLVERINE/LOGAN	0.215870	0.035200	0.557083	0.096823
SCARLET WITCH/WANDA	0.212746	0.016772	0.556164	0.099910
BEAST/HENRY & HANK & P	0.208529	0.022296	0.554574	0.097448
THOR/DR. DONALD BLAK	0.205405	0.023319	0.553710	0.093233

Ordinate secondo betweenness

	degree	betweenness	closeness	eigenvector
SPIDER-MAN/PETER PARKER	0.273977	0.072045	0.576705	0.098914
CAPTAIN AMERICA	0.299750	0.055967	0.586103	0.114234
IRON MAN/TONY STARK	0.244611	0.037706	0.565248	0.101891
WOLVERINE/LOGAN	0.215870	0.035200	0.557083	0.096823
HAVOK/ALEX SUMMERS	0.122462	0.035173	0.524969	0.066517
DR. STRANGE/STEPHEN	0.169947	0.029305	0.539843	0.082336
THING/BENJAMIN J. GR	0.226179	0.025048	0.561333	0.100163
HAWK	0.184630	0.024556	0.543325	0.087240
HULK/DR. ROBERT BRUC	0.168229	0.023834	0.542450	0.084981
MR. FANTASTIC/REED R	0.221181	0.023656	0.559518	0.099306

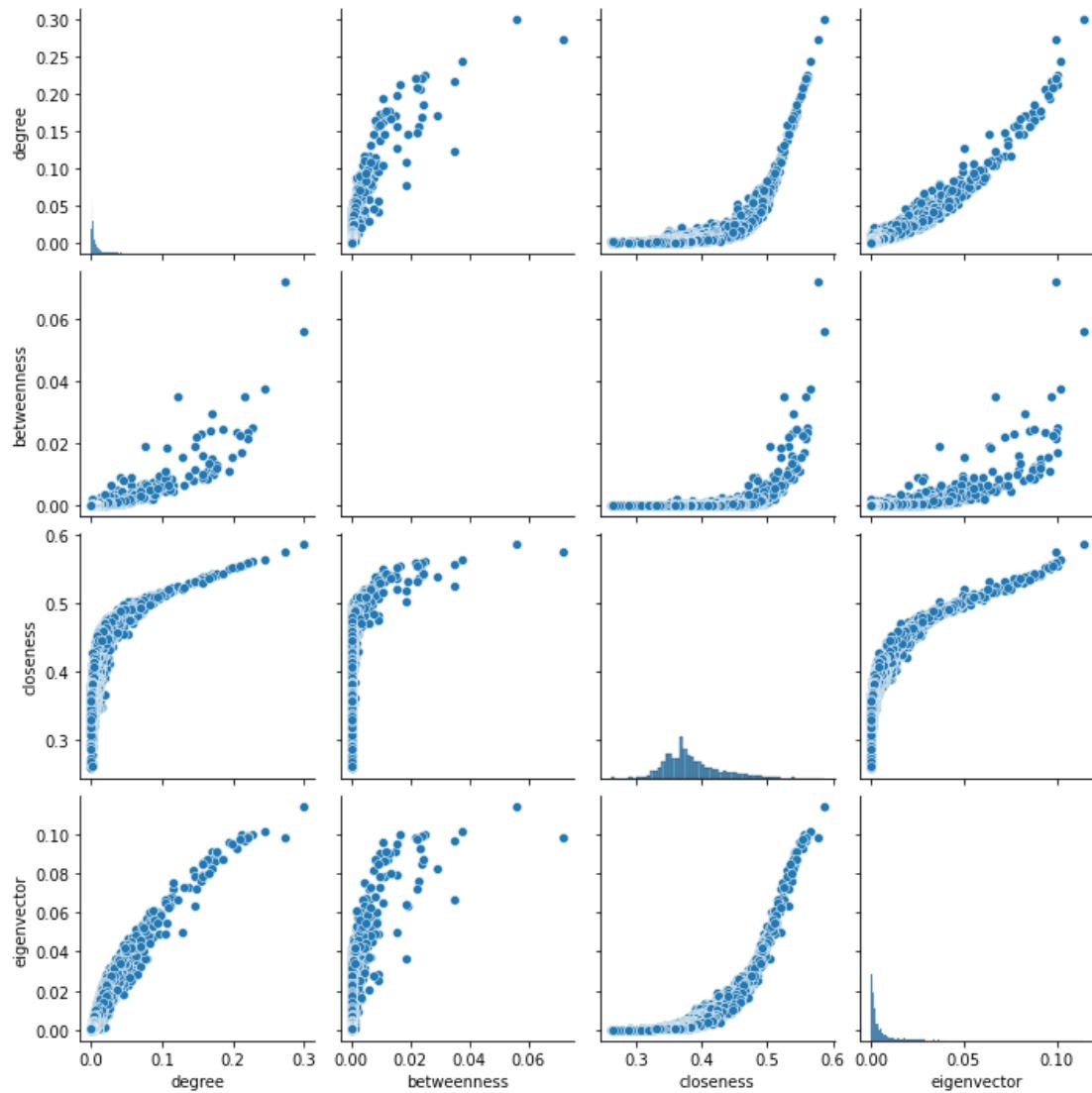
Ordinate secondo closeness

	degree	betweenness	closeness	eigenvector
CAPTAIN AMERICA	0.299750	0.055967	0.586103	0.114234
SPIDER-MAN/PETER PARKER	0.273977	0.072045	0.576705	0.098914
IRON MAN/TONY STARK	0.244611	0.037706	0.565248	0.101891
THING/BENJAMIN J. GR	0.226179	0.025048	0.561333	0.100163
MR. FANTASTIC/REED R	0.221181	0.023656	0.559518	0.099306
HUMAN TORCH/JOHNNY S	0.220556	0.021661	0.559078	0.098953
WOLVERINE/LOGAN	0.215870	0.035200	0.557083	0.096823
SCARLET WITCH/WANDA	0.212746	0.016772	0.556164	0.099910
BEAST/HENRY & HANK & P	0.208529	0.022296	0.554574	0.097448
THOR/DR. DONALD BLAK	0.205405	0.023319	0.553710	0.093233

Ordinate secondo Eigenvector

	degree	betweenness	closeness	eigenvector
CAPTAIN AMERICA	0.299750	0.055967	0.586103	0.114234
IRON MAN/TONY STARK	0.244611	0.037706	0.565248	0.101891
THING/BENJAMIN J. GR	0.226179	0.025048	0.561333	0.100163
SCARLET WITCH/WANDA	0.212746	0.016772	0.556164	0.099910
MR. FANTASTIC/REED R	0.221181	0.023656	0.559518	0.099306
HUMAN TORCH/JOHNNY S	0.220556	0.021661	0.559078	0.098953
SPIDER-MAN/PETER PARKER	0.273977	0.072045	0.576705	0.098914
BEAST/HENRY & HANK & P	0.208529	0.022296	0.554574	0.097448
WOLVERINE/LOGAN	0.215870	0.035200	0.557083	0.096823
VISION	0.194158	0.010944	0.549622	0.096031

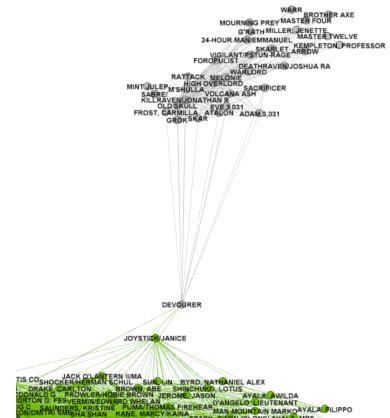
Notiamo subito un allineamento sui risultati della degree e della closeness centrality. Captain America risulterebbe essere uno dei leader sotto tutti i punti di vista, a meno della betweenness centrality, per cui è in testa, di diversi punti, Spider Man. Per cui sicuramente, per ottenere una comunicazione efficiente, sarebbe più indicato affidarsi a lui.



Bassa betweenness e alta eigenvector (cella 4:2) - Possiamo osservare che diversi nodi presentano una bassa betweenness (quindi un nodo dal quale non passano molte informazioni) e una alta eigenvector (quindi sono collegate a nodi a loro volta importanti).

Alta closeness - Bassa Betweenness (cella 3:2) - Abbiamo nodi topologicamente posizionati all'interno di un nucleo centrale della rete, tenendo conto della loro elevata closeness. La bassa betweenness centrality indica che però molti dei loro archi connettono nodi che erano già connessi tra loro. Probabilmente questa situazione si è creata a causa di percorsi multipli nella rete, nelle zone più "dense". Dimostriamo così una **buona resistenza "agli attacchi"** tipica delle reti Scale Free.

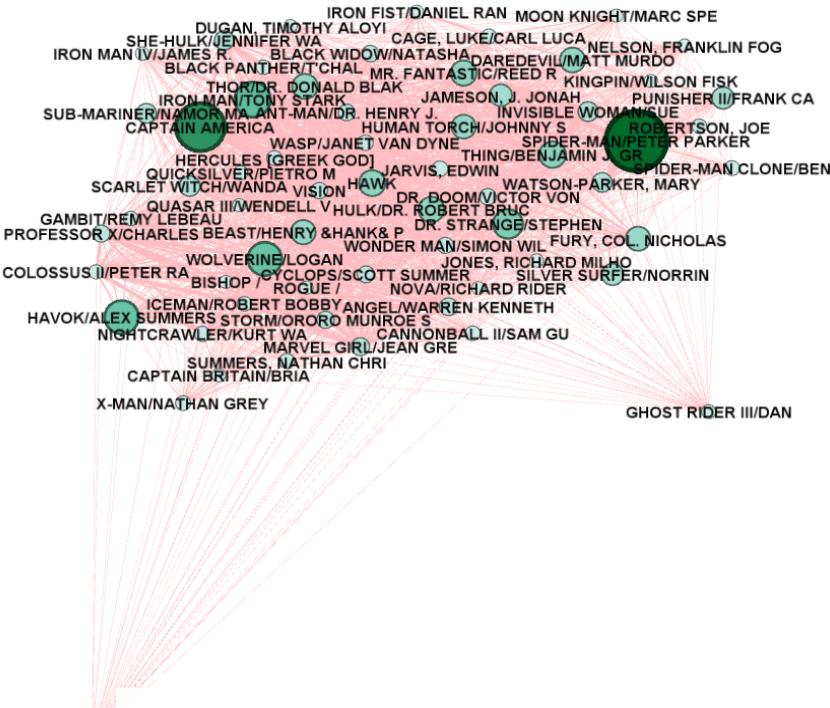
Alta closeness - Bassa Degree (cella 1:3) - I nodi con una closeness elevata hanno le distanze più brevi da tutti gli altri nodi. Contemporaneamente però sono collegati ad un numero relativamente basso di nodi, per cui si ipotizza che i nodi in questione siano collegati a nodi con un alto grado che sono a loro volta collegati a nodi diversi tra loro. Questi potrebbero essere i nodi che connettono tra loro due componenti (**articulation point**).



Visualizzazione tramite Gephi

Per una visualizzazione di queste misure ci spostiamo sullo strumento Gephi. Ci appoggiamo ad un layout di visualizzazione “force-based” in modo da avere vicini tra loro nodi connessi, e lontani quelli non connessi. Per cui è stato usato l'algoritmo Force Atlas 2 con uno Scaling molto alto, e perfezionato tramite il layout Label Adjust.

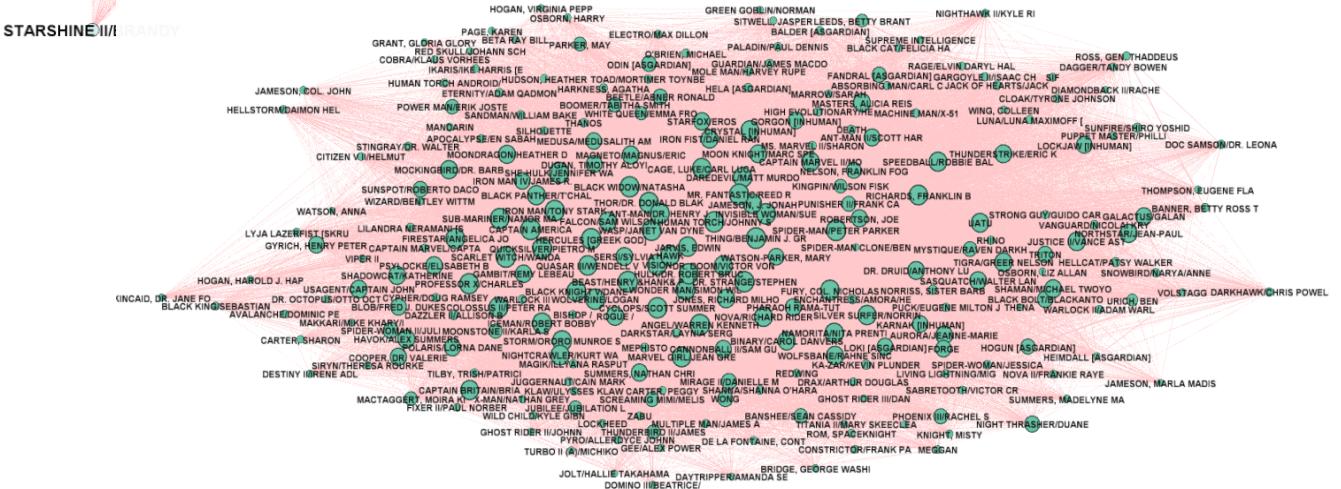
Dal momento in cui la rete analizzata è molto grande, per facilitare la visualizzazione omettiamo i nodi che hanno misure al di sotto di un certa soglia. Il colore e la dimensione dei nodi saranno rispettivamente più scuro e più grande relativamente alla misura analizzata



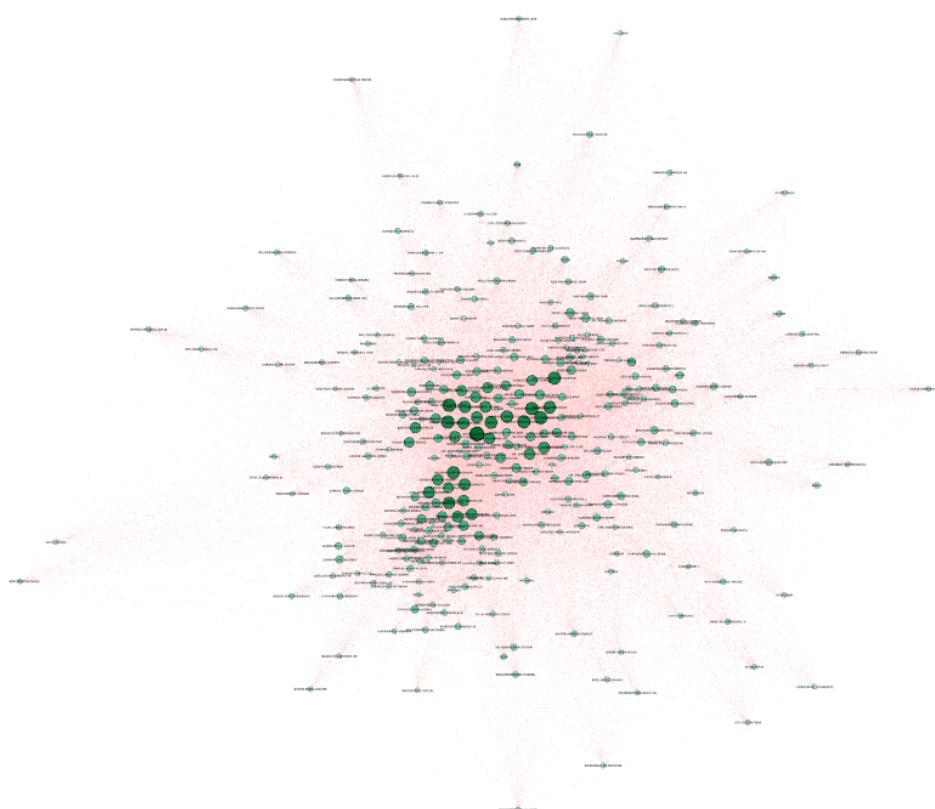
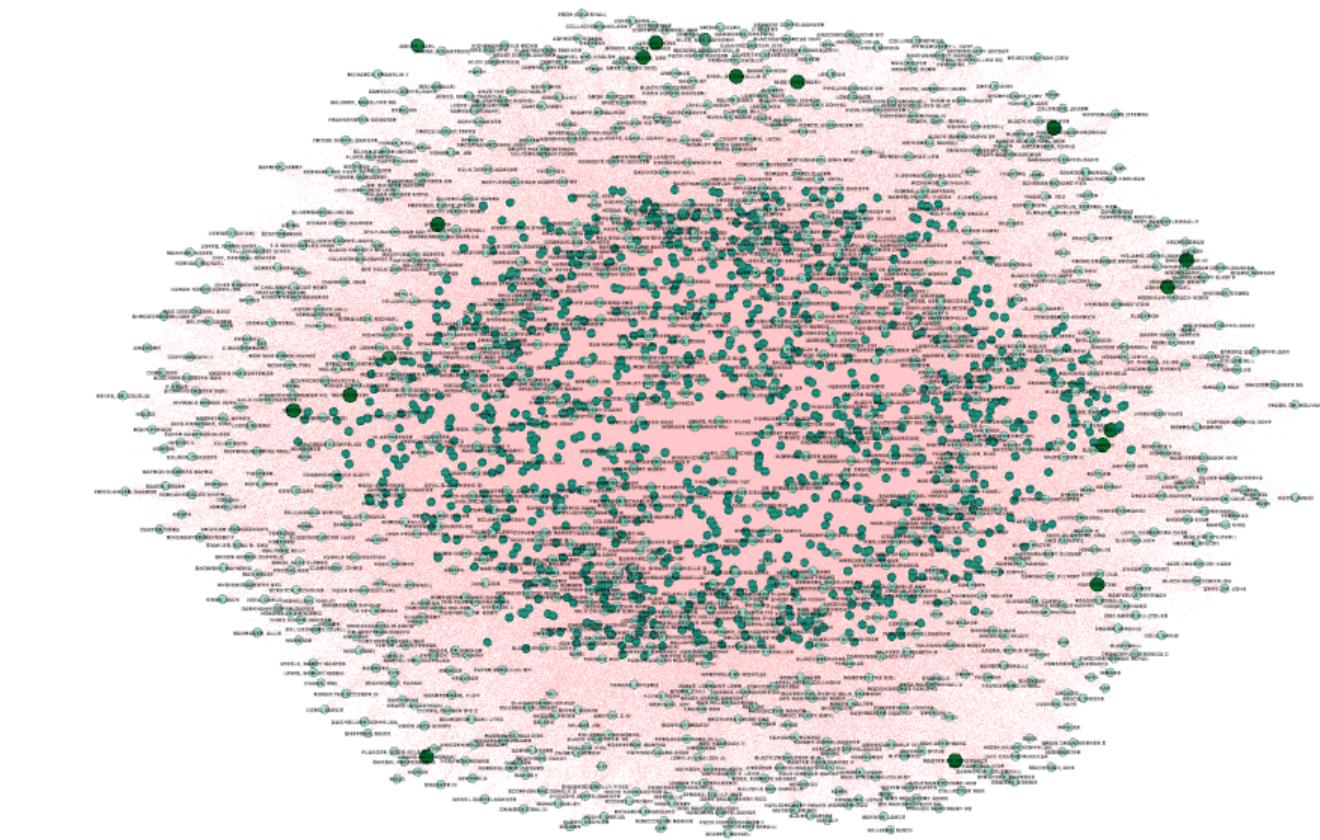
Betweenness centrality

Soglia minima: 0.005

Qui abbiamo invece una visualizzazione basata sul **degree**, su una sottorete comprendente nodi dal degree minimo di 200.



Visualizzazione basata sulla **closeness** centrality, senza soglia minima. Possiamo notare che i nodi con un alto punteggio sono numerosi.



In questo caso la visualizzazione dipende dalla misura **eigenvector** centrality con soglia minima 0.200.

Community Detection

Ricerchiamo adesso la presenza di comunità all'interno del grafo. Dal momento in cui non disponiamo di un ground truth sarà necessario utilizzare una serie di metriche dette metriche di validazione interna per la bontà dei risultati (in contrapposizione ad altre misure come Square Deviation, Adjusted Rand Score, Normalized Mutual Information, dette esterne). Per poter visualizzare le reti nel dettaglio dopo la clusterizzazione, è necessario lo strumento Gephi. È necessario prima aggiungere la label del cluster di ogni nodo come attributo tramite la funzione `set_node_attributes(G, values, name=None)` di Networkx, per poi esportare il grafo nel formato `.gexf` tramite la funzione `write_gexf()`. Una volta fatto ciò, è possibile importare il grafo su Gephi per una visualizzazione più esplicita.

Le metriche

Silhouette index - max[-1, +1]

Misura di **similarità interna di un nodo relativamente al cluster che lo contiene**. Il valore cade nell'intervallo [-1, 1]: Valori alti indicano una buona similarità, indice di una buona clusterizzazione. Se il valore di alcuni nodi è molto basso o negativo, potrebbe esserci un numero di cluster eccessivo o scarso. Quando si ha un valore prossimo allo zero vi è sovrapposizione di clusters.

L'indice è definito da:

b(i): distanza media minima tra i e ogni nodo di altri clusters.

a(i) distanza media tra i e nodi dello stesso cluster

è stata usata la funzione `silhouette_score` di scikit learn

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Calinski-Harabasz Index - max [0, +∞]

Detto anche Criterio del rapporto di varianza, è il rapporto tra la somma della dispersione tra cluster e della dispersione tra cluster per tutti i cluster. Maggiore è il punteggio, migliori sono le prestazioni. Il punteggio è più **alto quando i cluster sono densi e ben separati**, il che si riferisce a un concetto standard di cluster.

Davies Bouldin Score - min [0, +∞]

Questo indice indica la "somiglianza" media tra i cluster, dove la somiglianza è una misura che confronta la **distanza tra i cluster e la dimensione dei cluster stessi**. Un indice Davies-Bouldin più basso si riferisce a un modello con una migliore separazione tra i cluster (punteggio minimo zero). Se due cluster sono vicini ma hanno una grande area, questo rapporto sarà grande, indicando che questi cluster non sono molto distinti. È stata utilizzata la funzione `davies_bouldin_score()` di SciKit Learn.

Dunn Index - max[0, +∞]

Misura di validazione interna per il clustering, calcolata secondo il **rapporto tra minima distanza intra cluster e il diametro massimo della rete**. La bontà del clustering è proporzionale all'indice.

Algoritmi di clustering

K-means clustering

Algoritmo semplice ed efficace, si basa sulla distanza tra i nodi, per cui è necessaria una matrice di adiacenza del grafo. Si assegna un numero K di centroidi randomici. Dopo di che si ripetono i seguenti passaggi fino a quando ogni nodo viene assegnato ad un cluster.

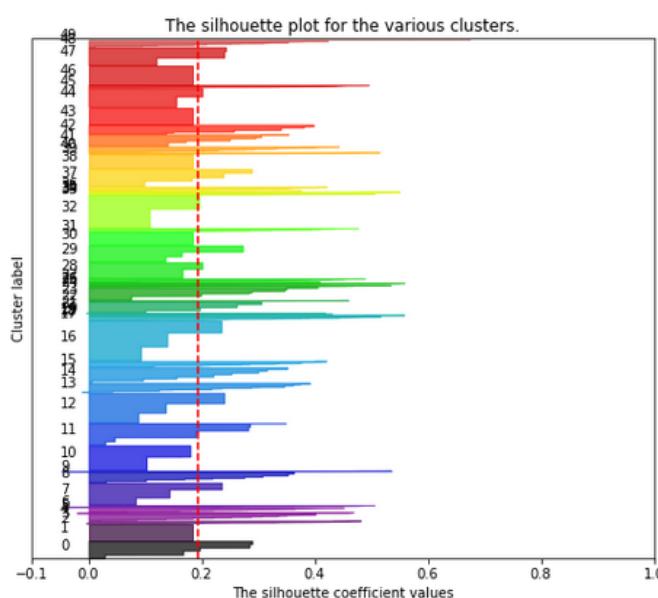
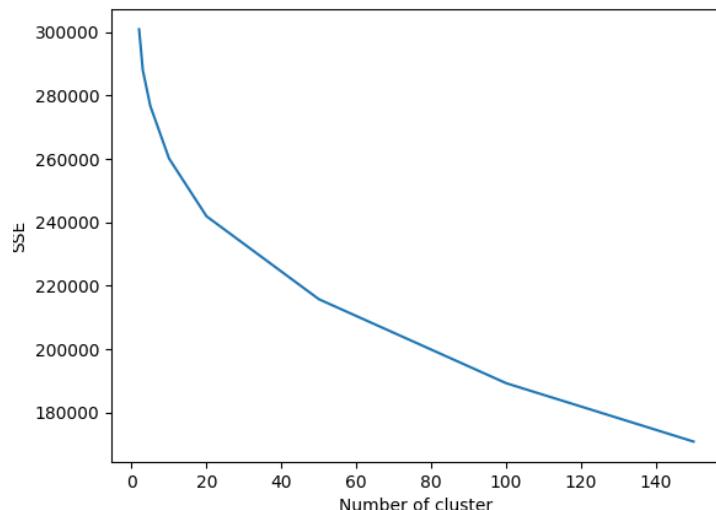
1. un punto viene assegnato al centroide più vicino
2. si riassegnano i centroidi a seconda del **punto il più vicino al centro** del cluster

Il numero di centroidi K deve essere noto. Inoltre la scelta dei centroidi iniziali può influire sulla qualità della soluzione finale, per cui necessitano multipli run con la scelta della soluzione migliore. La funzione KMeans() è resa disponibile dalla libreria Scikit Learn.

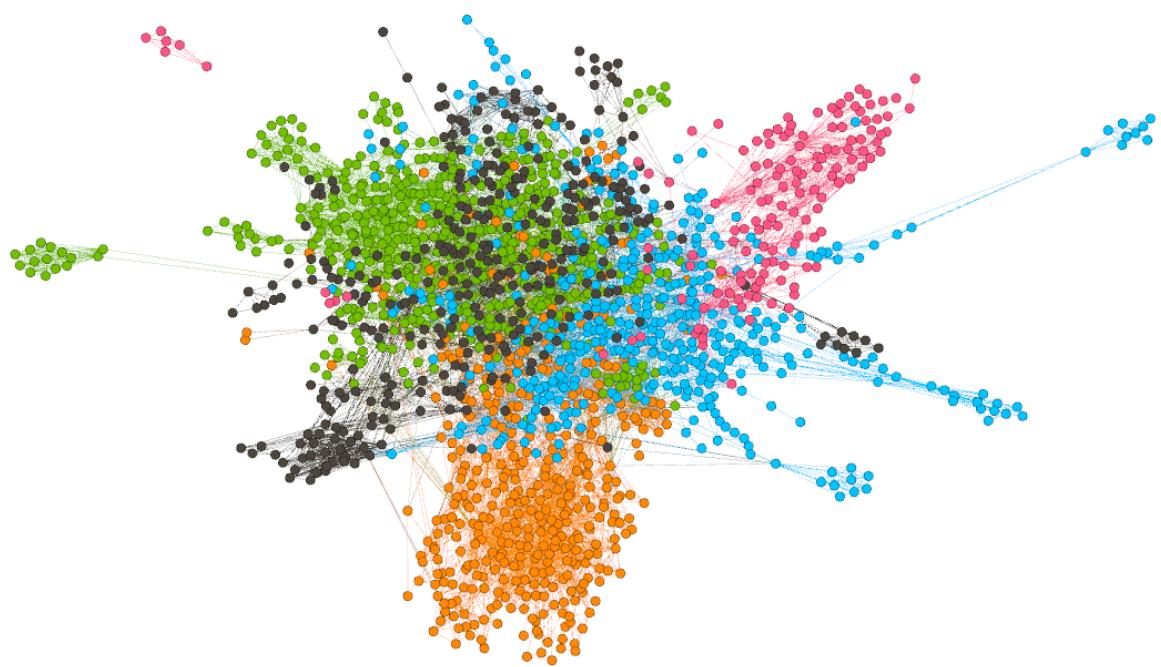
L'**Elbow Criterion** è un metodo aggiuntivo per valutare l'algoritmo K-means. Per usarlo dobbiamo calcolare l' SSE (sum of square errors) tramite la funzione KMeans.inertia_. L'idea è quella di scegliere il numero di cluster k per cui l'SSE è minimizzato.

Calcolata la SSE per ogni k, si traccia il grafico, per cui si osserva che la misura tende a diminuire man mano che aumentiamo k (SSE=0, quando k è uguale al numero di nodi del grafo, perché allora ogni punto è cluster di se stesso, riportando errore nullo).

Quindi l'obiettivo è scegliere un piccolo valore di k che ha già un SSE basso, e di solito rappresenta il gomito, dove iniziamo ad avere rendimenti decrescenti aumentando k, in questo caso k=50.



Silhouette_score: 0.0320
Davies bouldin score: 2.3274
Calinski-Harabaz score: 70.4412
Dunn index : 0.029



Agglomerative clustering

Algoritmo di clustering gerarchico, produce un set di cluster innestati. I due approcci al clustering gerarchico sono: agglomerativo (inserisce ogni nodo in un cluster diverso per poi accorpate quelli con distanza minima) o divisivo (parte da un unico cluster che viene poi diviso a seconda della massima distanza inter clustering). È stata usata la funzione AgglomerativeClustering di SciKitLearn su un range di interi per il numero di cluster.

Per questo algoritmo deve essere specificato il criterio di collegamento, ovvero la distanza specificata come funzione di distanze tra gli elementi del cluster. I parametri previsti sono i seguenti e definiscono le distanze in questo modo:

- Complete linkage: massima distanza tra due punti dei cluster
- Average linkage: la la distanza media di ogni punto dei due cluster
- Ward linkage: valore ottimo della funzione obiettivo (SSE)

Ward :

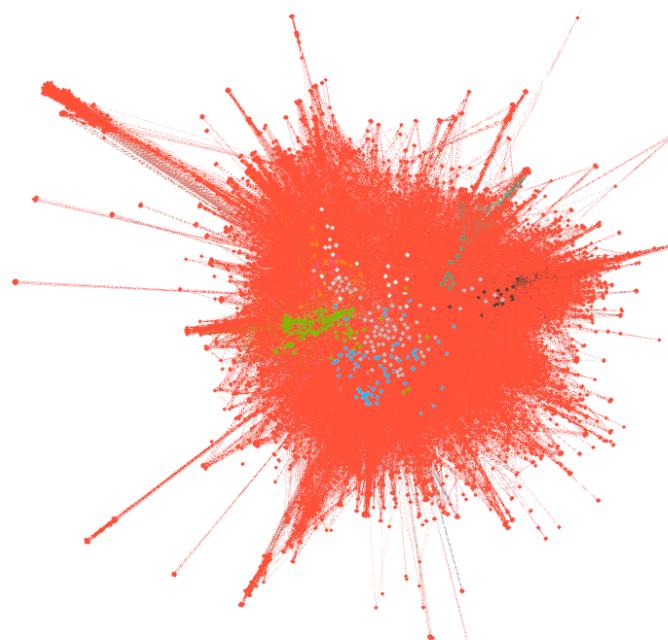
n_clusters	Silhouette score (max 1)	davies bouldin (min)	Calinski-Harabaz (max)	dunn index (max)
50	0.0234205	2.6862	68.8051	0.0565
100	0.0197	2.222	48.23	0.0606
150	0.0065	2.05406	40.133	0.06428

Average

n_clusters	Silhouette score (max 1)	davies bouldin (min)	Calinski-Harabaz (max)	dunn index (max)
50	0.6199	0.6571	6.494	0.5242
100	0.545	0.5853	22.3494	0.3946
150	0.5029	0.6085	19.4673	0.3682

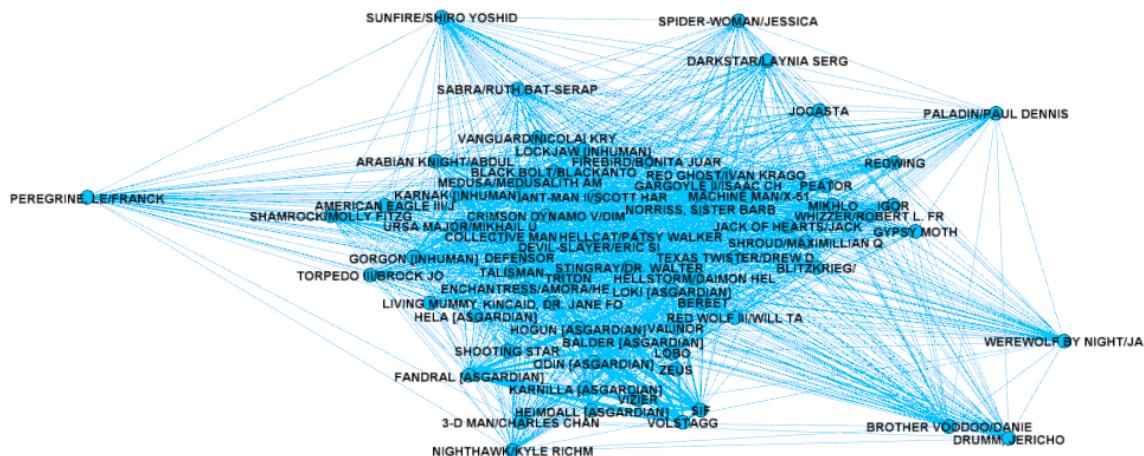
Complete

n_clusters	Silhouette score (max 1)	davies bouldin (min)	Calinski-Harabaz (max)	dunn index (max)
50	0.4058	1.1636	41.4923	0.2454
100	0.3055	1.0228	31.4360	0.1695
150	0.2919	1.0310	27.301	0.1930

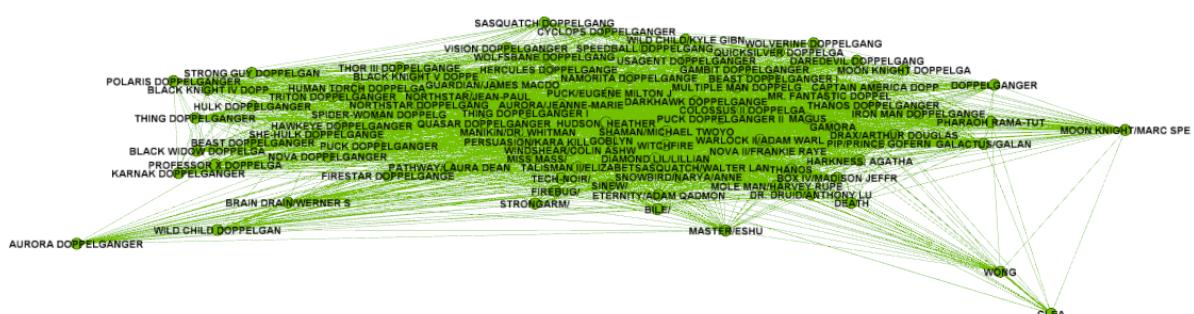


Si visualizza di seguito il clustering con linkage model Average, 50 clusters: Il numero di clusters è sottodimensionato per la rete, in quanto all'interno di un solo clusters sono presenti il 90% dei nodi. Questo problema di è dato dalla potenza computazionale non adeguata, che non mi ha permesso di fare computazioni su grandi numeri. Nonostante ciò, i clusters piccoli sono molto precisi. Di seguito due esempi:

Nel sottografo azzurro possiamo apprezzare una perfetta clusterizzazione del gruppo degli inumani, capitanati da Freccia Nera.



Nel sottografo verde apprezziamo la clusterizzazione del team doppelganger (in cui è presente anche un eroe particolare, ovvero quiksilver)



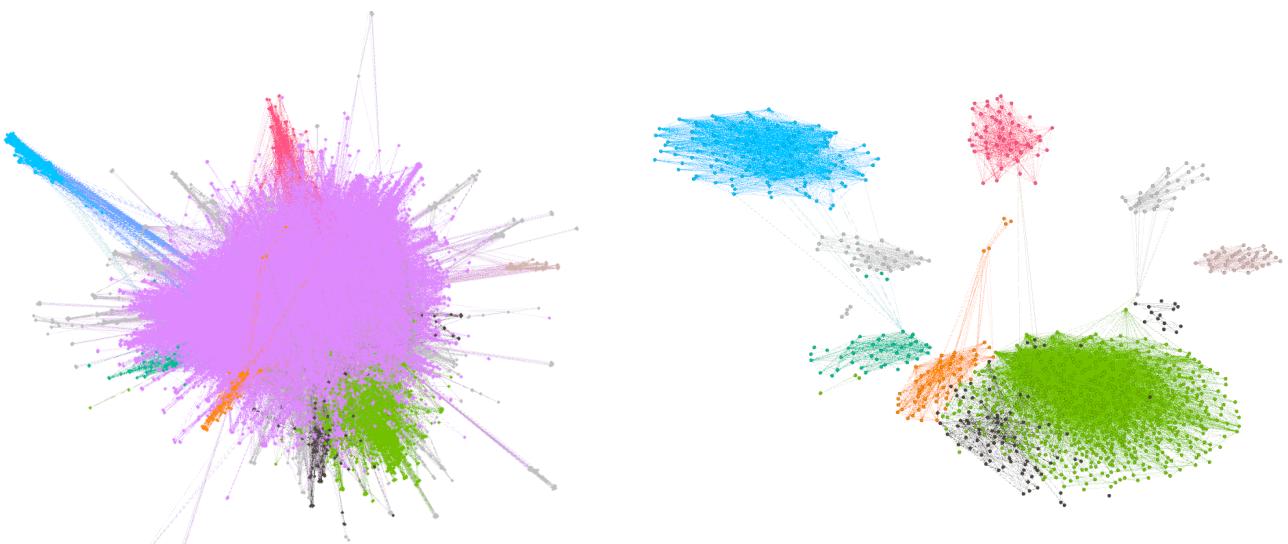
Spectral Clustering

Algoritmo che utilizza **il set di autovalori della matrice di similarità** per effettuare una riduzione di dimensionalità prima di partizionare il grafo. Presenta le stesse problematiche del metodo k-means (per cui la necessità che il numero di clusters sia noto)

È stata utilizzata la funzione `SpectralClustering()` di SciKit Learn, utilizzando un range di interi per il parametro `n_clusters`.

Il valore si silhouette è negativo per un numero di clusters fino a 500, evidenziando una scarsità di clusters. Il davies bouldin score (ottimo se vicino allo 0) rimane costante (circa 2) fino a 500, indica che i cluster non sono tra loro ben distinti. Il Calinski-Harabasz Index ha valori alti dai 2 ai 200 clusters, per poi decrescere, indicando clusters densi. Il Dunn index, che deve essere massimizzato, tiene dei valori molto bassi fino ai 500 clusters.

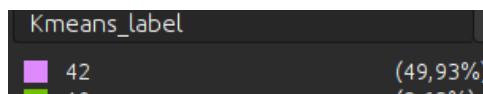
n_clusters	Silhouette score (max 1)	davies bouldin (min)	Calinski-Harabaz (max)	dunn index (max)
50	-0.2472	2.3777	6.1690	0.0390
100	-0.2044	2.3421	5.0594	0.0225
150	-0.1614	2.3005	4.6585	0.0226
500	0.0201	2.2723	2.2718	0.0313



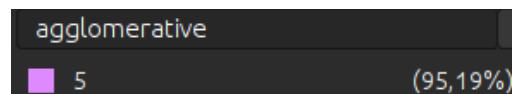
Reclustering

È stato applicato il reclustering di una parte dei risultati, quando si presentava un componente gigante. Per ogni algoritmo, sono stati esportati tramite gephi i soli nodi appartenenti alla community più grande con i relativi archi, creando in questo modo una sotto rete. Su questa rete è stato ri applicato l'algoritmo. Si espongono i risultati ottenuti sugli algoritmi:

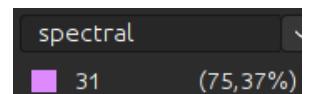
K-means



Agglomerative

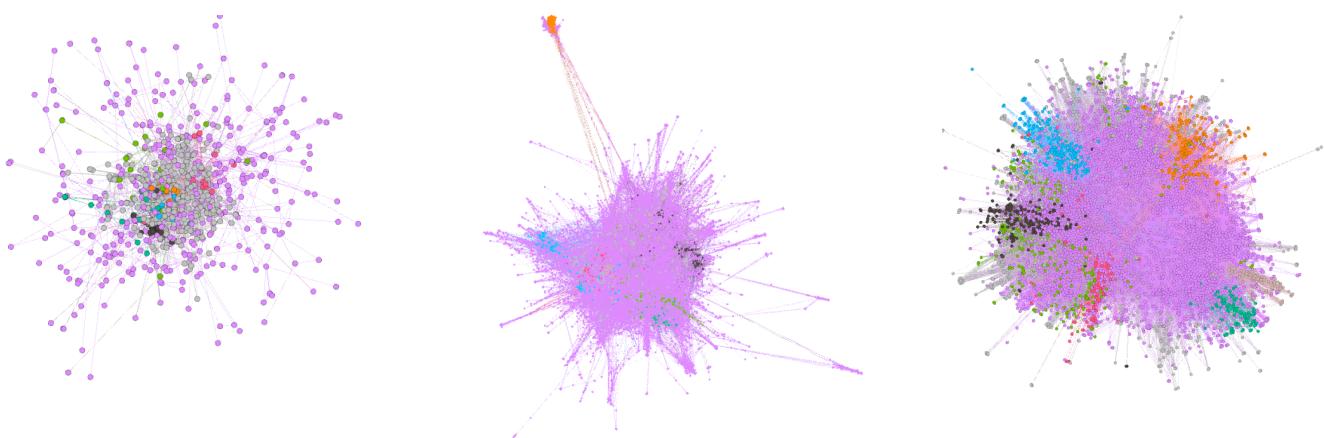


Spectral



	K	Silhouette score (max 1)	davies bouldin (min)	Calinski-Harabaz (max)	dunn index (max)	dim max cluster %
kmeans	50	0.0320	2.3274	70.4412	0.029	49
reclustered	50	0.236	1.34	23.16	0.166	42
	100	0.323	1.224	20.229	0.185	28
Agglomerative	50	0.6199	0.6571	6.494	0.5242	95
reclustered	50	0.786	0.445	21.824	0.558	89
	100	0.639	1.322	17.987	0.16	83
	400	0.1228	0.943	13.786	0.216	63
Spectral	50	-0.2472	2.3777	6.1690	0.039	75
reclustered	50	-0.255	2.564	5.978	0.0332	66
	100	-0.209	2.405	3.975	0.023	63
	600	0.046	2.177	1.511	0.032	39

I punteggi tendono a migliorare, ma si nota che nonostante si lavori su un grafo con un numero di nodi più contenuto, gli algoritmi hanno la tendenza di creare una community che prevale numericamente sulle altre. (Kmeans: 46%, Agglomerative: 89%, Spectral 66%).



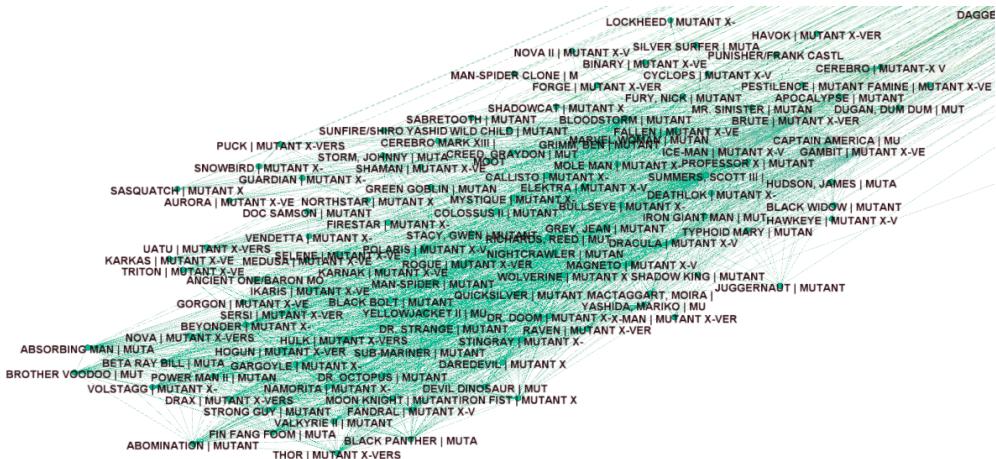
Affinity propagation

Algoritmo basato sul concetto di affinità tra nodi. Itera con message passing: ogni nodo si propone come exemplar del proprio cluster. Non richiede di sapere in anticipo il numero di clusters. Prende in input la matrice di similarità del sistema, identifica gli exemplar, i messaggi vengono scambiati fino a quando non si trovano gli exemplar ottimali. Sono calcolate delle matrici aggiuntive:

- responsibility matrix: indica quanto un nodo è adatto ad essere exemplar
 - availability matrix: quanto è appropriato per un elemento scegliere k come exemplar
 - criterion matrix: Data dalle somme delle altre due matrici. l'elemento col più alto valore per ogni riga viene designato come exemplar.

n_clusters	Silhouette score (max 1)	davies bouldin (min)	Calinski-Harabaz (max)	dunn index (max)
142	-0.2487	5.303	4.3931	0.0228

Da questa visualizzazione di Gephi possiamo evidenziare come questo algoritmo sia stato particolarmente efficiente nel clustering della comunità di mutanti che appaiono nel fumetto Uncanny X-Men (tra cui appare nuovamente il nostro **quicksilver**).



Clustering Ensemble

Questa tecnica consiste nell'applicazione di un set di L algoritmi di clustering in questo modo: Si combinano i set di L soluzioni ottenute in una matrice di co-occorrenza, in cui ogni cella a_{ij} rappresenta il numero di soluzioni per cui i vertici v_i e v_j sono nello stesso cluster, diviso per il numero totale di soluzioni L. Viene poi applicato un ultimo step di clustering utilizzando la matrice di co-occorrenza come input. In questo modo viene garantita una maggiore robustezza nella soluzione.

Gli algoritmi che sono stati agglomerati sono K-Means, Agglomerative Clustering, Spectral Clustering, tutti con un valore di clustering pari a 50. Di seguito visto il poco miglioramento (causato probabilmente dalle etichette dello spectral clustering) è stato effettuato un ensemble tra K-Means e Agglomerative Clustering. i punteggi sono di poco migliorati ma comunque meno di quanto atteso.

n_clusters	Silhouette score (max 1)	davies bouldin (min)	Calinski-Harabaz (max)	dunn index (max)
Spectral	-0.2472	2.3777	6.1690	0.0390
Agglomerative	0.6199	0.6571	6.494	0.5242
K-means	0.0320	2.3274	70.4412	0.029
Ensemble-3	-0.115	8.700	23.1165	0.0227
Ensemble-2	0.143	10.8216	23.1262	0.0226

In compenso è possibile apprezzare cluster molto ben equidistribuiti, senza una componente che ne inglobi la maggior parte, come era invece evidentemente nei clustering precedenti.



Community Detection

Coverage: è definita come il rapporto tra il numero di archi intra-comunità e il numero totale di archi appartenenti al grafo. La misura coverage di una partizione in cui tutte le comunità sono sconnesse l'una dall'altra è 1, poiché tutti gli archi ricadono all'interno delle comunità. Intuitivamente, maggiore è il valore del coverage, più le comunità sono isolate tra loro. La funzione è coverage() di Networkx.

Modularity: misura più usata per valutare la qualità di una partizione di rete in comunità. Data una divisione dei vertici in un certo numero di gruppi, la modularità di questa divisione è definita come la frazione degli archi che vanno verso questo gruppo meno quello che ci si aspetterebbe se gli archi fossero distribuiti casualmente.

Il suo valore è positivo se il numero degli archi presenti in una community è maggiore del numero di quelli attesi e negativo in caso contrario.

Reti con modularità alta hanno connessione dense tra i nodi interni ai cluster e connessione lasche tra i nodi appartenenti a comunità diverse. La funzione è modularity() di Networkx.

$$Q = \sum_i^C (e_{ii} - a_i^2)$$

c : comunità
e_{ii}: # di archi intra-comunità per la comunità c
a_i: # di archi con almeno un vertice interno alla comunità
Il tutto in rapporto al numero totale di archi m

La modularità ha un range che va da -1 a 1. È positiva se il numero di archi all'interno del gruppo sono più del numero previsto. Ogni variazione dal valore 0 indica una differenza dal caso randomico.

Label Propagation Method

Algoritmo di clustering molto veloce per trovare comunità in un grafo. Forma delle comunità basandosi su un processo di propagazione di label (etichette) nel grafo. L'intuizione alla base dell'algoritmo è che una singola etichetta può diventare rapidamente dominante in un gruppo di nodi densamente connessi, ma avrà difficoltà ad attraversare una regione scarsamente connessa.

L'algoritmo procede in questo modo:

- Ogni nodo è inizializzato con un'etichetta di comunità univoca
- Ad ogni iterazione le etichette si propagano, e ogni nodo aggiorna la sua etichetta a quella a cui appartiene la maggior parte dei suoi vicini;
- Si raggiunge la convergenza quando ogni nodo ha l'etichetta predominante nel vicinato.

Man mano che le etichette si propagano, gruppi di nodi densamente connessi raggiungono rapidamente un consenso su un'etichetta univoca. Alla fine della propagazione rimarranno solo poche etichette, la maggior parte sarà scomparsa. Quindi i nodi che hanno la stessa etichetta appartengono alla stessa comunità. Per applicarlo è stato utilizzato il metodo `label_propagation_communities()` di Networkx.



Modularity: 0.326
Coverage: 0.659

La modularità indica che il numero di archi all'interno delle community è maggiore a delle community randomiche.

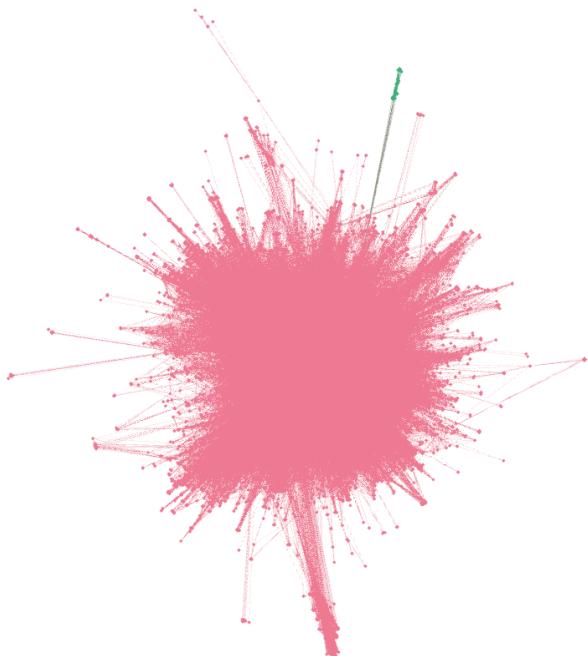
Mentre la coverage indica che la dimensione media della community è piuttosto grande, con una buona densità interna rispetto alla densità esterna. (Di fatto sono state clusterizzate zone isolate del grafo)

Clauset-Newman-Moore (Greedy)

Simile all'algoritmo di Louvain, è un algoritmo di clustering molto rapido che si basa sul concetto di massimizzazione della modularità. Algoritmo greedy euristico che mira all'identificazione rapida di comunità, adatto per reti di grandi dimensioni.

Ogni nodo della rete viene inizializzato con una differente comunità. Successivamente le combina ripetutamente, in modo da ottenere l'unione che produce il maggior incremento della modularità della struttura comunitaria. Tale valore può essere interpretato come una misura di affinità tra due comunità generiche e quindi mira a trovare due comunità simili in modo che possano essere unite.

La funzione utilizzata è `greedy_modularity_communities()` di `network` che restituisce una lista di set di nodi, rappresentanti ognuno una comunità.



Coverage: 0.9997960230319136
Modularity: 0.0013320868137045069

I risultati ottenuti sono particolari. Una altissima coverage indica che le comunità trovate sono estremamente isolate tra loro. Questo è quasi scontato dal momento in cui è stata creata una community che comprende quasi tutto il grafo.

Una modularità molto bassa, ma comunque positiva. Il numero di archi interni al cluster è molto simile a quelli attesi da una distribuzione di archi randomica.

Supposizione:

$$Q_{corr} = \frac{7}{24} - \left(\frac{10}{24}\right)^2 + \frac{3}{24} - \left(\frac{5}{24}\right)^2 + \frac{5}{24} - \left(\frac{7}{24}\right)^2 + \frac{5}{24} - \left(\frac{6}{24}\right)^2 = 0.4687$$

$$Q_{clue2} = \frac{12}{24} - \left(\frac{13}{24}\right)^2 + \frac{5}{24} - \left(\frac{7}{24}\right)^2 + \frac{5}{24} - \left(\frac{6}{24}\right)^2 = 0.4757 \quad \longleftarrow$$

$$Q_{clue3} = \frac{13}{24} - \left(\frac{16}{24}\right)^2 + \frac{3}{24} - \left(\frac{5}{24}\right)^2 + \frac{5}{24} - \left(\frac{6}{24}\right)^2 = 0.3246$$

Minimum Cut

Algoritmo molto semplice, che si basa sul concetto di cut, ovvero il numero di archi che attraversano diverse comunità. Lo scopo è quello di minimizzarlo.

1. Viene assegnata una soluzione iniziale randomica (una soluzione è accettabile solo se il numero di nodi per ogni cluster è lo stesso)
2. Si valuta uno scambio tra due vertici (u,v) appartenenti a due community diverse solo se questo porta ad un improvement.

L'improvement viene calcolato in questo modo: Inizialmente per ogni nodo v si calcolano:

C_v Numero di cut di quel nodo

U_v Numero di uncut di quel nodo

I_v Potenziale improvement del nodo stesso

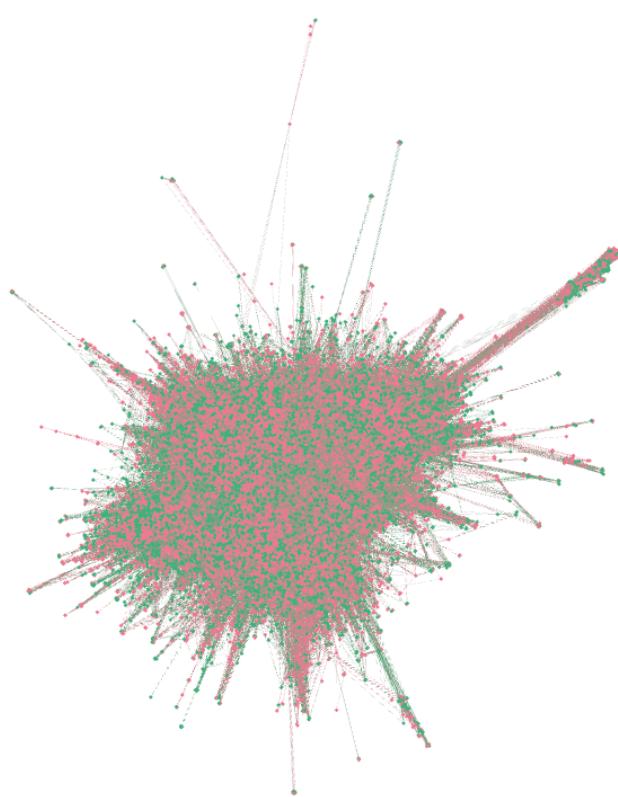
Poi per ogni coppia di nodi (u,v) si calcola la improvement totale dato dal loro scambio, che vale $I_v + I_u - 2$ se i due nodi sono connessi da un arco, $I_v + I_u$ altrimenti.

In networkx è implementata la funzione `kernighan_lin_bisection()`, con $k=2$.

Coverage: 0.500

Modularity: 0.042

La coverage è intuitivamente 0.5 ed indica una equa divisione delle due comunità. Mentre dalla modularità notiamo che non ci spostiamo molto dal caso randomico.



Tecnologie utilizzate

- Neo4J: per la creazione della rete pesata di interazioni tra eroi.
- Gephi: per la visualizzazione ed esplorazione del grafo, delle relative misure di centralità ed esplorazione dei cluster formati.
- Jupyter Notebook: per il testing del codice. Sui notebook utilizzati sono stati sviluppati e testati algoritmi su un grafo più leggero. Successivamente (a causa delle continue morti del kernel) il codice è stato convertito in python script per poterlo eseguire in un ambiente più stabile.

Le librerie utilizzate sono:

- Networkx - Per lo studio della rete, rappresentarla e applicazione di algoritmi di clustering nati per la community detection. Networkx è un pacchetto Python per la creazione, la manipolazione e lo studio della struttura, delle dinamiche di reti complesse. In particolare è stata utilizzata la versione 2.3.
- ScikitLearn - Per applicare algoritmi di clustering classici e valutarne la bontà dei risultati con metriche interne. è stata utilizzata la versione 0.24.1.

Divisione in moduli:

- Dataset
- MARVEL-Definition and Centrality.ipynb
- MARVEL-community_detection.ipynb
- Scripts: Directory contenente gli script, uno per ogni tecnica
- Graphs: Directory contenente i grafici generati dagli scripts
- Snapshots: Directory contenente le immagini

Conclusioni

Abbiamo dimostrato che la rete in analisi è una rete Small World, e che di fatto, nonostante sia plasmata su personaggi non reali, possiamo dire che questi abbiano un comportamento paragonabile a reti reali. Dalle misure di centralità sono stati evidenziati diversi nodi molto importanti, che non hanno smentito le aspettative (trattandosi dei supereroi più famosi in assoluto). I cluster hanno dimensioni variabili (dal 10% dei nodi totali all'1%), evidenziando, come atteso cluster piccoli tipici delle reti reali. I macro clusters sono dovuti ad una scelta del numero di clusters sottodimensionata.

Un paradigma dell'Universo Marvel potrebbe essere **Quicksilver**, che come risulta dalle analisi di clustering, appartiene a diversi cluster di eroi. Di fatto, è apparso prima come membro della Confraternita dei Mutanti Evil di Magneto nei primi numeri di **Uncanny X-Men**, poi è diventato un membro dei Vendicatori e successivamente di X-Factor, per finire come il capo dei Cavalieri di Wundagore; è anche il figlio di Magneto, il fratello gemello di Scarlet Witch, e ha sposato Crystal, ex fidanzata di Human Torch dei Fantastici Quattro e membro degli **Inumani**.