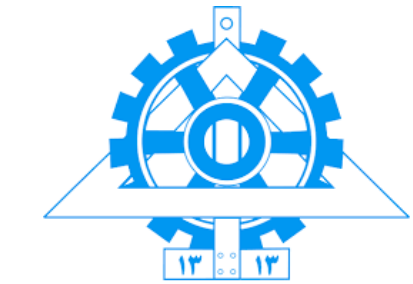
بسمه تعالی



دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر

درس شبکه‌های اجتماعی

پروژه‌ی پایانی

بهمن ۱۳۹۹

زهرا محقق راد

۸۹۰۱۹۹۲۶۰

فهرست

قسمت اول – مطالعه‌ی ابزارهای موجود:

۱) Gephi : نرم افزار Gephi یک نرم افزار Open Source و رایگان است، که برای تجزیه و تحلیل و بصری سازی(visualization) شبکه ها به کار می‌رود و روی Linux ، Mac OS X و Windows اجرا می‌شود. Gephi برای بصری سازی شبکه دارای layout های مختلف زیادی می‌باشد از جمله "openOrd" ،"Force atlas " ، "Yifan Hu" و ... . علاوه بر امکانات موجود در این نرم افزار، افزونه های (Plugins) زیادی هم برای آن موجود است.

این ابزار احتمالاً محبوب ترین پکیج برای visualization شبکه است. Gephi به دانش برنامه نویسی نیاز ندارد. نقطه قوت این ابزار، آن است که قادر به تولید visualizations با کیفیت بسیار بالا است. همچنین می‌تواند نمودارهای نسبتاً بزرگی را اداره کند (اندازه واقعی به سخت افزار شما (مخصوصاً RAM) بستگی دارد). توانایی محاسبه چند معیار متداول مانند Average Degree ، Centrality ، Diameter و غیره را دارد. اما Gephi یک ابزار قوی تر برای تجسم است تا تجزیه و تحلیل.

فرمت فایل‌های ورودی و خروجی:

ابزار Gephi عمده فرمت‌های معمول و معروف گراف را پشتیبانی می‌کند از جمله : CSV ، PAJEK NET ، Guess GDF، GEXF، GraphML، Graphviz DOT، UCInet DL، NetdrawVNA، Tulip TLP، Excel Spreadsheetater و ..... البته فرمت خاص این نرم افزار پسوند gephi دارد.

این نرم افزار برخی از فرمت‌های معمول گراف را به عنوان خروجی تولید می‌کند، از جمله : CSV، PAJEK NET ، Guess GDF، GEXF، GraphML، Excel Spreadsheetater، SVG، PDF، PNG و ...

۲) Cytoscape : نرم افزار Cytoscape یک ابزار Open Source و رایگان برای ترسیم و آنالیز شبکه ها است و از انواع شبکه‌ها (وزن دار، جهت‌دار، bipartite، multiedged و ...) پشتیبانی می‌کند. علاوه بر امکانات موجود در این نرم افزار، افزونه های (Plugins) زیادی هم برای آن موجود است. در ابتدا بیشترین کاربرد این نرم‌افزار در حوزه بیوانفورماتیک بود، اما امروز در تحلیل شبکه های اجتماعی نیز استفاده می‌شود. از شبکه های معروف در حوزه بیوانفورماتیک می توان به شبکه های تعامل پروتئین – پروتئین(PPI) ، شبکه بیان ژن، شبکه متابولیکی و شبکه فیلوژنی اشاره کرد. این ابزار نیز layout های مختلفی برای ترسیم گراف دارد از جمله hierarchical، Circular، orthogonal و tree. این نرم افزار برای ترسیم و به دست آوردن معیارها گراف نیاز به مقدار زیادی حافظه دارد. مانند نرم افزار Gephi، این نرم افزار نیاز به دانش برنامه نویسی ندارد.

فرمت فایل‌های ورودی و خروجی:

نرم افزار Cytoscape از فرمت‌های مختلفی به عنوان ورودی پشتیبانی می‌کند؛ از جمله : SIF، NNF، GraphML، XGMML، SBML، BioPAX ، GML، Delimited text، Excel Workbook ، Cytoscape.js JSON ، Cytoscape CX.

همچنین این نرم افزار برخی از فرمت‌ها را به عنوان خروجی تولید می‌کند از جمله :SIF ، NNF، XGMML، GraphML، PSI-MI Level 1 and 2.5، Cytoscape.js JSON، CX JSON و ...

۳) NetworkX : کتابخانه‌ی NetworkX یکی از کتابخانه‌های python برای تحلیل و آنالیز شبکه می‌باشد. برخلاف دو ابزار بالا برای استفاده از این ابزار نیاز به دانش برنامه‌نویسی می‌باشد. این کتابخانه توابع زیادی برای محاسبه‌ی انواع معیارهای مختلف از جمله centrality، community detection، Diameter، clustering coefficient و ... دارد، همچنین لی‌اوت‌های مختلفی برا ترسیم گراف دارا است : spring، bipartite، circular، planner، shell، spiral و ...

فرمت فایل‌هایی که این کتابخانه می‌تواند داده‌ها را خوانده یا در آن بنویسد عبارت است از : adjlist، edgelist، GEXF، GML، Pickle، GraphML، JSON، LEDA، YAML، Pajek، GIS Shapefile، SparseGraph6.

مقایسه:

به منظور مقایسه‌ی سه ابزار فوق، 5 مجموعه داده مختلف را روی هرکدام تجزیه و تحلیل کرده و زمان اجرای الگوریتم‌های مختلف و نیز visualization و کیفیت خروجی هرکدام را مورد بررسی قرار داده‌ایم. نتایج به شرح زیر می‌باشد. (سخت افزار مورد استفاده جهت مقایسه :RAM = 6 GB, Core i5)

مجموعه داده‌ی اول:

اولین شبکه‌ی مورد بررسی با استفاده از داده‌های ایمیل، از یک موسسه تحقیقاتی بزرگ اروپایی تولید شده است. ما اطلاعات ناشناس در مورد همه‌ی ایمیل‌های ورودی و خروجی بین اعضای موسسه تحقیقاتی داریم. اگر شخصی حداقل یک ایمیل برای شخصی دیگر ارسال کنید ، یک یال در شبکه به وجود می‌آید. ایمیل‌ها فقط ارتباط بین اعضای سازمان را نشان می‌دهند و مجموعه داده حاوی پیام های ورودی یا پیام های خروجی به بقیه افراد خارج از سازمان نیست. این مجموعه داده یک شبکه‌ی جهت دار بوده و شامل "1005" نود و "25571" یال می‌باشد.

ابتدا این مجموعه داده را در نرم‌افزار Cytoscape مورد بررسی قرار دادیم. این نرم افزار توانست در مدت زمان 00:02:35 معیارهای مختلف را برای هر نود محاسبه کند که این معیارها عبارتند از :

- Average shortest path length
- clustering coefficient
- closeness Centrality
- betweenness Centrality
- neighborhood connectivity
- out Degree , in Degree , Degree
- Eccentricity
- Stress

و نیز معیارهای Diameter، radius، clustering coefficient، connected component، Density، number of self-loops، avg number of neighbors را برای کل شبکه محاسبه نماید. همچنین این ابزار قادر به رسم نمودار برای توزیع درجه در شبکه و دیگر معیارهای موجود که در بالا نام برده شد، می‌باشد.

این معیارها برای مجموعه داده‌ی مورد نظر توسط نرم افزار Gephi نیز محاسبه شد، برخلاف نرم افزار Cytoscape که تمام این معیارها را یک جا حساب کرده و نشان می‌دهد، در Gephi می‌توان هرکدام از معیارها را می‌خواهیم به صورت جداگانه حساب کنیم. البته برخی از معیارهای موجود در Cytoscape در Gephi وجود ندارد؛ مانند: stress، Average shortest path، neighborhood connectivity و avg number of neighbors . تعدادی از معیارها نیز درابزار Gephi موجود می‌باشد که در Cytoscape وجود ندارد، از جمله : HIT centrality، Page Rank، Modularity و ...

تایم زمانی برای محاسبه‌ی معیارهای مشترک بین دو ابزار در مجموعه داده مورد استفاده به شرح زیر می‌باشد:

Algorithm	Time	output
• clustering coefficient	2s	0.473
• closeness Centrality • Eccentricity • Diameter • betweenness Centrality	5s	diameter = 7
• Avg Degree • out Degree • in Degree • Degree	0.6s	--
• Density	0.8s	0.025

• Connected component	1s	203

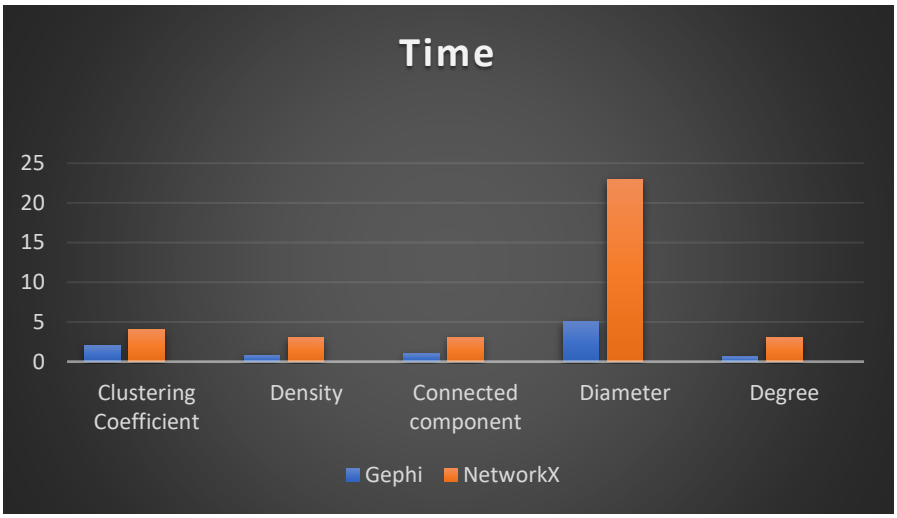
محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 1005 نود و25571 یال، به وسیله‌ی نرم‌افزار Gephi

علت آن‌که زمان اجرای closeness ، betweenness ، diameter وeccentricity درجدول مربوط به نرم‌افزار Gephi باهم آمده‌اند، این است که این ابزار به هنگام محاسبه‌ی diameter، همزمان مابقی معیارهای را برای هر نود نیز حساب می‌کند. همین اتفاق در محاسبه‌ی degree ، indegree ، outdegree نیز رخ می‌دهد.

کتابخانه‌ی NetworkX براخلاف دو نرم افزار قبل، مجموعه‌ی کاملی از انواع توابع برای محاسبه‌ی معیارهای مختلف در گراف را دارا می‌باشد. با استفاده از کتابخانه‌ی NetworkX نیز معیارهای مشترک در هر سه ابزار محاسبه شده و تایم آن‌ها به شرح زیر می‌باشد:

Algorithm	Time	output
• clustering coefficient	4s	0.365
• closeness Centrality	3s	-
• betweenness Centrality	13 s	-
• Eccentricity	1:45 h	-
• Diameter	1:50 h	7
• Degree	3s	-
• In Degree	3s	-
• Out Degree	3s	-
• Density	3s	0.025
• Connected component	3s	203

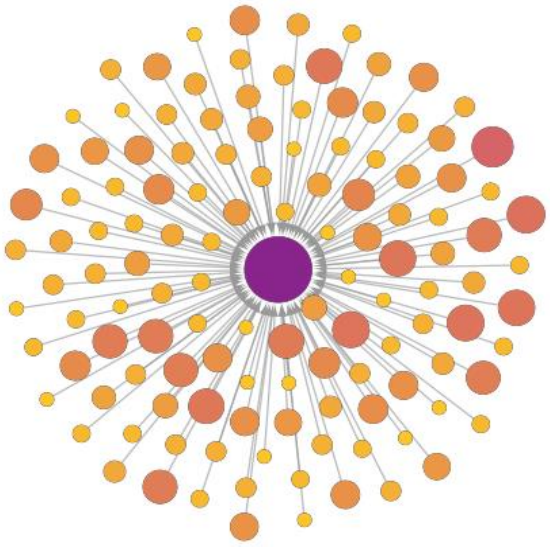
محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 1005 نود و25571 یال، به وسیله‌ی کتابخانه NetworkX



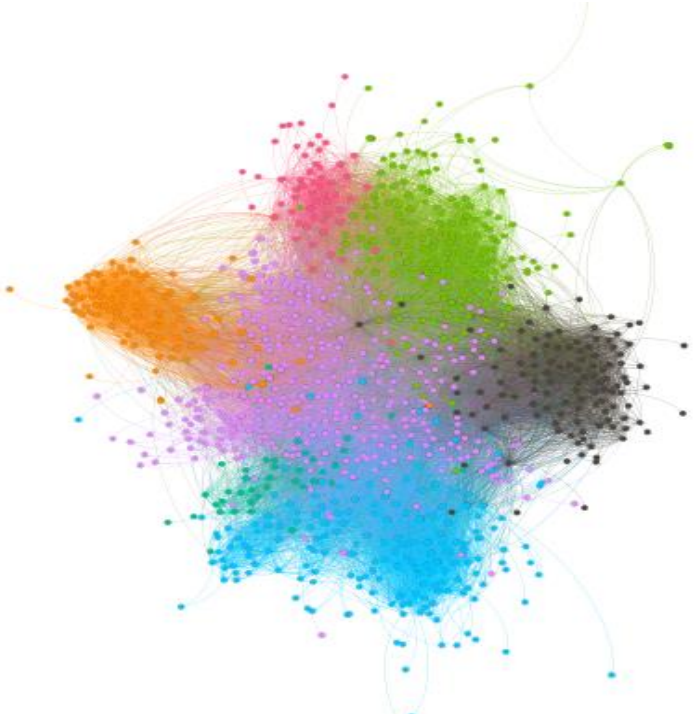
مقایسه‌ی زمان اجرای الگوریتم‌ها در نرم‌افزار Gephi و کتابخانه‌ی NetworkX
نمودار عمودی، زمان برحسب ثانیه می‌باشد.

همانطور که در دو جدول بالا قابل مشاهده‌است، مقداری که نرم‌افزار Gephi برای Avg Clustering Coefficient برمی‌گرداند با مقداری که NetworkX برمی‌گرداند متفاوت است.

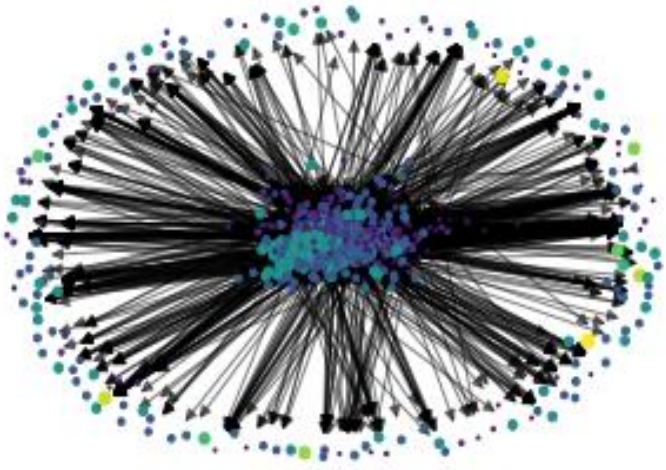
برای visualization گراف، هرسه ابزار فیلترهایی برای کاهش سایز گراف و نیز رنگبندی نودها براساس معیارهای مختلف و نیز پارتیشن بندی نودها، دارا هستند. البته لازم به ذکر است که کتابخانه‌ی NetworkX برای نمایش نمودارها و گراف از کتابخانه matplotlib کمک می‌گیرد. سرعت رسم گراف در Gephi و Cytoscape بالاتر از NetworkX بوده و نیز از لحاظ کیفیت خروجی Gephi از دو ابزار دیگر، مصور سازی بهتری انجام می‌دهد. visualization شبکه‌ی مورد استفاده در این بخش، در سه ابزار نام برده به صورت زیر می‌باشد:



مصور سازی با استفاده از نرم‌افزار Cytoscape.
با نصب پلاگین community detection بر روی این ابزار و با اجرای الگوریتم Louvain موجود در این پلاگین، انجمن‌های شبکه به دست آمده است. هر نودی که در تصویر مشاهده می‌فرمایید، یک انجمن را نشان می‌دهدو رنگ آمیزی و سایز هر نود بر اساس تعداد نودهای موجود در هر انجمن می‌باشد. به عنوان مثال نود بنفش رنگی که در تصویر مشاهده می‌فرمایید بزرگترین انجمن می‌باشد. هرچه رنگ به سمت نارنجی کم‌رنگ می‌رود و از سایز نودها کاسته می‌شود، اندازه‌ی انجمن‌ها نیز کاهش می‌یابد. لی‌اوت مورد استفاده نیز Prefuse Force Directed OpenCL می‌باشد.



مصور سازی با استفاده از نرم‌افزارGephi.
نودها براساس مقدار Modularity هر نود رنگ‌بندی شده‌اند.
لی‌اوت مورد استفاده نیز MultiGravity ForceAtlas 2 می‌باشد. این لی‌اوت در خود نرم‌افزار موجود نبوده و پلاگین آن به طور جداگانه نصب شده‌است.



مصورسازی با استفاده از کتابخانه‌ی NetworkX.
با استفاده از این کتابخانه نیز، ابتدا با استفاده از الگوریتم Louvain انجمن‌های شبکه را محاسبه کرده، سپس گراف را براساس مقادیر انتساب داده شده به هرنود توسط این الگوریتم، رسم کرده‌ایم. نودهای هم‌رنگ در یک پارتیشن قرار می‌گیرند و نیز هرچه سایز نود بزرگتر است عدد به دست آمده برای آنجمنی که در آن قرار گرفته است بزرگتر می‌باشد.

کد زیر نحوه‌ی نمایش گراف را نشان می‌دهد:

```
G = nx.read_edgelist("drive/MyDrive/CA-
CondMat.txt",nodetype=int, edgetype=int,create_using=nx.DiGraph)


partition = community.best_partition(nx.to_undirected(G))
pos = nx.spring_layout(G)
cmap = cm.get_cmap('viridis', max(partition.values()) + 1)
plt.axis('off')
nx.draw_networkx_nodes(G, pos, node_size=list(partition.values()),
                       cmap=cmap, node_color=list(partition.values()))
```

مجموعه داده‌ی دوم:

مقایسه‌ی بعدی با استفاده از مجموعه داده‌ی استخراج شده از رای‌گیری‌های ویکی‌پدیا برای ارتقاء اعضا عادی به سمت ادمین می‌باشد. برای اینکه یک کاربر بتواند ادمین شود ، درخواست RfA صادر می شود و انجمن ویکی پدیا از طریق یک بحث عمومی یا رای گیری تصمیم می گیرد چه کسی را به سمت ادمین ارتقا دهد. با استفاده از جدیدترین تاریخچه ویرایش صفحه ویکی پدیا (از ۳ ژانویه ۲۰۰۸) ما تمام اطلاعات مربوط به انتخابات سرپرست و تاریخچه رأی را استخراج کردیم که به ما 2794 انتخابات با 103663 رأی و 7066کاربر شرکت کننده در انتخابات به ما داد. پس مجموعه داده‌ی ما متشکل از "7066" نود و "103663" یال می‌باشد. تمامی مراحل انجام شده در مجموعه داده قبلی نیز در اینجا تکرار شده است و نتایج به دست آمده در هر کدام از ابزارهای مورد استفاده به شرح زیر می‌باشد:

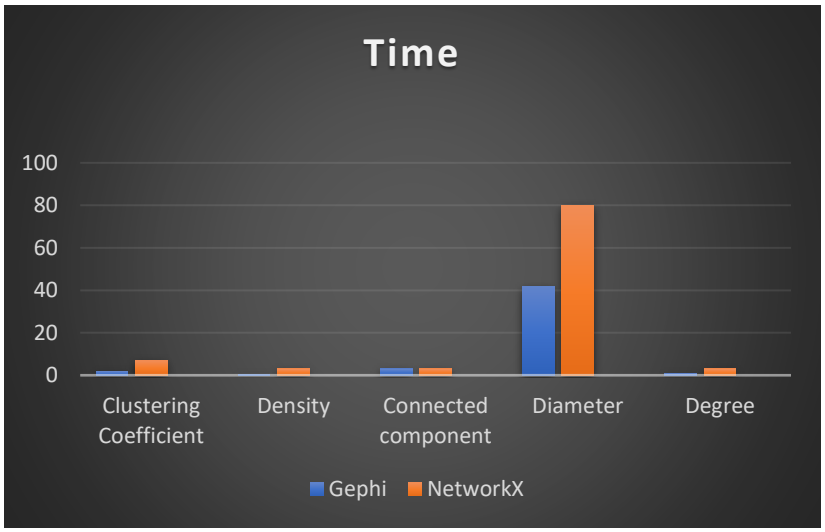
نرم‌افزار Cytoscape به هنگام آنالیز گراف و به دست آوردن معیارهای مختلف در این مجموعه داده، بیشتری تایم زمانی را نسبت به دو ابزار دیگر داشت. زمان اجرای آن برای شبکه با "7066" نود با "103663" یال برابر 01:23:00 می‌باشد و نیز مصرف حافظه‌ی آن نسبت به دو الگوریتم دیگر بسیار بیشتر بود.

Algorithm	Gephi	output
• clustering coefficient	2s	0.209
• closeness Centrality • Eccentricity • Diameter • betweenness Centrality	43s	diameter = 10
• Avg Degree • out Degree • in Degree • Degree	1s	--
• Density	0.4s	0.002
• Connected component	3s	5816

محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 7066 نود و103663 یال، به وسیله‌ی نرم‌افزار Gephi

Algorithm	NetworkX	output
• clustering coefficient	7s	0.140
• closeness Centrality	35s	-
• Eccentricity	1:16 min	-
• Diameter	1:20 min	10
• betweenness Centrality	3:40 min	-
• Degree	3s	-
• in Degree	3s	-
• out Degree	3s	-
• Density	3s	0.002
• Connected component	3s	5816

محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 7066 نود و103663 یال، به وسیله‌ی نرم‌افزار NetworkX

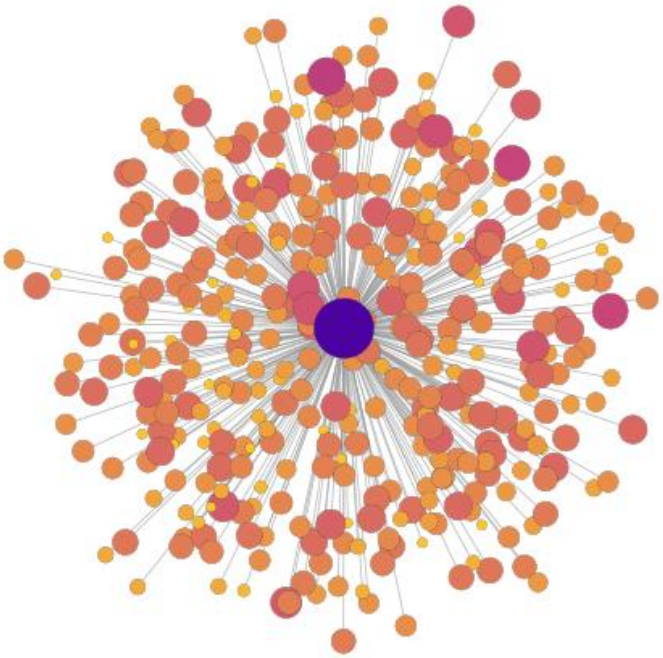


مقایسه‌ی زمان اجرای الگوریتم‌ها در نرم‌افزار Gephi و کتابخانه‌ی NetworkX

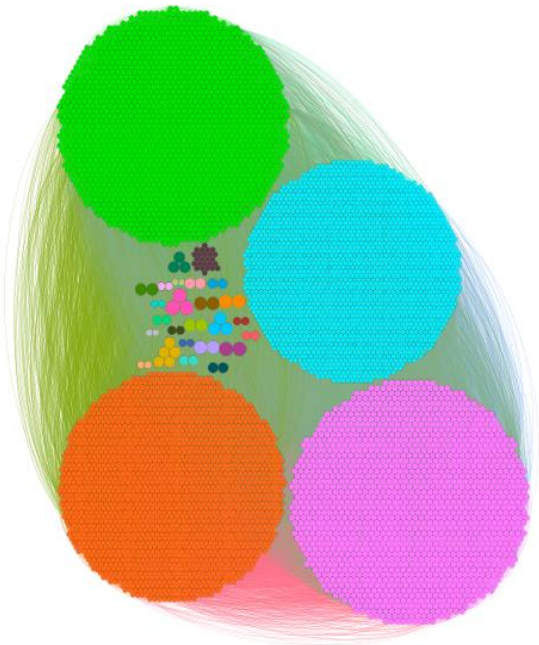
نمودار عمودی، زمان برحسب ثانیه می‌باشد.

همانطور که در دو جدول بالا قابل مشاهده‌است، مانند آنچه در مجموعه‌داده‌ی قبل رخ داد، مقداری که نرم‌افزار Gephi برای Avg Clustering Coefficient برمی‌گرداند با مقداری که NetworkX برمی‌گرداند متفاوت است.

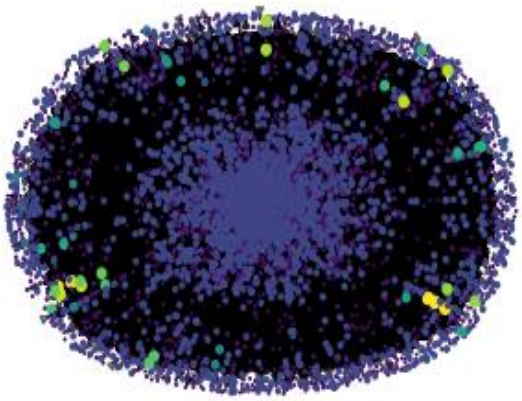
ترسیم گراف در سه ابزار به صورت زیر می‌باشد:



مصور سازی با استفاده از نرم‌افزار Cytoscape.
با نصب پلاگین community detection بر روی این ابزار و با اجرای الگوریتم Louvain موجود در این پلاگین، انجمن‌های شبکه به دست آمده است. هر نودی که در تصویر مشاهده می‌فرمایید، یک انجمن را نشان می‌دهدو رنگ آمیزی و سایز هر نود بر اساس تعداد نودهای موجود در هر انجمن می‌باشد. به عنوان مثال نود بنفش رنگی که در تصویر مشاهده می‌فرمایید بزرگترین انجمن می‌باشد. هرچه رنگ به سمت نارنجی کم‌رنگ می‌رود و از سایز نودها کاسته می‌شود، اندازه‌ی انجمن‌ها نیز کاهش می‌یابد. لی‌اوت مورد استفاده نیز Prefuse Force Directed OpenCL می‌باشد.



مصور سازی با استفاده از نرم‌افزارGephi.
نودها براساس مقدار Modularity هر نود رنگ‌بندی شده‌اند. هر رنگ نشان دهنده‌ی یک انجمن می‌باشد. لی‌اوت مورد استفاده نیز Circle Pack Layout می‌باشد. این لی‌اوت در خود نرم‌افزار موجود نبوده و پلاگین آن به طور جداگانه نصب شده‌است.



مصورسازی با استفاده از کتابخانه‌ی NetworkX.

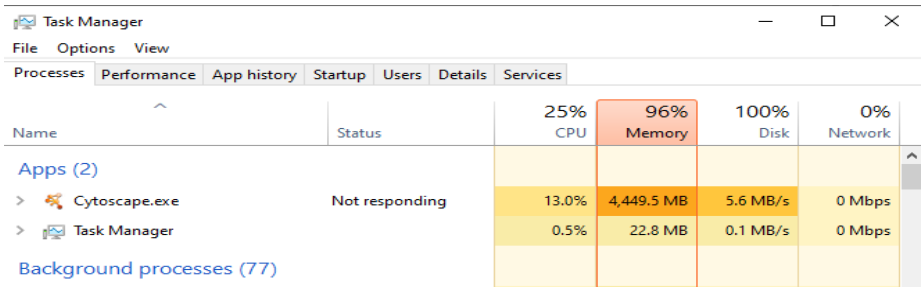
با استفاده از این کتابخانه نیز، ابتدا با استفاده از الگوریتم Louvain انجمن‌های شبکه را محاسبه کرده، سپس گراف را براساس مقادیر انتساب داده شده به هرنود توسط این الگوریتم، رسم کرده‌ایم. نودهای هم‌رنگ در یک پارتیشن قرار می‌گیرند و نیز هرچه سایز نود بزرگتر است عدد به دست آمده برای آنجمنی که در آن قرار گرفته است بزرگتر می‌باشد. کدی که برای رسم این گراف زده‌شده‌است، مانند شبکه قبلی می‌باشد.

مجموعه‌داده سوم:

مجموعه‌ی داده‌ی بعد مورد استفاده برای این قسمت، شبکه‌ی همکاری Arxiv COND-MAT (Condense Matter Physics) استفاده شده است که همکاری علمی بین نویسندگان مقاله‌های ارسال شده در دسته‌ی Condense Matter را پوشش می‌دهد. اگر نویسنده i با نویسنده j مقاله مشترک داشته باشد، در شبکه یک یال بدون جهت از i به j رسم می‌شود. اگر مقاله توسط k نویسنده‌ی مشترک باشد، یک گراف کاملاً متصل K نودی تشکیل خواهد شد. مجموعه داده‌ی ما متشکل از "23133" نود و "93497" یال می‌باشد و نیز یک شبکه بدون جهت است.

تمامی مراحل انجام شده در مجموعه داده‌های قبلی نیز در اینجا تکرار شده است و نتایج به دست آمده در هر کدام از ابزارهای مورد استفاده به شرح زیر می‌باشد:

با توجه به اینکه تعداد نودها نسبت به دو مجموعه داده‌ی قبلی افزایش یافت، نرم‌افزار Cytoscape به دلیل استفاده بالای Memory، نتوانست این مجموعه داده را آنالیز و بعد از گذشتن 3 ساعت"hang" کرد. اما با استفاده از پلاگین community detection بر روی این نرم‌افزار و تشخیص انجمن‌های موجود در شبکه با استفاده از الگوریتم "Louvain" هر انجمن را به صورت یک سوپر نود در نظر گرفته و توانستیم آن را رسم کنیم.



تایم اجرای الگوریتم ها در دو ابزار دیگر به صورت زیر می‌باشد:

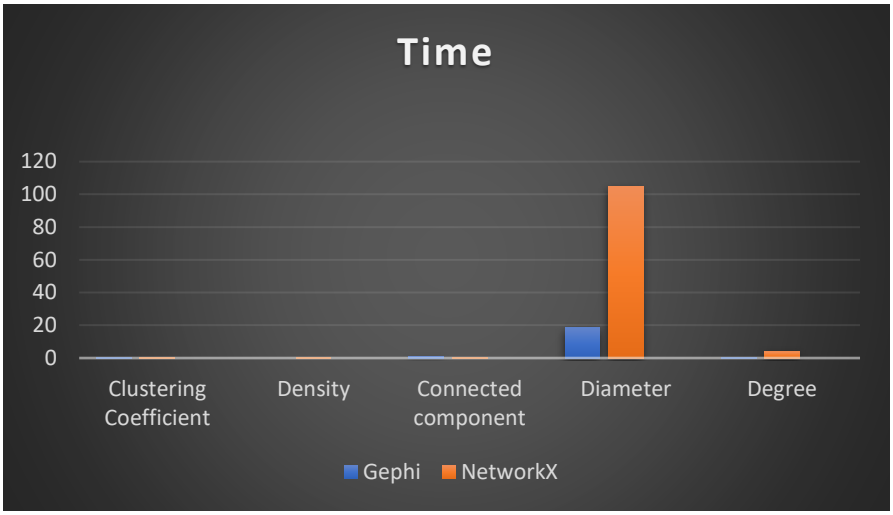
Algorithm	Gephi	output
• clustering coefficient	1s	0.706
• closeness Centrality • Eccentricity • Diameter • betweenness Centrality	30 min	diameter = 15
• Avg Degree • Degree	1s	-
• Density	0.2s	0
• Connected component	3s	576

محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 23133 نود و93497 یال، به وسیله‌ی نرم‌افزار Gephi

Algorithm	NetworkX	output
• clustering coefficient	4s	0.633
• closeness Centrality	25min	-
• betweenness Centrality	30 min	-
• Eccentricity	1:45 h	-
• Diameter	1:50 h	15
• Degree	4s	-
• Density	3s	0.0003
• Connected component	4s	576

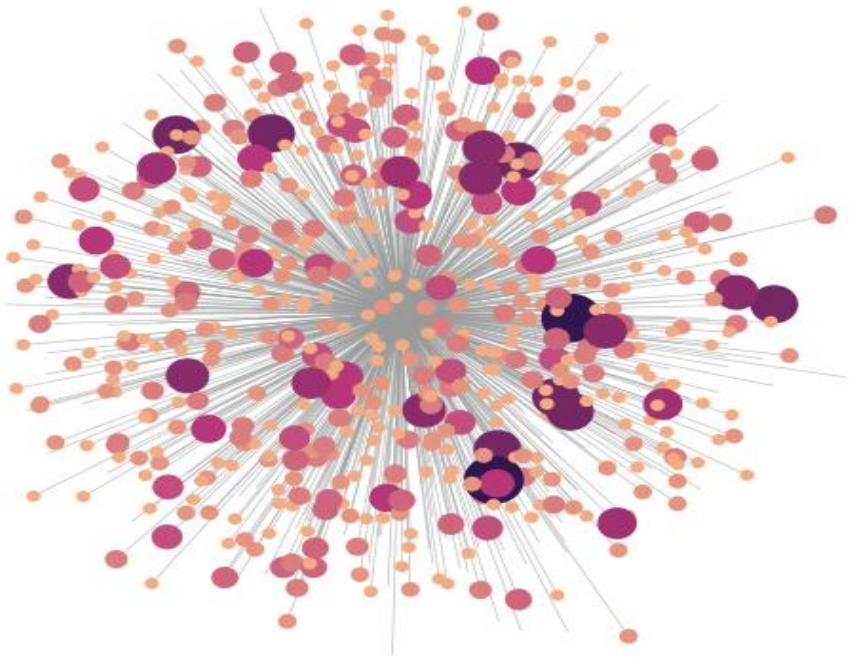
محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 23133 نود و93497 یال، به وسیله‌ی نرم‌افزار NetworkX

همانطور که در جدول بالا قابل مشاهده است، نرم فزار Gephi قادر به اندازه‌گیری density نیست و مقدار 0 را برمی‌گرداند و در اینجا نیز مانند دو مجموعه‌ی داده‌ی قبلی، مقداری که نرم‌افزار Gephi برای Avg Clustering Coefficient نمایش می‌دهد با خروجی که NetworkX می‌دهد، متفاوت است.



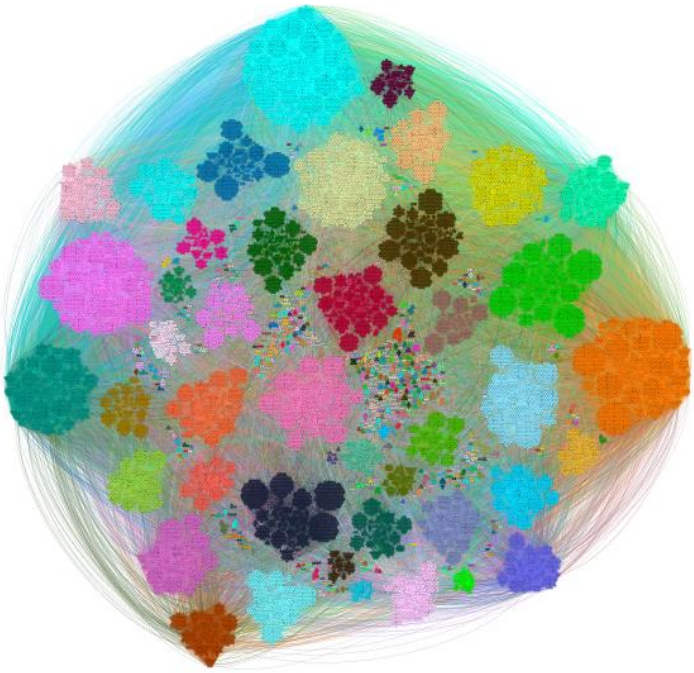
مقایسه‌ی زمان اجرای الگوریتم‌ها در نرم‌افزار Gephi و کتابخانه‌ی NetworkX
نمودار عمودی، زمان برحسب دقیقه می‌باشد.

ترسیم گراف در سه ابزار به صورت زیر می‌باشد:

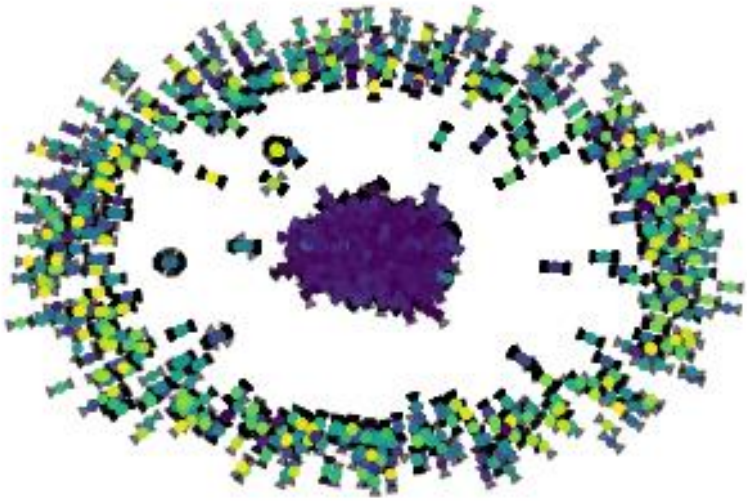


مصور سازی با استفاده از نرم‌افزار Cytoscape.

با نصب پلاگین community detection بر روی این ابزار و با اجرای الگوریتم Louvain موجود در این پلاگین، انجمن‌های شبکه به دست آمده است. هر نودی که در تصویر مشاهده می‌فرمایید، یک انجمن را نشان می‌دهدو رنگ آمیزی و سایز هر نود بر اساس تعداد نودهای موجود در هر انجمن می‌باشد. هرچه رنگ به سمت صورتی کم‌رنگ می‌رود و از سایز نودها کاسته می‌شود، اندازه‌ی انجمن‌ها نیز کاهش می‌یابد. لی‌اوت مورد استفاده نیز Prefuse Force Directed OpenCL می‌باشد.



مصور سازی با استفاده از نرم‌افزارGephi.
نودها براساس مقدار Modularity هر نود رنگ‌بندی شده‌اند. هر رنگ نشان دهنده‌ی یک انجمن می‌باشد. لی‌اوت مورد استفاده نیز Circle Pack Layout می‌باشد. این لی‌اوت در خود نرم‌افزار موجود نبوده و پلاگین آن به طور جداگانه نصب شده‌است.



مصورسازی با استفاده از کتابخانه‌ی NetworkX.
با استفاده از این کتابخانه نیز، ابتدا با استفاده از الگوریتم Louvain انجمن‌های شبکه را محاسبه کرده، سپس گراف را براساس مقادیر انتساب داده شده به هرنود توسط این الگوریتم، رسم کرده‌ایم. نودهای هم‌رنگ در یک پارتیشن قرار می‌گیرند و نیز هرچه سایز نود بزرگتر است عدد به دست آمده برای آنجمنی که در آن قرار گرفته است بزرگتر می‌باشد. کدی که برای رسم این گراف زده‌شده‌است، مانند شبکه قبلی می‌باشد.

مجموعه‌داده چهارم:

مجموعه داده‌ی مورد استفاده بعدی، مربوط به اطلاعات جمع آوری شده در مورد شبکه ارتباطی ایمیل Enron می‌باشد. شبکه ارتباطی ایمیل Enron تمام ارتباطات ایمیل را در یک مجموعه‌داده حدود نیم میلیون ایمیل پوشش می‌دهد. این داده‌ها توسط کمیسیون تنظیم مقررات انرژی فدرال در جریان تحقیقات عمومی ، منتشر و به وب ارسال شد. نودهای شبکه آدرس‌های ایمیل هستند و اگر یک آدرس i حداقل یک ایمیل به آدرس j ارسال کند ، در شبکه یک یال بدون جهت از i به j رسم می‌شود. تعداد نودهای این شبکه "36692" و تعداد یال‌های آن "183381" می‌باشد و نیز شبکه بدون جهت است.
تمامی مراحل انجام شده در مجموعه داده‌های قبلی نیز در اینجا تکرار شده است و نتایج به دست آمده در هر کدام از ابزارهای مورد استفاده به شرح زیر می‌باشد:

نرم‌افزار Cytoscape به دلیل استفاده بالای Memory، نتوانست این مجموعه داده را آنالیز و بعد از گذشتن 3 ساعت"hang" کرد. اما با استفاده از پلاگین community detection بر روی این نرم‌افزار و تشخیص انجمن-های موجود در شبکه با استفاده از الگوریتم "Louvain" هر انجمن را به صورت یک سوپر نود در نظر گرفته و توانستیم آن را رسم کنیم.

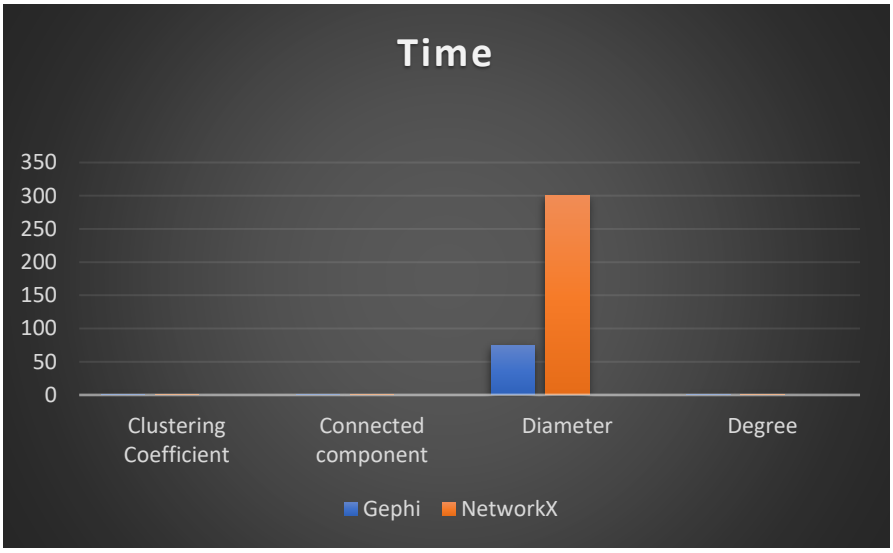
Algorithm	Gephi	output
• clustering coefficient	3s	0.716
• closeness Centrality • Eccentricity • Diameter • betweenness Centrality	1:25 h	diameter = 13
• Degree	1s	--
• Density	0	--
• Connected component	4s	1065

محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 36692 نود و183381 یال، به وسیله‌ی نرم‌افزار Gephi

Algorithm	NetworkX	output
• clustering coefficient	8s	0.496
• closeness Centrality	52min	-
• betweenness Centrality	1 h	-
• Eccentricity	$\geq$ 5 h	-
• Diameter	$\geq$ 5 h	13
• Degree	5s	-
• Density	5s	0.0002
• Connected component	5s	1065

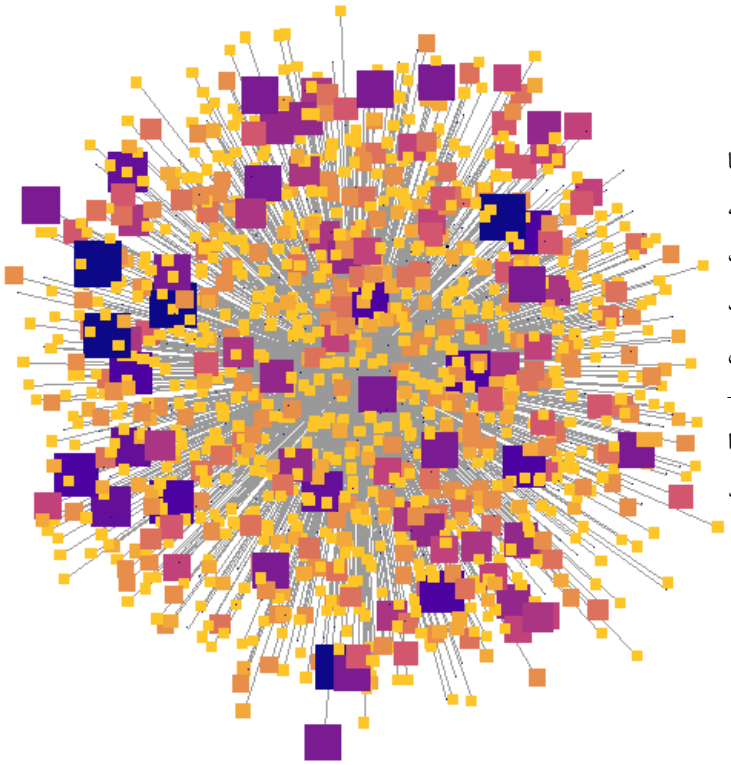
محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 36692 نود و183381 یال، به وسیله‌ی نرم‌افزار NetworkX

در اینجا نیز مانند دو مجموعه‌ی داده‌ی قبلی مقداری که نرم‌افزار Gephi برای Avg Clustering Coefficient نمایش می‌دهد با خروجی که NetworkX می‌دهد، متفاوت است و قادر به محاسبه‌ی density نیز نمی‌باشد. به همین دلیل از نمایش زمان اجرای Density خودداری کرده‌ایم.

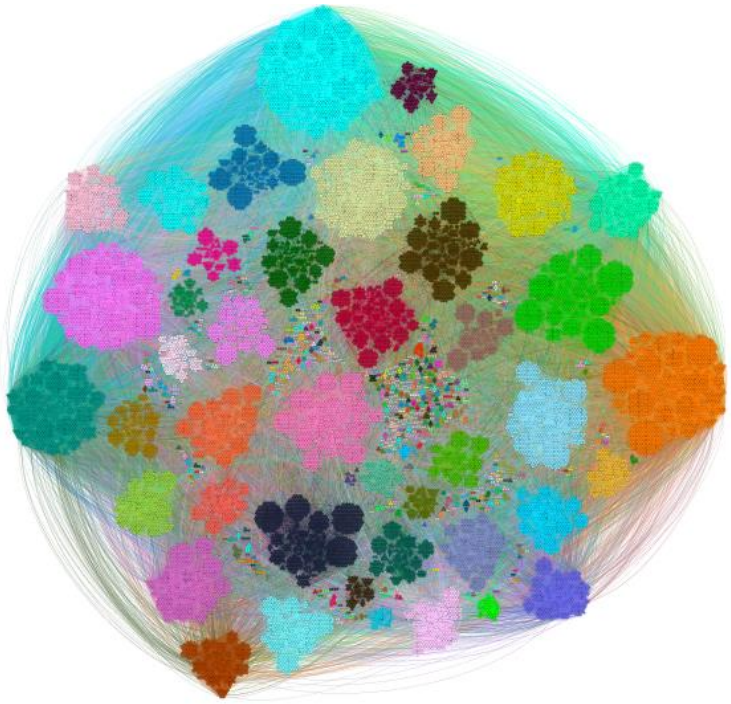


مقایسه‌ی زمان اجرای الگوریتم‌ها در نرم‌افزار Gephi و کتابخانه‌ی NetworkX
نمودار عمودی، زمان برحسب دقیقه می‌باشد.

ترسیم گراف در سه ابزار به صورت زیر می‌باشد :



مصور سازی با استفاده از نرم‌افزار Cytoscape.
با نصب پلاگین community detection بر روی این ابزار و با اجرای الگوریتم Louvain موجود در این پلاگین، انجمن‌های شبکه به دست آمده است. هر نودی که در تصویر مشاهده می‌فرمایید، یک انجمن را نشان می‌دهدو رنگ آمیزی و سایز هر نود بر اساس تعداد نودهای موجود در هر انجمن می‌باشد. به عنوان مثال نودهای بنفش رنگی که در تصویر مشاهده می‌فرمایید بزرگترین سایز انجمن می‌باشند. هرچه رنگ به سمت نارنجی کم‌رنگ می‌رود و از سایز نودها کاسته می‌شود، اندازه‌ی انجمن‌ها نیز کاهش می‌یابد. لی‌اوت مورد استفاده نیز Prefuse Force Directed OpenCL می‌باشد.



مصور سازی با استفاده از نرم‌افزارGephi.
نودها براساس مقدار Modularity هر نود رنگ‌بندی شده‌اند. هر رنگ نشان دهنده‌ی یک انجمن می‌باشد. لی‌اوت مورد استفاده نیز Circle Pack Layout می‌باشد. این لی‌اوت در خود نرم‌افزار موجود نبوده و پلاگین آن به طور جداگانه نصب شده-است.

به دلیل زمان بالای ترسیم گراف در ابزار NetworkX در شبکه‌های بزرگ، از کشیدن آن خودداری کرده‌ایم.

مجموعه‌داده پنجم:

و در نهایت آخرین مجموعه داده‌ی استفاده شده برای این قسمت، مربوط به وب‌سایت "Slashdot" می‌باشد. Slashdot یک وب‌سایت خبری مرتبط با فناوری است که برای جامعه‌ی کاربری خاص خود شناخته شده است. در سال 2002 ، Zoo Slashdot Slashdot را معرفی کرد، که به کاربران اجازه می‌دهد یکدیگر را به عنوان دوست یا دشمن برچسب‌گذاری کنند. این شبکه ارتباطات دوست / دشمن را بین کاربران Slashdot برقرار می‌کند. این شبکه در فوریه ۲۰۰۹ به دست آمد. این شبکه جهت‌دار بوده و تعداد نودهای آن برابر "82144" و تعداد یال‌های آن برابر "549202" می‌باشد.

تمامی مراحل انجام شده در مجموعه داده‌های قبلی نیز در اینجا تکرار شده است و نتایج به دست آمده در هر کدام از ابزارهای مورد استفاده به شرح زیر می‌باشد:
نرم افزار Cytoscape با توجه به memory مورد استفاده، قادر به آنالیز و ترسیم شبکه نبود، اما در دو نرم‌افزار دیگر زمان اجرای الگوریتم‌ها به صورت زیر می‌باشد:

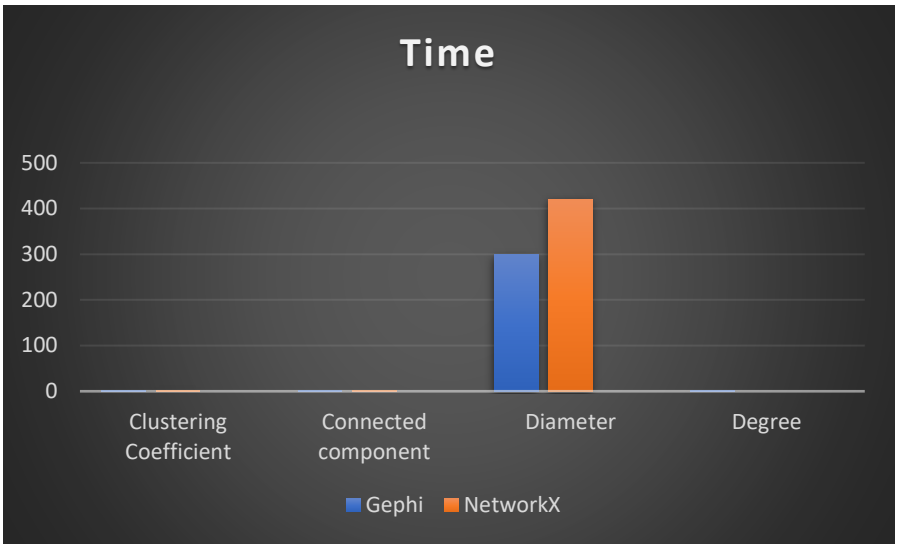
Algorithm	Gephi	
• clustering coefficient	5s	0.527
• closeness Centrality • Eccentricity • Diameter • betweenness Centrality	≥ 5 h	diameter = 13
• out Degree • in Degree • Degree	1s	--
• Density	--	
• Connected component	9s	10559

محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 82144 نود و 549202 یال، به وسیله‌ی نرم‌افزار Gephi

Algorithm	NetworkX	output
• clustering coefficient	1 min	0.06
• closeness Centrality	1:40 h	-
• betweenness Centrality	1:50 h	-
• Eccentricity	≥ 7 h	
• Diameter	≥ 7 h	13
• Degree	9s	-
• in Degree	8s	–
• out Degree	7s	–
• Density	7s	0.00014
• Connected component	9s	10559

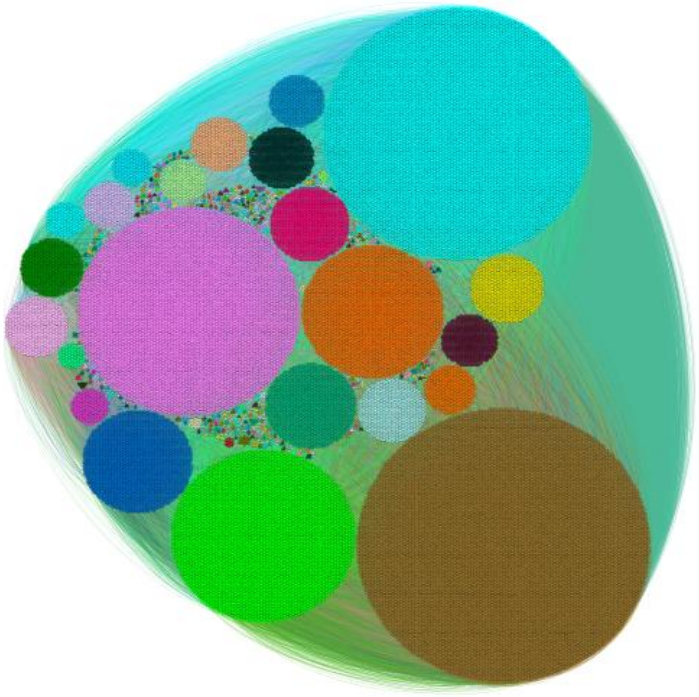
محاسبه‌ی زمان اجرای الگوریتم‌ها در شبکه‌ای با 82144 نود و549202 یال، به وسیله‌ی نرم‌افزار NetworkX

در اینجا نیز مانند دو مجموعه‌ی داده‌ی قبلی مقداری که نرم‌افزار Gephi برای Avg Clustering Coefficient نمایش می‌دهد با خروجی که NetworkX می‌دهد، متفاوت است و قادر به محاسبه‌ی density نیز نمی‌باشد.



مقایسه‌ی زمان اجرای الگوریتم‌ها در نرم‌افزار Gephi و کتابخانه‌ی NetworkX
نمودار عمودی، زمان برحسب دقیقه می‌باشد.

ترسیم گراف به وسیله‌ی ابزار Gephi: (به دلیل زمان بالای ترسیم گراف در ابزار NetworkX در شبکه‌های بزرگ، از کشیدن آن خودداری کرده‌ایم.)



مصور سازی با استفاده از نرم‌افزارGephi.
نودها براساس مقدار Modularity هر نود رنگ‌بندی شده‌اند. هر رنگ نشان دهنده‌ی یک انجمن می‌باشد. لی‌اوت مورد استفاده نیز Circle Pack Layout می‌باشد. این لی‌اوت در خود نرم‌افزار موجود نبوده و پلاگین آن به طور جداگانه نصب شده‌است.

نتیجه گیری:

همانطور که در پنج مجموعه‌داده‌ی بالا مشاهده شد، نرم افزار Cytoscape علی‌رغم مزیت‌هایی که در ترسیم شبکه و نیز به دست‌آوردن ویزگی‌های شبکه داشت، به دلیل استفاده از حافظه‌ی بالا و نیز باتوجه به حافظه‌ای که ما در اختیار داشتیم، نتوانستیم شبکه‌ای با بیشتر از 10k نود را در آن آنالیز کنیم و تنها توانستیم آن‌ها را به طریقی که در بالا ذکر شد توسط این نرم‌افزار رسم کنیم. اگر حافظه به اندازه‌ی کافی در اختیار داشته باشیم این ابزار، یکی از ابزارهای خوب برای ترسیم و آنالیز شبکه می‌باشد؛ به دلیل بهره‌مندی از layout های مختلف و نیز روش‌های مختلف برای فیلتر کردن شبکه از جمله degree، k_core، Giant component و ... همچنین می‌توان نودها را براساس معیارهای موجود در این ابزار(... ,degree , betweenness, clustering coefficient) سایز بندی و رنگ-آمیزی کرد. علاوه بر امکانات موجود در خود این نرم‌افزار، پلاگین‌های زیادی برای لی‌اوت و محاسبه‌ی معیارهای گراف نیز وجود دارد (از جمله پلاگین Community detection) که در شبکه‌های بالا مورد استفاده قرار گرفت. از دیگر مزیت‌های این ابزار می‌توان به این موضوع اشاره کرد که نیاز به دانستن زبان برنامه نویسی ندارد و به راحتی می‌توان یک شبکه را در این ابزار آنالیز و ترسیم کرد.

نرم افزار Gephi برخلاف Cytoscape حافظه‌ی کمتری مصرف می‌کند و به راحتی توانستیم با توجه به سخت افزاری که داشتیم تا 100K نود را آنالیز و ترسیم کنیم. مزیت Gephi نسبت به دو ابزار دیگر در ترسیم گراف است. Gephi شامل لی‌اوت‌های زیادی برای ترسیم گراف می‌باشد و نیز می‌توان نودها را برحسب معیارها و ویزگی‌هایی که دارند سایزبندی، رنگ‌آمیزی و نیز پارتیشن بندی کرد؛ همچنین می‌توان تعداد نودهای گراف را براساس ویزگی‌ها و معیارهای مختلف (degree، k_core، Giant component و ...) فیلتر کرد. با وجود آنکه نرم‌افزار Cytoscape نیز دارای لی‌اوت‌های زیادی می‌باشد، اما برخی از لی‌اوت‌های Gephi، ترسیم بهتری از شبکه می‌دهند. در Gephi نیز می‌توان پلاگین‌های مختلفی برای ترسیم و محاسبه‌ی معیارهای گراف(newman_giravn clustering, label propagation clustering, leiden algorithm, ...) نیز نصب کرد. همانند نرم‌افزار Cytoscape از مزیت‌های Gephi نیز می‌توان به این موضوع اشاره کرد که نیاز به دانستن زبان برنامه نویسی ندارد و به راحتی می‌توان یک شبکه را در این ابزار آنالیز و ترسیم کرد. از معایب این نرم‌افزار نیز همانطور که در پنج مجموعه‌داده‌ی بالا مشاهده می‌شود، مقداری که برای Avg Clustering Coefficient برمی‌گرداند، نادرست است (برای محاسبه‌ی این معیار علاوه بر NetworkX از کتابخانه‌ی Snap نیز استفاده شده و این دو کتابخانه، عددی یکسان برای مجموعه-داده‌ی مشابه برمی‌گردانند که با مقداری که Gephi نمایش می‌دهد، متفاوت می‌باشد.) و نیز این ابزار برای محاسبه‌ی density اگر چگالی گراف خیلی کم باشد و به صفر میل کند، نمی‌تواند مقدار درست را برگرداند و عدد 0 را نمایش می‌دهد.

مزیت کتابخانه‌ی NetworkX نسبت به دو ابزار دیگر در محاسبه‌ی معیارهای گراف می‌باشد. این کتابخانه، انواع مختلف و زیادی از توابع را برای محاسبه‌ی ویزگی‌های گراف دارا می‌باشد، برخلاف دو ابزار دیگر که تعداد محدودی از این معیارها را می‌توانستند محاسبه کنند. همانطور که در بالا مشاهده کردیم هرچه تعداد نودهای شبکه زیادتر شد،

زمان اجرای برخی از الگوریتم‌ها (به عنوان مثال "diameter") بسیار افزایش داشت (با توجه به سخت افزار مورد استفاده). راه حلی که برای این قسمت می‌توان برای این مشکل ارائه داد؛ استفاده از کتابخانه‌های دیگری ست که وقتی مجموعه داده‌ی بزرگی داشتیم می‌توان از آن‌ها استفاده‌کرد. به عنوان مثال یکی از این کتابخانه‌ها، کتابخانه‌ی "Snap" می‌باشد که توسط دانشگاه "Stanford" پیاده‌سازی شده‌است[1] و به عنوان مثال معیار زمان اجرای diameter که در کتابخانه NetworkX برای شبکه‌ای با "23313" نود و "93497" یال ، حدود "2" ساعت به طول انجامید، با استفاده از کتابخانه‌ی snap زمان اجرای آن به طرز چشم‌گیری کاهش یافته و به زمانی حدود "5" دقیقه می‌رسد و در شبکه‌ی با "36692" نود و "183381" یال، زمان اجرای این الگوریتم با کتابخانه‌ی snap برابر "7" دقیقه و برای شبکه‌ی بعدی با "82144"نود و "549202" یال زمان اجرای این الگوریتم حدود "1" ساعت می‌باشد. اما در شبکه‌های کوچکتر تا 10K نود تفاوت خیلی زیادی مانند آنچه در بالا اشاره کردیم بین این دو کتابخانه وجود ندارد. در همین راستا، اگر بخواهیم معیارهای مختلف را در شبکه‌های خیلی بزرگ محاسبه کنیم، برخی از این توابع را که تایم زیادی دارند در کتابخانه‌ی NetworkX ، می‌توان با استفاده از توابع موجود در کتابخانه‌های دیگر محاسبه کرد از جمله کتابخانه‌ی snap که در بالا نام برده شد یا کتابخانه igraph و ...

برای ترسیم گراف نیز، کتابخانه‌ی NetworkX لی‌اوت‌های متفاوتی داردو نیز همانند دو ابزار قبل می‌توان تعداد نودها براساس معیارهای مختلف فیلتر کرد و نیز نودها را براساس همین معیارها سایزبندی و رنگ‌آمیزی کرد . اما وقتی تعداد نود حدودا بیش 2K باشد، (البته بستگی به سخت افزار مورد استفاده دارد که مقادیر گفته شده در این قسمت با توجه به سخت افزاریست که در ابتدای گزارش ذکر شد) زمان بسیار زیادی می‌گیرد و نکته‌ی دیگر اینکه هنگام ترسیم شبکه در دو ابزار قبلی می‌توان مرحله به مرحله ساخت گراف را شاهد بود و هرجا که لازم بود آن را متوقف کرد و نیز با کلیک بر هر نود موجود در visualization شبکه، اطلاعات آن نود را مشاهد کرد. اما با استفاده از کتابخانه‌ی NetworkX این کارها ممکن نیست.

و در نهایت، بهترین گزینه برای visualization شبکه بین سه ابزار موجود، جایگاه اول به نرم افزار Gephi تعلق دارد و بعد نرم افزار Cytoscape به دلیل محدودیتی که برای حافظه داشت و نیز لی‌اوت‌های بهتری که در نرم‌افزار Gephi وجود دارد. بهترین گزینه برای محاسبه‌ی معیارها و ویژگی‌ها و آنالیز شبکه، بین سه ابزار مورد بررسی، کتابخانه‌ی NetworkX می‌باشد.

به منظور اینکه بتوان به‌صورت همزمان از مزیت‌های هر یک از این ابزارها به هنگام آنالیز شبکه بهره برد، می‌توان معیارها مورد نظر را در گراف با استفاده از کتابخانه‌ی NetworkX محاسبه کرده و آن را ذخیره کنیم و سپس این گراف را در نرم افزار Gephi یا Cytoscape وارد کرده و براساس آن‌ها گراف را ترسیم کنیم.

قسمت دوم-تحلیل شبکه:

higgs_twitter (Mention Network):

مجموعه داده‌ی هیگز(higgs) پس از نظارت بر روند انتشار قبل ، حین و پس از اعلام خبر کشف ذره ای جدید با ویژگی elusive Higgs boson در توییتر در تاریخ ۴ ژوئیه ۲۰۱۲ به دست آمده است و پیام های ارسال شده در توییتر در مورد این کشف از تاریخ اول تا ۷ ژوئیه ۲۰۱۲ در نظر گرفته شده است.
مجموعه داده‌ی استفاده شده در این پروژه، قسمت شبکه‌ی mentionهای این مجموعه داده می‌باشد، که نشان دهنده‌ی کاربرانی است که در توییت های ریتوییت شده منشن(mention) شده اند. نودها نشان دهنده‌ی افراد و یال‌های ورودی نشان دهنده این که هر فرد توسط چند نفر منشن شده است و یال‌های خروجی نشان دهنده‌ی آن است که هر فرد در توییت هایی که در طی این جریان زده است، چند نفر را منشن کرده است. مجموعه داده استفاده شده برای تحلیل، یک شبکه‌ی جهت دار و وزن دار است.

توپولوژی شبکه:

در قدم اول، برای خواندن مجموعه داده‌ی مورد نظرکه شامل اطلاعات یال‌های شبکه است، از کتابخانه‌ی NetworkX و تابع read_edgelist() موجود درآن بهره گرفته شده است. که قطعه کد زده‌شده برای این قسمت به صورت زیر می‌باشد:

```
G = nx.read_weighted_edgelist("drive/MyDrive/higgsmention_network.edgelist",create_using=nx.DiGraph, nodetype=int)
```

اطلاعات کلی گراف که شامل تعداد نودها و یال‌ها و نیز متوسط درجه‌ی ورودی و خروجی می‌باشد که با استفاده از تابع info() موجود در کتابخانه‌ی NetworkX به دست آمده به شرح زیر می‌باشد:

```
print(nx.info(G))
```

Type: DiGraph

Number of nodes: 116408

Number of edges: 150818

Average in degree: 1.2956

Average out degree: 1.2956

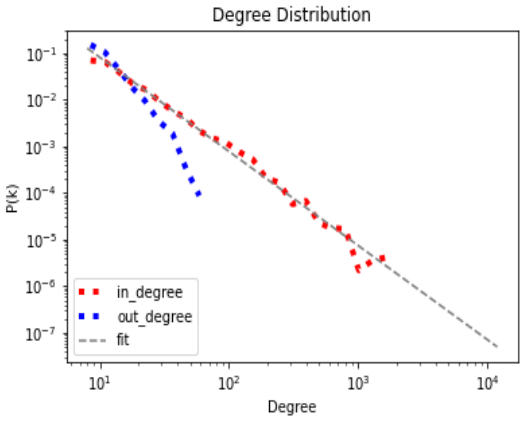
توزیع درجه(Degree Distribution):

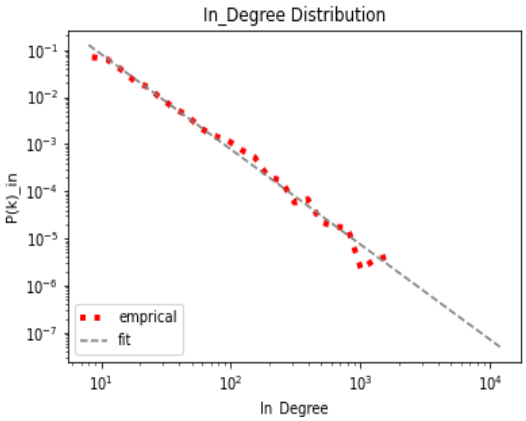
برای به دست آوردن توزیع درجه (پارامتر گاما : γ) از کتابخانه‌ی power law استفاده شده است. توزیع درجه، برای یال‌های خروجی و برای یال‌های ورودی به طور جداگانه محاسبه شده است. که نتیجه‌ی آن به شرح زیر می‌باشد:

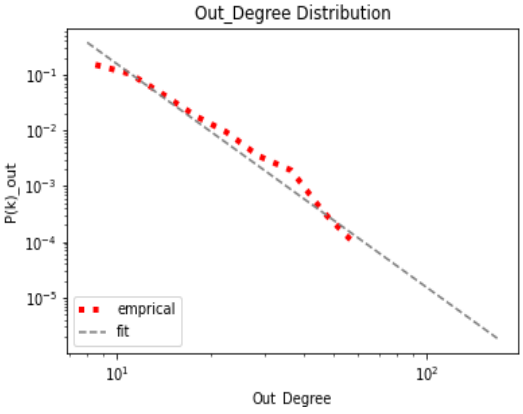
پارامتر γ برای یال‌های خروجی : 4.0045

پارامتر γ برای یال‌های ورودی : 2.0167

نمودار power law را بر روی نمودار توزیع توزیع درجه fit کرده‌ایم، نتیجه در تصویر پایین قابل مشاهده است:





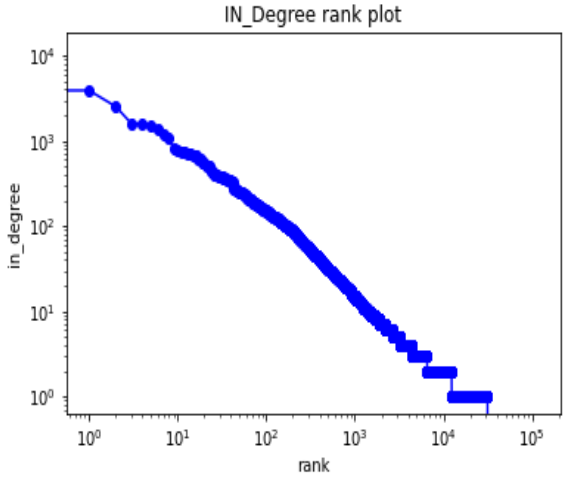


همانطور که می‌دانیم اگر $2 \leq \gamma \leq 3$ آنگاه ما با یک شبکه‌ی بدون مقیاس روبه رو خواهیم بود و به این معنی است که لینک‌هایی که یک نود می‌تواند داشته باشد، نسبت به متوسط تعداد لینک هایی که نودها در کل شبکه دارند، می‌تواند خیلی فاصله بگیرد و $< k^2 >$ واگرا می‌شود و درجه‌ی نودها به شکل زیر تعریف می‌شود :

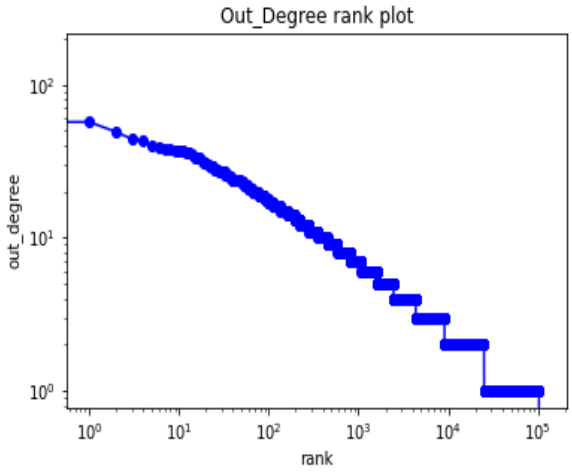
$K = <K> \pm \infty$

با توجه به توضیحات داده شده و پارامتر گامای به دست آمده برای یال‌های ورودی مجموعه داده‌ی مورد نظر و نیز با توجه به نمودارهای توزیع درجه شکل ۱، ما با یک شبکه‌ی بدون مقیاس روبه رو هستیم. در واقع به این معنی است که در شبکه‌ی مورد نظر ما یک فرد می‌تواند توسط تمام یا اکثر افراد موجود در شبکه mention شده باشد و به همین دلیل درجه ی ورودی(تعداد افرادی که یک فرد را mention می‌کنند) نودهای موجود در این شبکه این ویژگی را دارا هستند که حد و مقیاسی نداشته و از متوسط درجه ورودی نودها خیلی فراتر بروند. ماکسمیم درجه ورودی در شبکه "11953" است، به این منظور است که نودی که نقش هاب شبکه را دارد ، توسط "11953" منشن شده است. نمودار Rank درجه ورودی در شکل پایین نمایش داده شده است. طبق این نمودار تنها یک نود با که درجه‌ی ورودی "11953" داریم.

برخلاف آنچه در توزیع درجه‌ی ورودی افراد در بالا مشاهده کردیم، پارامتر گاما برای یال‌های خروجی بیشتر از 3 است و درجه‌ی خروجی افراد نمی‌تواند خیلی فراتر از متوسط درجه ی خروجی آن برود. ما در اینجا برای درجه ی خروجی حد و مرز داریم، پس از نظر توزیع درجه‌ی خروجی، یک شبکه‌ی بدون مقیاس نخواهیم داشت؛ این موضوع به این جهت اتفاق می‌افتد،که در واقع یک فرد این ویژگی را دارا نیست که بتواند تمام یا اکثر افراد را mention کند اما آن فرد می‌تواند توسط تمام یا اکثر افراد mention شود. ماکسمیم درجه ی خروجی 169 می‌باشد، به این معنی است که یک فرد نهایت توانسته است 169 نفر را mention کند. نمودار Rank درجه خروجی در شکل زیر نمایش داده شده است. طبق این نمودار تنها یک نود با که درجه‌ی خروجی "169" داریم.



نمودار Rank درجه‌ی ورودی



نمودار Rank درجه‌ی خروجی

چگالی:

چگالی گراف عبارت است از نسبت تعداد یال‌های گراف به کل تعداد یال‌هایی که گراف می‌تواند داشته باشد: (|E|تعداد یال‌های گراف و |V| تعداد رئوس گراف است.)

$$D = \frac{|E|}{|V|(|V|-1)}$$

برای به دست آوردن چگالی شبکه‌ی از تابع ()density موجود در کتابخانه‌ی NetworkX استفاده شده است و عدد به دست آمده به شرح زیر است:

Density = 0.00001112989

چگالی در گراف‌های بدون یال برابر 0 و در گراف کامل برابر 1 است واگر multigraph داشته باشیم، چگالی آن می‌تواند بیش از 1 باشد. در گراف هایی که دارای حلقه (self loop) می‌باشند، چون این حلقه‌ها جزو یال‌ها محسوب می‌شود، بنابراین در این گراف‌ها نیز چگالی می‌تواند بیش از 1 شود.

ضریب خوشه‌بندی(clustering coefficient):

برای به دست آوردن Avg clustering coefficient شبکه از کتابخانه‌ی snap و تابع ()GetClustCf موجود در آن استفاده شده است.(این تابع برای به دست آوردن ضریب خوشه‌بندی گراف را بدون جهت در نظر گرفته می‌گیرد). همچنین با استفاده از تابع ()clustering موجود در کتابخانه‌ی NetworkX ، این مقدار به ازای هر نود محاسبه‌شده‌است.

clustering coefficient Avg = 0.082511

تعداد مولفه‌های همبند:

چون مجموعه داده‌ی ما یک گراف جهت دار است، به همین جهت تعداد مولفه‌های متصل قوی و نیز متصل ضعیف را به دست آورده‌ایم، که به همین منظور از توابع ()number_strongly_connected_components و ()number_weakly_connected_components موجود در کتابخانه‌ی NetworkX استفاده شده است. مقادیر به دست آمده به شرح زیر است:

Number of SCC : 110704

Number of WCC : 10503

و همچنین تعداد یال‌ها و نودها در بزرگترین مولفه‌ی متصل قوی و نیز در بزرگترین مولفه‌ی متصل ضعیف به قرار زیر است:

Number of edges in SCC : 1801

Number of nodes in SCC : 7069

Number of nodes in WCC : 91606

Number of edges in WCC : 132068

قطر(Diameter) **و** Average Path Length :

قطر یک گراف عبارت است از ماکسمیم Eccentricity که Eccentricity خود به معنای ماکسیمم فاصله ای است که یک نود به بقیه نودها دارد. برای به دست آوردن قطر در شبکه‌ی هم از ابزار Gephi استفاده شده است و هم از تابع ()GetBfsFullDiam موجود در کتابخانه‌ی snap و هردو عدد 27 را به عنوان قطر مجموعه داده‌ی مورد نظر نشان می‌دهند.

Diameter : 27

Average Path Length: 8.19

همبستگی درجه نودهای مرتبط (assortativity):

برای به دست آوردن همبستگی درجه نودهای مرتبط از تابع ()degree_pearson_correlation_coefficient در کتابخانه‌ی NetworkX استفاده شده است. به جهت آنکه شبکه‌ی ما یک شبکه‌ی جهت دار است؛ همبستگی درجه نودها، هم برای درجه‌ی ورودی و هم برای درجه خروجی محاسبه می‌شود و به همین دلیل چهار نوع همبستگی درجه خواهیم داشت که نتایج به دست آمده به شرح زیر است:

۱) اگر source و target ، هردو یال ورودی باشند:

```
nx.degree_pearson_correlation_coefficient(G, x='in', y='in', weight="weight")
```

In-In-assortativity : - 0.0023

۲) اگر source و target ، هردو یال خروجی باشد:

```
nx.degree_pearson_correlation_coefficient(G, x='out', y='out', weight="weight")
```

Out-Out-assortativity : 0.0603

۳) اگر source یال ورودی و target یال خروجی باشد:

```
nx.degree_pearson_correlation_coefficient(G, x='in', y='out', weight="weight")
```

In-Out-assortativity : 0.0191

۴) اگر source یال خروجی و target یال ورودی باشد:

```
nx.degree_pearson_correlation_coefficient(G, x='out', y='in', weight="weight")
```

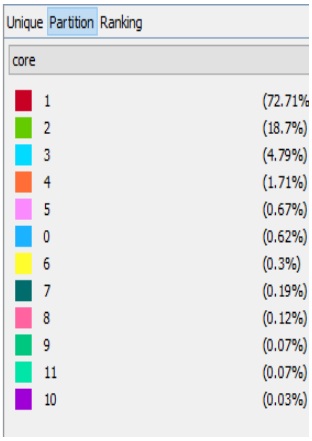
Out-In-assortativity : - 0.0207

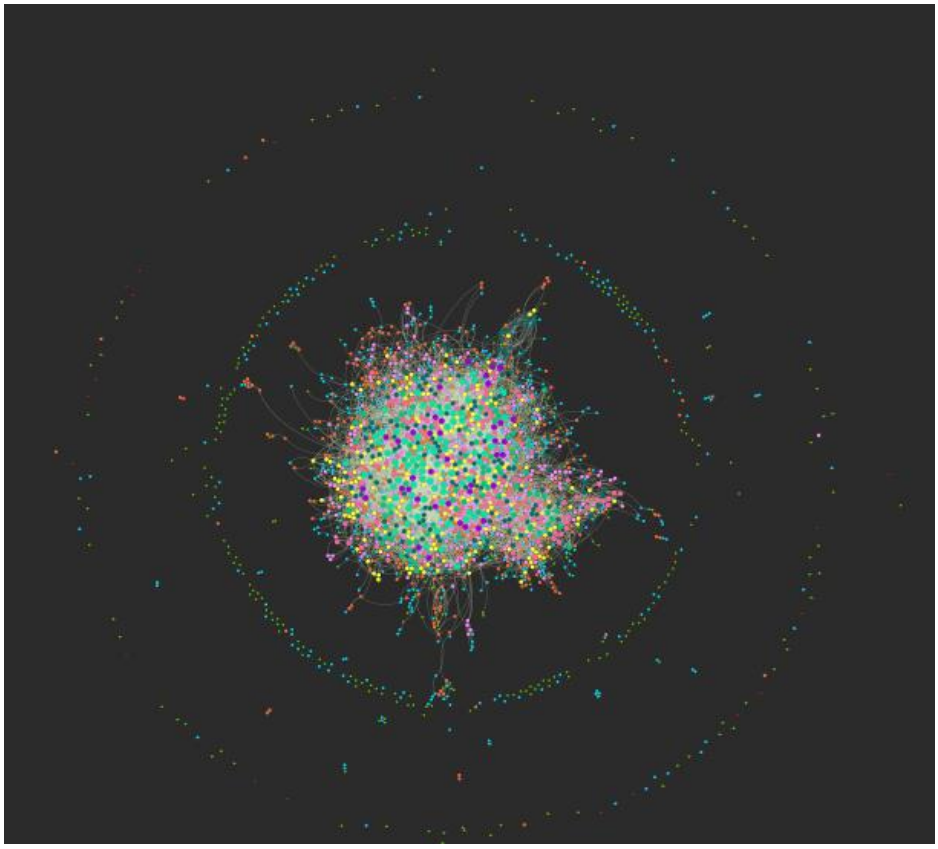
همانطور که از نتایج مشاهده می‌شود، شبکه برای حالت اول disassortative می‌باشد که به این معنی است، نودهای با درجه‌ی ورودی بالا تمایل بیشتر برای ارتباط با نودهایی با درجه‌ی ورودی پایین دارند؛ یا به عبارتی دیگر، افرادی که توسط حداکثر افراد موجود در شبکه منشن شده اند(افرادی که درجه‌ی ورودی بالایی دارند)، بیشتر با افرادی ارتباط دارند که توسط تعداد کمی از افراد موجود در شبکه منشن شده اند(افرادی که درجه‌ی ورودی پایینی دارند) یا به عبارتی دیگر افرادی که توسط تعداد زیادی از آدم‌ها منشن شده اند، خود، بیشتر افرادی را منشن کرده‌اند که توسط افرادی کمی در شبکه منشن شده اند. به طور عکس برای حالت دوم، همبستگی درجه نودها مثبت بوده و در واقع assortative می‌باشد؛که به این معناست، نودهای با درجه‌ی خروجی بالا تمایل بیشتری برای ارتباط با نودهای درجه خروجی بالا و نودهای با درجه خروجی پایین تمایل بیشتری برای ارتباط با نودهای مشابه خود دارند. در حالت سوم نیز همبستگی درجه نودها مثبت می‌باشد و assortative است و در حالت چهارم این مقدار منفی بوده و disassortativity داریم.

Centrality :

۱) K_shell : برای نودهای گراف یک مقدار تعریف می‌شود تحت عنوان K_shell، که در حقیقت این مقدار مشخص کننده‌ی پوسته‌ای (shell) است که نودها در آن قرار می‌گیرند. اگر ما الگورتیم K_core را به ازای K های مختلف روی گراف اجرا کنیم، مشاهده خواهیم کرد که هر چه K افزایش پیدا می‌کند، زیر مجموعه ای از مرحله‌ی قبل را نمایش خواهد داد یا به عبارتی دیگر K_core زیر مجموعه‌ای از (K-1)_core خواهد بود و (K-1)_core زیر مجموعه‌ای از (k-2)_core خواهد بود و این روند به همین ترتیب ادامه خواهد داشت. به این عملیات که K_core ها را مشخص کنیم و یک ساختار لایه‌ای از نودهای گراف تشکیل دهیم، K_shell Decomposition گفته می‌شود. به منظور مشخص کردن آنکه هر نود در مجموعه داده‌ی مورد استفاده در کدام shell قرار می‌گیرد، از تابع core_number() موجود در کتابخانه‌ی NetworkX استفاده شده است، که این تابع به هر نود عدد shell ای را که در آن قرار گرفته است، اختصاص می‌دهد. سپس به منظور visualization گراف و پارتیشن بندی نودها براساس

پوسته‌ای که در آن قرار گرفته‌اند، این مقادیر به همراه id هر نود در یک فایل csv ذخیره کرده و آن را در Gephi وارد می‌کنیم . ماکسیمم پوسته‌ای که نودها در این شبکه می‌توانند در آن حضور داشته باشند k =12 می‌باشد.
کدهای زده شده برای این قسمت در پیوست می‌باشد.

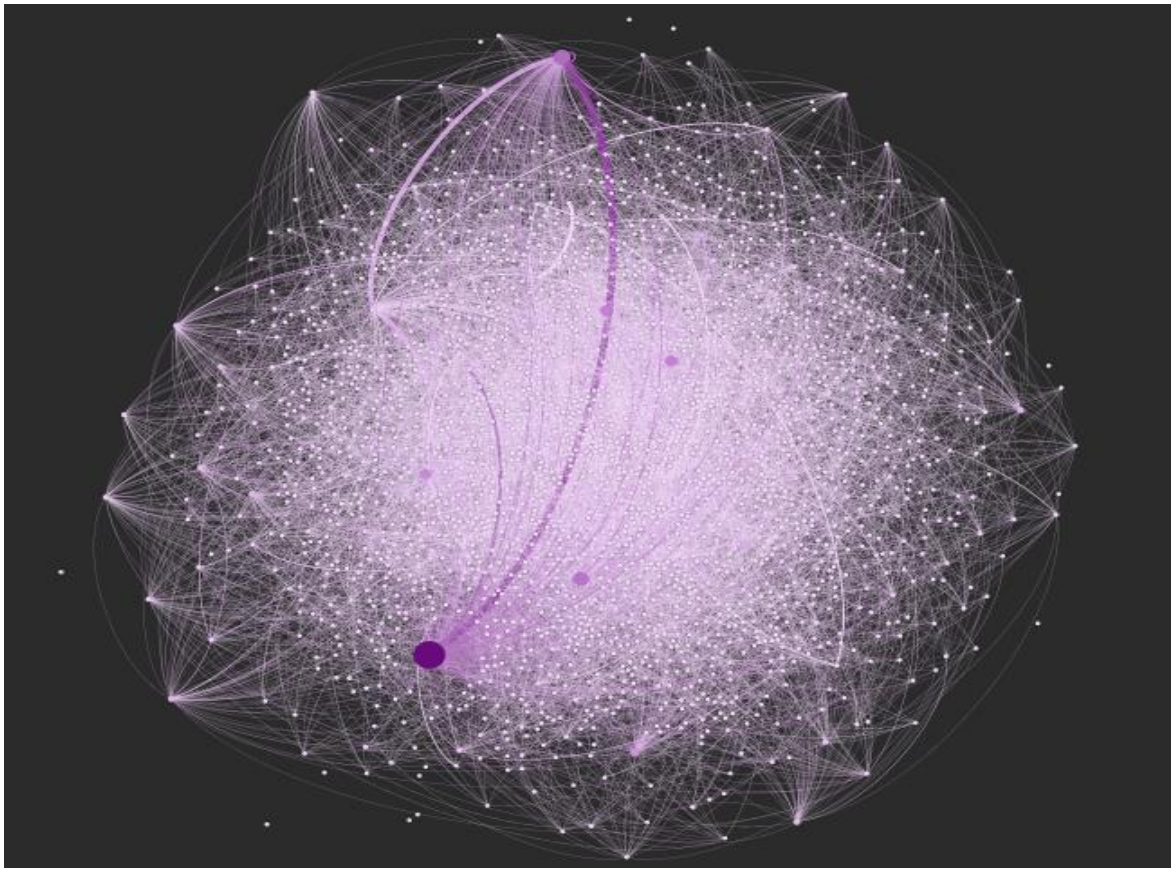




برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است و نیز با اعمال فیلتر Degree range ≥ 10 می‌باشد

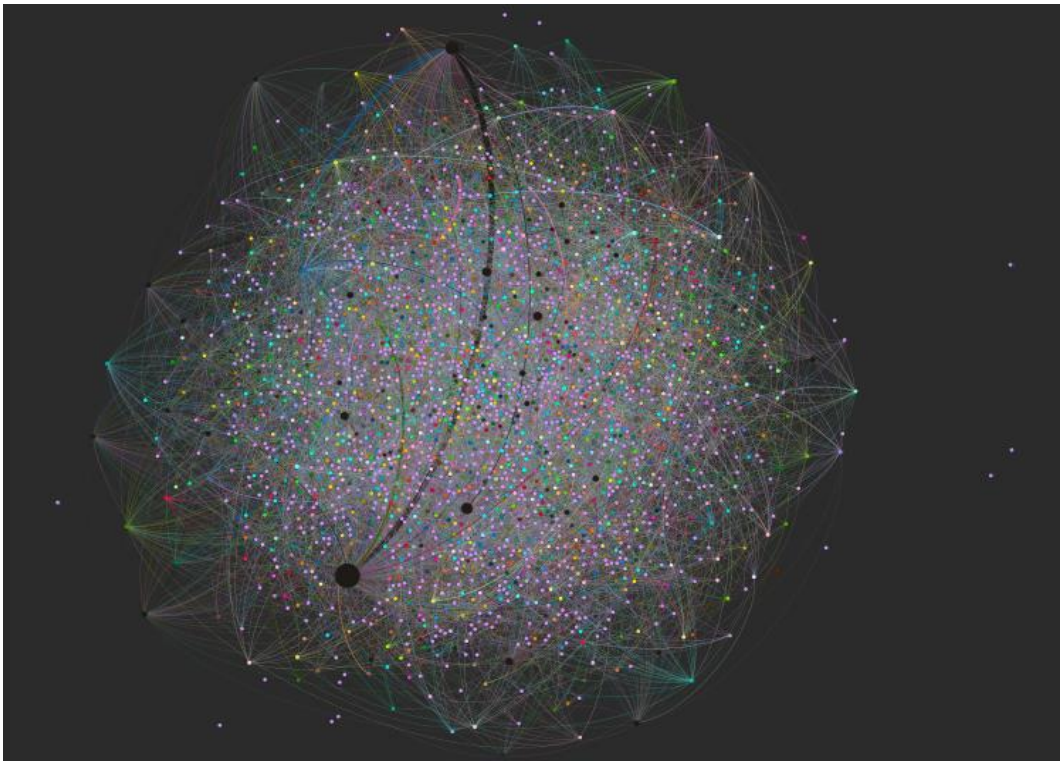
۲) HIT : با توجه به این معیار به هر نود دو امتیاز تعلق می‌گیرد : ۱) Authority ۲) Hub
نودهایی که good Hub هستند، در عمل محتوای مهمی ندارند، ویژگی که این نودها دارا می‌باشند، لینک به محتواهای خوب (good Authority) است و نیز نودی good Authority است که از Hub های خوبی به آن‌ها لینک داده باشند.
برای به دست آوردن معیار مرکزیت hit در شبکه ابتدا از تابع hits() موجود در کتابخانه‌ی NetworkX استفاده شده است. این تابع، دو مقدار برمی‌گرداند که یکی مقدار Hub نودها و دیگری مقدار Authority نودها می‌باشد.

بیشترین مقدار Authority برای نود با آیدی "88" بوده و مقدار آن "0.3027290" می‌باشدکه این نود نیز بیشترین درجه ورودی را در گراف دارا می‌باشد. بیشترین مقدار Hub نیز برای نود با آیدی "89805" بوده و مقدار آن "0.006013" می‌باشد و این نود هم دارای بیشترین درجه‌ی خروجی در کل شبکه است.
جایگاه نودها از لحاظ مقدار Authority و Hub در شکل‌های زیر نمایش داده شده است:



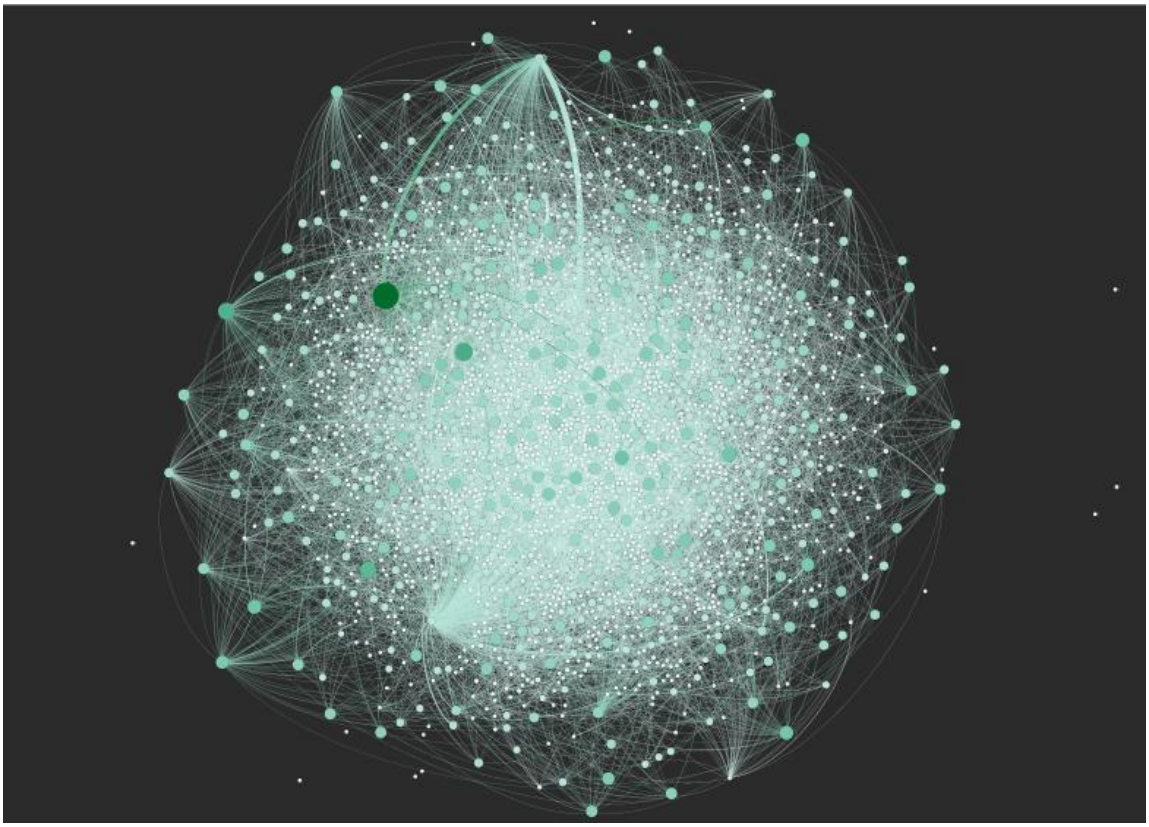
برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است و نیز با اعمال فیلتر Degree range ≥ 10 می‌باشد

نود بنفش پررنگ که سایز قابل توجهی نسبت به بقیه نودها دارد، همان نود با آیدی "88" است که بیشترین مقدار Authority را دارد، به این دلیل است که این نود بالاترین درجه‌ی ورودی را در کل شبکه دارا می‌باشد یا به عبارتی افراد زیادی در شبکه، فرد با آیدی"88" را منشن کرده و به آن لینک داده اند. که همین امر باعث می‌شود این نود از good Hubs زیادی لینک داشته باشد (از جمله نود با آیدی "89805" که بالاترین مقدار Hub را دارد) به همین دلیل مقداری که برای Authority این نود به دست می‌آید بیشترین مقدار است. هرچه از سایز نودها کاسته شده و رنگ آن نیز به سمت سفید می‌رود مقدار Authority آن نود نیز کاهش می‌یابد.



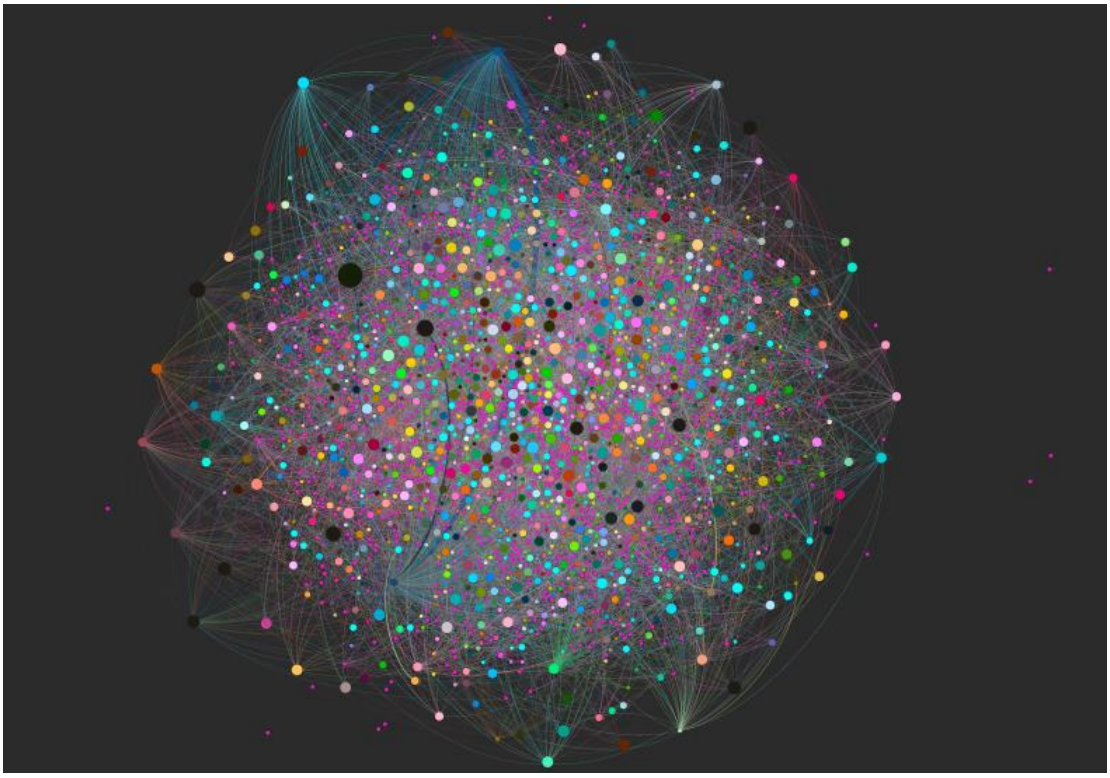
برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است و نیز با اعمال فیلتر Degree range ≥ 10 می‌باشد

این visualization بخشی از کل گراف است که براساس مقادیر authority نودها پارتیشن‌بندی شده اند و نیز هرچه از سایز نودها کاسته می‌شود، Authority نیز کاهش می‌یابد. رنگ بنفش که 87.75% نودهای گراف را شامل می‌شود، دارای Authority صفر می‌باشند و در نودهای سیاه رنگ این مقدار نسبت به بقیه رنگ‌ها بیشتر می‌باشد.



برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است و نیز با اعمال فیلتر Degree range ≥ 10 می‌باشد

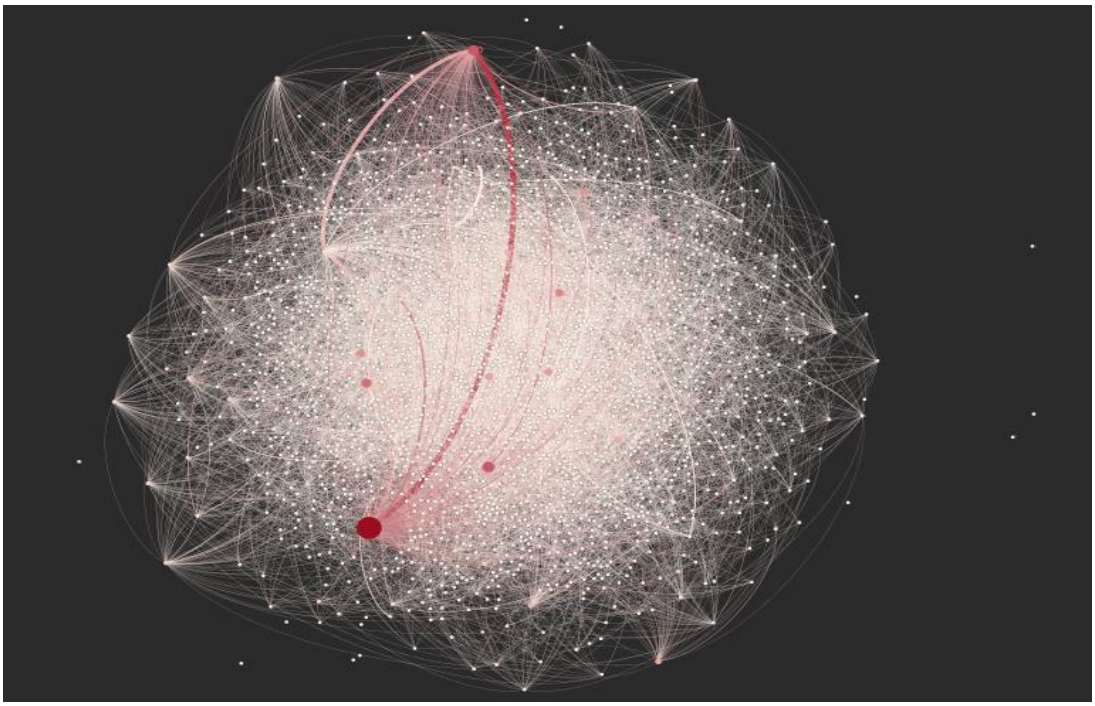
نودهای سبز رنگ که نسبت به بقیه نودها سایز بزرگتری دارند، دارای مقدار Hub بالایی هستند و هرچه رنگ نودها به سمت رنگ سفید رفته و سایزشان کوچک‌تر شده، مقدارHub در آن کاهش یافته است. علت آنکه مانند بالا تفاوت زیادی بین سایز نودهای با مقدار بالای Hub شاهد نیستیم، این است که مقدار Hub به دست آمده برای این نودها تفاوت بسیار کمی باهم دارند. نود سبز پررنگی که در شکل بالا مشاهده می‌کنید، همان نود با آیدی "89805" می‌باشد که بالاترین مقدار Hub در شبکه را دارد به دلیل اینکه این نود بالاترین درجه‌ی خروجی را در کل شبکه دارا می‌باشد، که بع این منظور است، این نود به good authorities زیادی لینک داده‌است (از جمله نود با آیدی "88" که بالاترین authority را دارد) به همین دلیل مقداری که برای Hub این نود به دست می‌آید بیشترین مقدار است.



برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است و نیز با اعمال فیلتر Degree range ≥ 10 می‌باشد

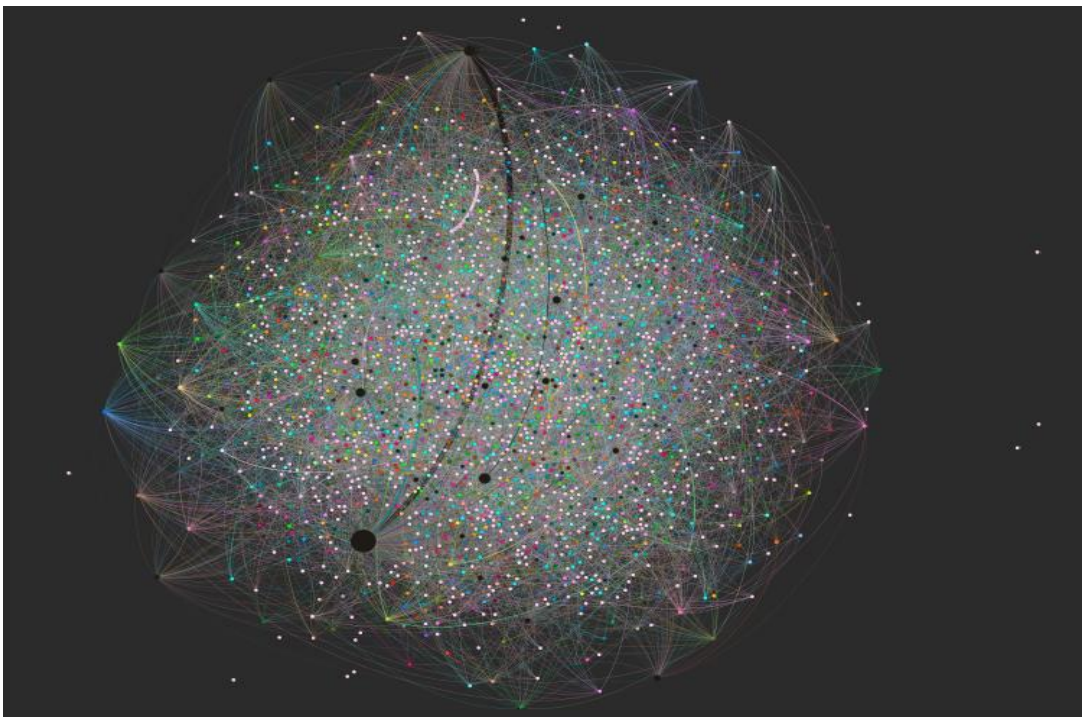
این visualization نیز بخشی از کل گراف است که براساس مقادیر به‌دست‌آمده برای Hub نودها پارتیشن‌بندی شده اند و نیز هرچه از سایز نودها کاسته می‌شود، مقدار Hub نیز کاهش می‌یابد.

۳)PageRank: در این معیار برخلاف معیار HIT به جای آن‌که دو عدد(Authority, Hub) به نودها اختصاص داده شود، یک عدد بر هر نود اختصاص داده می‌شود که معروف به Page Rank می‌باشد. به منظور محاسبه‌ی این معیار بر روی شبکه، از تابع ()pagerank موجود در کتابخانه‌ی NetworkX استفاده شده است. بیشترین مقدار این معیار برای نود "88" می‌باشد که برابر "0.02890" است. visualization نودها از لحاظ مقدار pagerank در شبکه به صورت زیر می‌باشد:



برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است و نیز با اعمال فیلتر Degree range ≥ 10 می‌باشد

همانطور که در تصویر نیز مشخص است، نود زرشکی رنگ که سایز بزرگتری دارد، همان نود با آیدی "88" است که بیشترین مقدار Page Rank را دارا می‌باشد و هرچه به سمت رنگ سفید می‌رویم و از سایز نودها کاسته می‌شود، مقدار Page Rank نیز کاهش می‌یابد.



برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است و نیز با اعمال فیلتر Degree range ≥ 10 می‌باشد

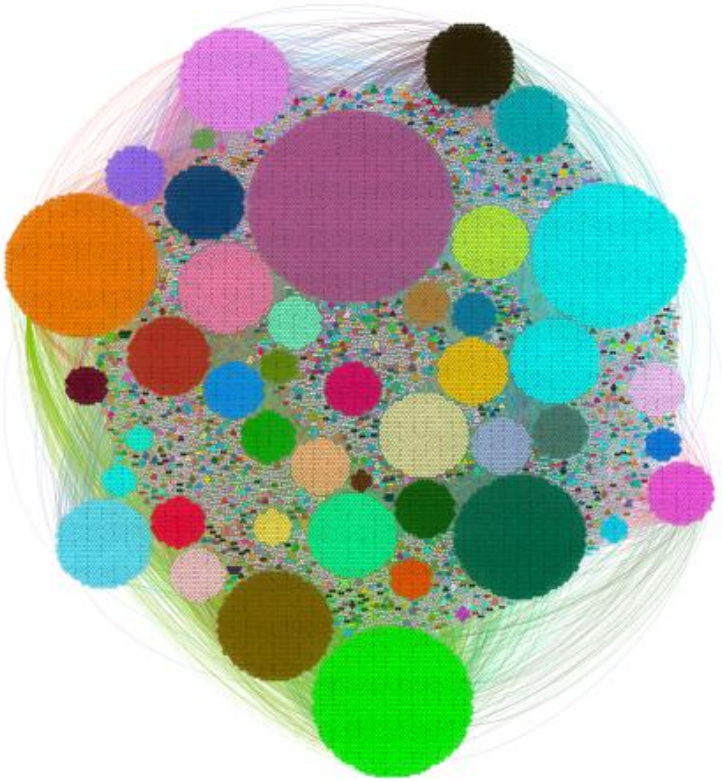
این visualization نیز بخشی از کل گراف است که براساس مقادیر به‌دست‌آمده برای Page Rank نودها پارتیشن-بندی شده اند وکه در اینجا نیز برای نودهای آبی کم‌رنگ مقدار Page Rank بالاتر از نودهای با رنگ دیگریست که در تصویر مشاهده می‌شود و نیز هرچه از سایز نودها کاسته می‌شود، مقدار Page Rank نیز کاهش می‌یابد. درصد قابل توجهی از شبکه(73.88%) نیز دارای Page Rank صفر می‌باشند.

: Community Detection

۱) K_core : این الگوریتم در واقع یک زیر گرافی را مشخص می‌کند که نودهای آن حداقل به k نود دیگر متصل باشند. در حقیقت در این گراف انتظار داریم که درجه‌ی هر نود حداقل برابر k باشد. K_core ها نواحی از گراف هستند که ما می‌توانیم انجمن‌های خوبی در آن پیدا کنیم. این الگوریتم از لحاظ مرتبه‌ی زمانی بسیار سریع بوده و نیز cost effective می‌باشد و یک روش فیلترینگ راحت است که می‌توان روی گراف اعمال نموده تا قسمت‌های بلااستفاده‌ی گراف حذف شود. سپس روش‌های تشخیص انجمن را که پیچیدگی زمانی بالایی دارند، به‌جای گراف اصلی در این زیرگراف محاسبه کنیم که این امر موجب می‌شود تایم اجرای این الگوریتم‌ها بسیار کاهش یابد.

- مقدار K_{max} برای شبکه‌ی ما برابر 12 می‌باشد که به این معناست، نهایت لایه‌ای که نودها میتوانند در آن قرار بگیرند، 13 است. تعداد نودها در این لایه برابر "44" عدد و تعداد یال‌ها برابر "415" عدد می‌باشد. با استفاده از این روش، در نرم‌افزار Gephi گراف را تا حد 2_core فیلتر کرده‌ایم که یک زیرگراف با "33769" نود و "75204" یال ساخته خواهد شد؛ سپس مقدار Modularity را برای نودهای باقی مانده در این زیرگراف محاسبه کرده‌ایم. تعداد انجمن‌های به دست‌آمده برابر است با "3589" که در حالت گراف اصلی با اعمال الگوریتم Modularity، تعداد انجمن‌های به دست آمده برابر "10648" می‌باشد. همانطور که در بالا نیز اشاره شد، تایم اجرای این الگوریتم بر روی گراف اصلی برابر 10 ثانیه و در زیر گراف 2_core برابر 2 ثانیه می‌باشد.

ترسیم زیرگراف 2_core با اعمال پارتیشن‌بندی براساس مقدار Modularity به صورت زیر می‌باشد:



برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است.

همانطور که مشاهده می‌کنید هر community رنگ مخصوص به خود را دارد ولی به علت محدودیت تولید رنگ در ابزار Gephi ، فقط قادر به رنگ آمیزی و مشخص کردن 1000 جامعه از بین جوامع موجود هستیم و بقیه جوامع به رنگ طوسی مشخص شده‌اند.

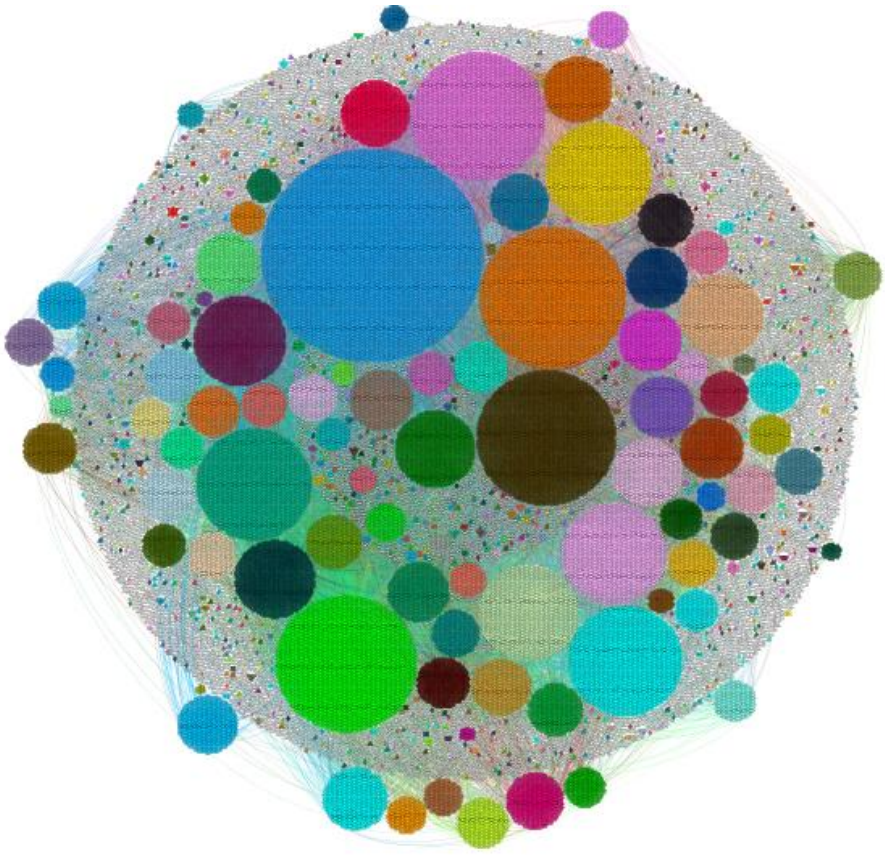
۲) **Louvain** : این الگوریتم، یک الگوریتم حریصانه‌ی سریع مبتنی بر Modularity optimization می‌باشد و شامل دو فاز است؛

فاز اول : ابتدا هر نود را به عنوان یک جامعه(community) مجزا در نظر می‌گیریم، سپس به ازای هر نود i و همسایه‌ی آن، نود j، Modularity به دست آمده با وجود نود i در جامعه j و بدون وجود نود i در این جامعه را محاسبه کرده و در هر حالتی که مقدارModularity ماکسیمم شود و مثبت باشد، نود i به community نود j اضافه می‌شود، در غیر این صورت همان حالت اولیه باقی‌می‌ماند. این روند به طور مکرر روی هر نود تکرار می‌شود تا جایی که community هیچ نودی تغییر نکند و ماکسمیم مقدار محلی برای modularity به دست آید. همچنین لازم به ذکر است که خروجی الگوریتم به ترتیب در نظر گرفتن نودها در اجرا، بستگی دارد، البته تفاوت چندانی در خروجی ایجاد نخواهد کرد و بیشتر منجر به افزایش یا کاهش زمان اجرا خواهد شد.

فاز دوم : در این فاز، یک شبکه‌ی جدید می‌سازیم که نودهای آن جوامعی هستند که در فاز قبلی به دست‌آمده‌اند. برای این کار، وزن یال بین دو جامعه، به وسیله‌ی جمع وزن لینک‌های بین نودهای موجود در آن دو جامعه به دست می‌آید. زمانی که فاز دوم به پایان رسید، می‌توان دوباره فاز اول را روی شبکه‌ی وزنی حاصل شده تکرار کرد.

به ترکیبی از این دوفاز "pass" گفته می‌شود. تعداد meta communities در هر pass کاهش می‌یابد و در نتیجه بیشتر زمان محاسبات مربوط به pass اول می‌باشد. این گذرها (pass) تا زمانی تکرار می‌شوند که دیگر تغییری در ایجاد نشده و حداکثر مقدار modularity به دست آید.[1] مرتبه‌ی زمانی این الگوریتم نیز برابر O(nlogn) می‌باشد.[2]

به منظور یافتن جوامع (communities) موجود در شبکه‌ی مورد بررسی، از پکیج community API و تابع community.best_partition() استفاده شده است. که تعداد جوامع به دست آمده در شبکه با استفاده از این الگوریتم برابر "10709" می‌باشد. متاسفانه به دلیل نداشتن Label نودها قادر به تحلیل هموفیلی اعضای یک انجمن نیستیم. اما خروجی این تابع را به عنوان یک فایل csv (که شامل id هر نود و شماره‌ی انجمنی است که در آن قرار گرفته) ذخیره کرده و در نرم‌افزار Gephi وارد کردیم. با استفاده از نرم‌افزار Gephi، شبکه را بر اساس این مقادیر پارتیشن‌بندی کرده و ترسیم کرده‌ایم . visualization به‌دست‌آمده به صورت زیر می‌باشد:

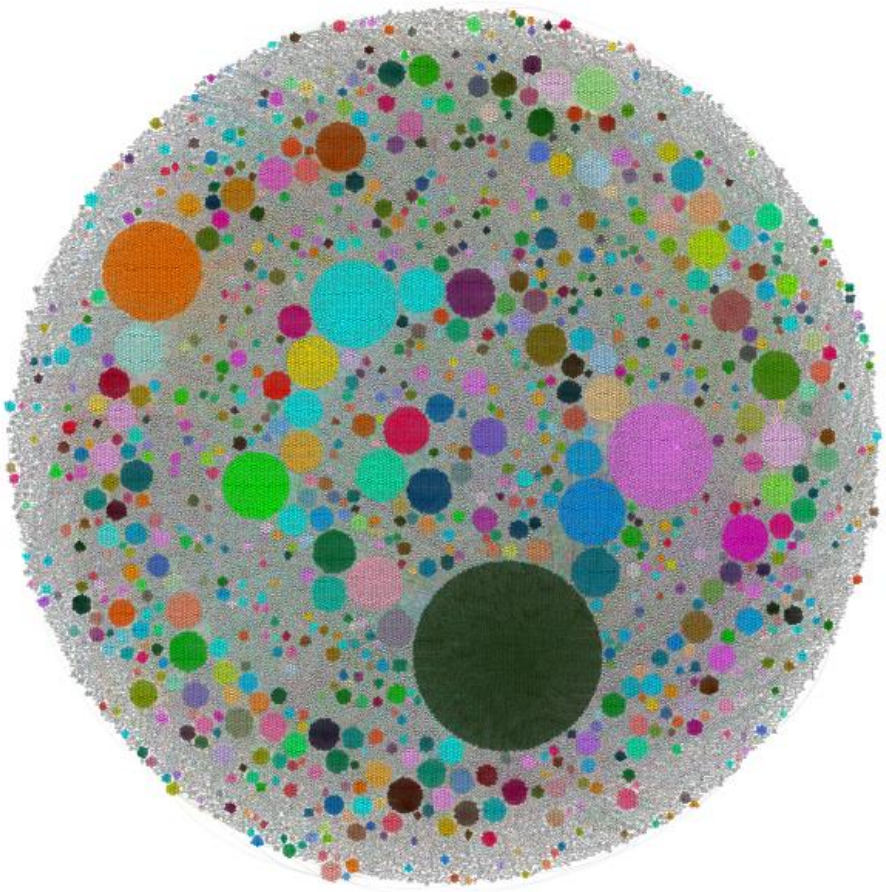


برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است.

همانطور که مشاهده می‌کنید هر community رنگ مخصوص به خود را دارد ولی به علت محدودیت تولید رنگ در ابزار Gephi ، فقط قادر به رنگ آمیزی و مشخص کردن 1000 جامعه از بین جوامع موجود هستیم و بقیه جوامع به رنگ طوسی مشخص شده‌اند.

۳) **Info Map** : این روش، یافتن بهترین ساختار انجمنی را معادل با کمینه‌سازی کمیت اطلاعات موردنیاز برای بیان گام تصادفی (Random walk) در یک شبکه میداند. کشف ساختار گراف با تعدادی گام تصادفی با طول‌های متفاوت و با بررسی احتمال پرش به یک گره تصادفی دیگر انجام خواهد شد. برای هر گره انجمن یک نام منحصربه فرد انتخاب می‌شود. در این حالت، گره‌های داخلی انجمن (گره‌های پرتکرار) نام کوتاه‌تری دارند. فشرده‌سازی مناسب در دو مرحله توسط کد هافمن انجام خواهد شد که یک مرحله آن برای تشخیص انجمن‌ها در گراف شبکه و مرحله دیگر برای تشخیص گره‌های درون یک انجمن است. بهترین حالت انجمن‌یابی، کوتاه‌ترین طول نامگذاری را در میان گام‌های تصادفی ممکن خواهد داشت. روش Infomap یک روش پرکاربرد و دقیق می‌باشد اما همپوشان نیست. پس این الگوریتم نیز مانند Louvain از دو فاز تشکیل شده است و اما فرق آن با الگوریتم Louvain این است که Info Map به دنبال بهینه کردن مقدار modularity نیست، بلکه به دنبال بهینه کردن مقداری تحت عنوان Map equation می‌باشیم. [3] مرتبه‌ی اجرای این الگوریتم در گراف های خلوت (sparse) نیز برابر O(nlogn) می‌باشد.

به منظور پیدا کردن انجمن‌ها با استفاده از این روش در شبکه مورد بررسی، تابع community_infomap() موجود در کتابخانه‌ی igraph استفاده شده است. خروجی این تابع را به عنوان یه فایل csv (که شامل id هر نود و شماره‌ی کلاستری است که در آن قرار گرفته) ذخیره کرده و در نرم‌افزار Gephi وارد کردیم. در تصویری که در پایین مشاهده می‌فرمایید، گراف باتوجه به Infomap به دست آمده برای هر نود، کلاستر بندی شده است و تعداد انجمن‌های به‌دست آمده برابر "16271" عدد می‌باشد.



برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است.

همانطور که مشاهده می‌کنید هر community رنگ مخصوص به خود را دارد ولی به علت محدودیت تولید رنگ در ابزار Gephi ، فقط قادر به رنگ آمیزی و مشخص کردن 1000 جامعه از بین جوامع موجود هستیم و بقیه جوامع به رنگ طوسی مشخص شده‌اند.

: (LPA) Label Propagation

این روش مبتنی بر توزیع اولیه تعدادی برچسب میان گره‌های شبکه است. هر گره برچسبی را اخذ می‌کند که بیشترین تکرار میان همسایگانش را دارد. قسمت‌های چگال شبکه تا حد امکان بسط می‌یابند و به برچسب‌های مشترک می‌رسند. در پایان کار، هر انجمن شامل تعدادی گره با برچسب یکسان (متفاوت با برچسب دیگر انجمن‌ها) خواهد بود. این روش نیازی به پارامتر ورودی ندارد. یک چالش مهم در انتشار برچسب، حالتی است که دو یا چند گره از همسایگان دارای برچسب با بیشترین فرکانس تکرار باشند. ساده‌ترین راه برای حل این مشکل، مکانیزم tie-breaking است بدین

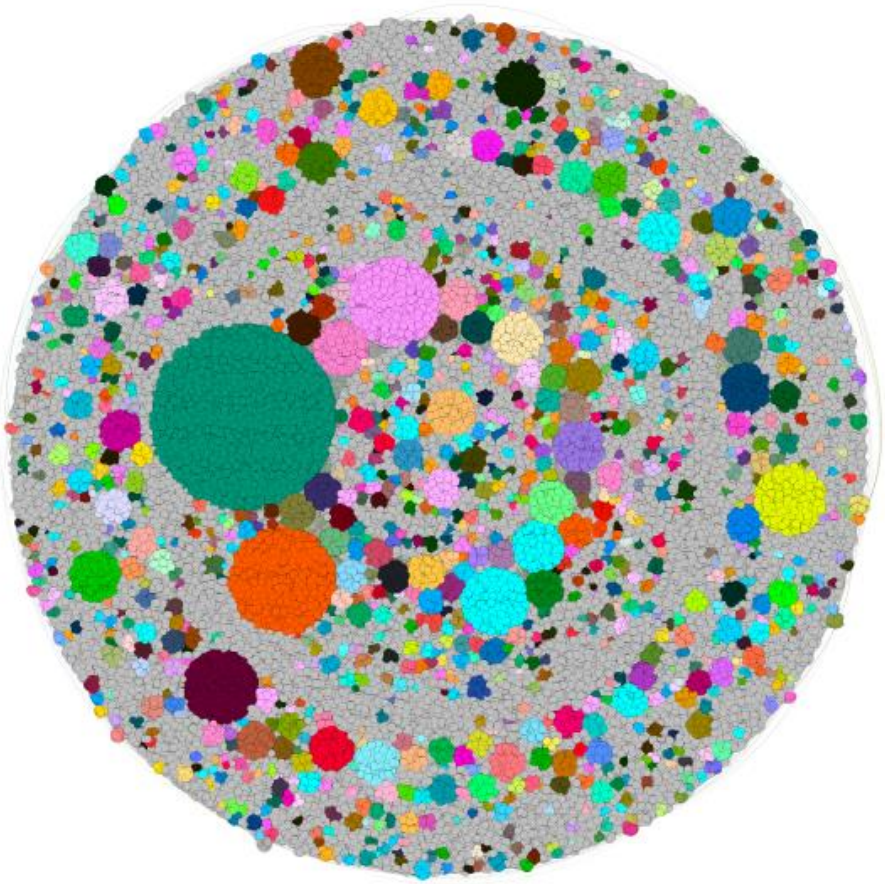
معنا که برچسب یکی از این همسایگان به‌طور تصادفی انتخاب شود و مراحل الگوریتم ادامه یابد. امکان اخذ برچسب-های متفاوت در حین اجرای روش باعث میشود تا LPA یک روش غیرقطعی باشد، ینی در هربار اجرا خروجی متفاوتی تولید می‌کند. اجرای چندباره الگوریتم، راهکاری است که برای این مورد توصیه میشود. این روش، سریع است و به‌عنوان یک روش مناسب در کار با گراف‌های کم‌تراکم مطرح می‌باشد. مشخص است که این روش از هم‌پوشانی حمایت نمی‌کند زیرا هر گره فقط یک برچسب دارد و لذا فقط به یک انجمن تعلق خواهد داشت.[4]

یکی از این تعمیم‌های LPA برای رسیدن به هم‌پوشانی، روش COPRA6 است. برای تحقق هم‌پوشانی، هرگره با زوج مرتب‌های (X_1, X_2) توصیف می‌شود که در آن، X_1 بیانگر شناسه یک انجمن خاص بوده و X_2 ضریب تعلق (بین 0 و 1) به آن انجمن را بیان می‌کند. در ابتدای این روش، به هر گره یک برچسب منحصربه‌فرد اختصاص می‌یابد و در هر مرحله اجرا، ضرایب تعلق هر گره بر اساس میانگین ضرایب همسایگانش تغییر می‌کند. اجرای این روش تا رسیدن به همگرایی و عدم‌تغییر برچسب‌ها ادامه می‌یابد. خروجی این روش نیز به دلیل استفاده از مکانیزم انتخاب تصادفی، غیرقطعی بوده و نیاز به تکرار در دفعات اجرا دارد.[5]

دیگر روش هم‌پوشان در انتشار برچسب (LPA)، روش SLPA است که با ایده نحوه‌ی مکالمه افراد با یکدیگر(شنیدن و تصمیم‌گیری بر اساس سابقه ذهنی) ارائه شده است . در این روش، هر گره حافظه‌ای دارد که سابقه برچسب‌ها را نگهداری می‌کند. این روش همچنین دارای دو پارامتر ورودی است که باید مقداردهی اولیه شوند: حداکثر تعداد مراحل اجرا و حدآستانه مرحله نهایی (پس‌پردازش). در هر مرحله، یک گره به‌عنوان شنونده انتخاب می‌شود. هر یک از همسایگان گره شنونده، به‌صورت تصادفی یک برچسب با احتمال متناسب با فرکانس وقوع آن در حافظه خودش را برای گره شنونده ارسال می‌کند. گره شنونده پرتکرارترین برچسب دریافت‌شده را به حافظه خود اضافه می‌کند. نهایتاً در مرحله پس‌پردازش، انجمن‌ها با اعمال حدآستانه تولید می‌شوند. روش SLPA برای گراف‌های کوچک دارای سرعت بالایی است.[7][6]

روشی به نام DEMON همپوشانی را در انجمن‌یابی شبکه‌های مقیاس‌آزاد (Scale Free)، مدنظر قرار می‌دهد. ایده کلی بدین شرح است: هر گره دید محدودی از کل شبکه دارد که شامل انجمن‌های عضو در آن‌ها و نیز انجمن‌های مجاور (محلی) می‌باشد. با ادغام این نقطه‌نظرات در یک رویکرد دموکراتیک می‌توان به ساختارهای همپوشان رسید یعنی عملاً خود گره‌ها در مورد ساختارهای انجمنی موجود قضاوت می‌کنند. برای تعبیر دیدگاه هر رأس دلخواه i نسبت به کل گراف، یک شبکه Ego برای آن رأس تعریف می‌شود که معادل با زیرگرافی شامل رأس i و رئوس مجاور آن و یال‌های موجود بین این رئوس است. گروهی از روش‌ها، انتشار برچسب را با یک مسئله استنتاج آماری مدل میکنند. تابع تعلق این روشها بجای یک عدد ساده، مشتمل بر برداری از احتمالات است. P(X, Y) احتمال وجود یال میان گرهی از انجمن X با گرهی از انجمن Y را بیان می‌کند و با میانگین وزنی P(X, X) و P(Y, Y) به دست می‌آید.[8] هیچ کدام از روش‌های بالا قادر به از بین بردن غیر قطعی بودن این الگوریتم نخواهد بود.

به‌منظور تشخیص انجمن توسط label propagation، کد زده شده جهت پیاده‌سازی این الگوریتم؛ به این صورت عمل می‌کند که، ابتدا به هر نود یک برچسب مجزا انتساب می‌دهد، سپس به صورت ناهمزمان (asynchronous) به هر نود ، برچسبی که به تعداد بار بیشتری در نودهای مجاور آن تکرار شده‌است(در مرحله قبل)، انتساب می‌دهد. این فرایند تا جایی ادامه می‌یابد که برچسب نودها تغییر نکنند. در نهایت نودهای با برچسب مشابه تشکیل یک انجمن خواهند داد. (کد این قسمت، با نام " Label_propagation_async.py " در پوشه‌ی "Code" قرار گرفته است.) براساس این کد، انجمن‌های موجود در شبکه را محاسبه کردیم که تعداد انجمن‌های به‌دست‌آمده برابر "21657" می‌باشد. سپس این اطلاعات به دست‌آمده را در یک فایل csv ذخیره نموده و در نرم‌افزار Gephi وارد کرده‌ایم. گراف را بر اساس آن پارتیشن بندی و رسم نمودیم؛ نتیجه در تصویر زیر قابل مشاهده می‌باشد:



برای visualization از برنامه‌ی Gephi استفاده شده است و multiGravity force atlas 2 برای layout انتخاب شده است.

الگوریتم **label propagation asynchronous**:

```
import networkx as nx
import collections
import numpy as np
import csv


def read_graph_from_file(path):
    print("loading Graph...")
    graph = nx.read_edgelist(path, nodetype=int, edgetype=int,
data=(("weight", float),))
    if nx.is_directed(graph):
        UNdir_graph = nx.to_undirected(graph)
        return UNdir_graph
    else:
        return graph


def save_community(community):
    dict_com = {}
    for index, item in enumerate(community):
        for node in item:
            dict_com[node] = index

    print("Saving in file...")
    with open("community/labelPropagation.csv", 'w', newline='') as csv_file:
        writer = csv.writer(csv_file)
        writer.writerow(["Id", "lpa"])
        for key, value in dict_com.items():
            writer.writerow([key, value])


def lpa_communities(G):
    print("compute LPA...")
    # initial label for each node
    labels = {n: i for i, n in enumerate(G)}

    # at least one node label change
    exp = True
    while exp:
        exp = False

        shuffled_nodes = list(G.nodes())
        np.random.shuffle(shuffled_nodes)

        for node in shuffled_nodes:
            u_nbr = G[node]
            if len(u_nbr) > 0:

                nbr_labels = [labels[v] for v in u_nbr]
                nbr_label_counter = dict(collections.Counter(nbr_labels))
                max_freq_label = max(nbr_label_counter.values())

                best_labels = [k for k, v in nbr_label_counter.items() if
```

```
                v      ==    max_freq_label]
                v      ==    max_freq_label]


                choosed_label = np.random.choice(best_labels)

                if labels[node] != choosed_label:
                    labels[node] = choosed_label
                    exp = True

        label_to_nodes_dict = {}
        for n, label in labels.items():
            if label in label_to_nodes_dict:
                label_to_nodes_dict[label].append(n)
            else:
                label_to_nodes_dict[label] = [n]
        return list(label_to_nodes_dict.values())


G = read_graph_from_file("higgs-mention_network.edgelist")
lpa = lpa_communities(G)
save_community(lpa)
```

تابع‌های read_graph_from_file() و save_communities() همانطور که از اسم آن‌ها مشخص می‌باشد، گراف را از فایل خوانده و اگر جهت‌دار باشد، آن را بدون جهت می‌کند (زیرا این الگوریتم جهت دار بودن گراف را تشخیص نمی‌دهد.) و تابع بعدی، عددی که به نود در تابع lpa_communities() انتساب داده می‌شود را به همراه id آن نود در فایل csv ذخیره می_کند.

مقایسه روش‌های تشخیص انجمن :

نام روش	مدل	پیچیدگی زمانی	جهت‌دار بودن گراف ورودی	وزن دار بودن گراف ورودی	نیاز به پارامتر ورودی	همپوشانی	سال
Louvain	تابع کیفیت	O(m)	×	✓	ندارد	×	2008
LPA	انتشار	O(m+n)	×	✓	ندارد	×	2007
COPRA	انتشار	O(mlog(m/n))	×	✓	ندارد	✓	2010
SLPA	انتشار	O(Tm)	✓	✓	دارد	✓	2012
DEMON	انتشار	$O(nK^{3-\alpha})$	×	×	ندارد	✓	2012
Infomap	نزدیکی	O(nlogn)	✓	✓	دارد	×	2008

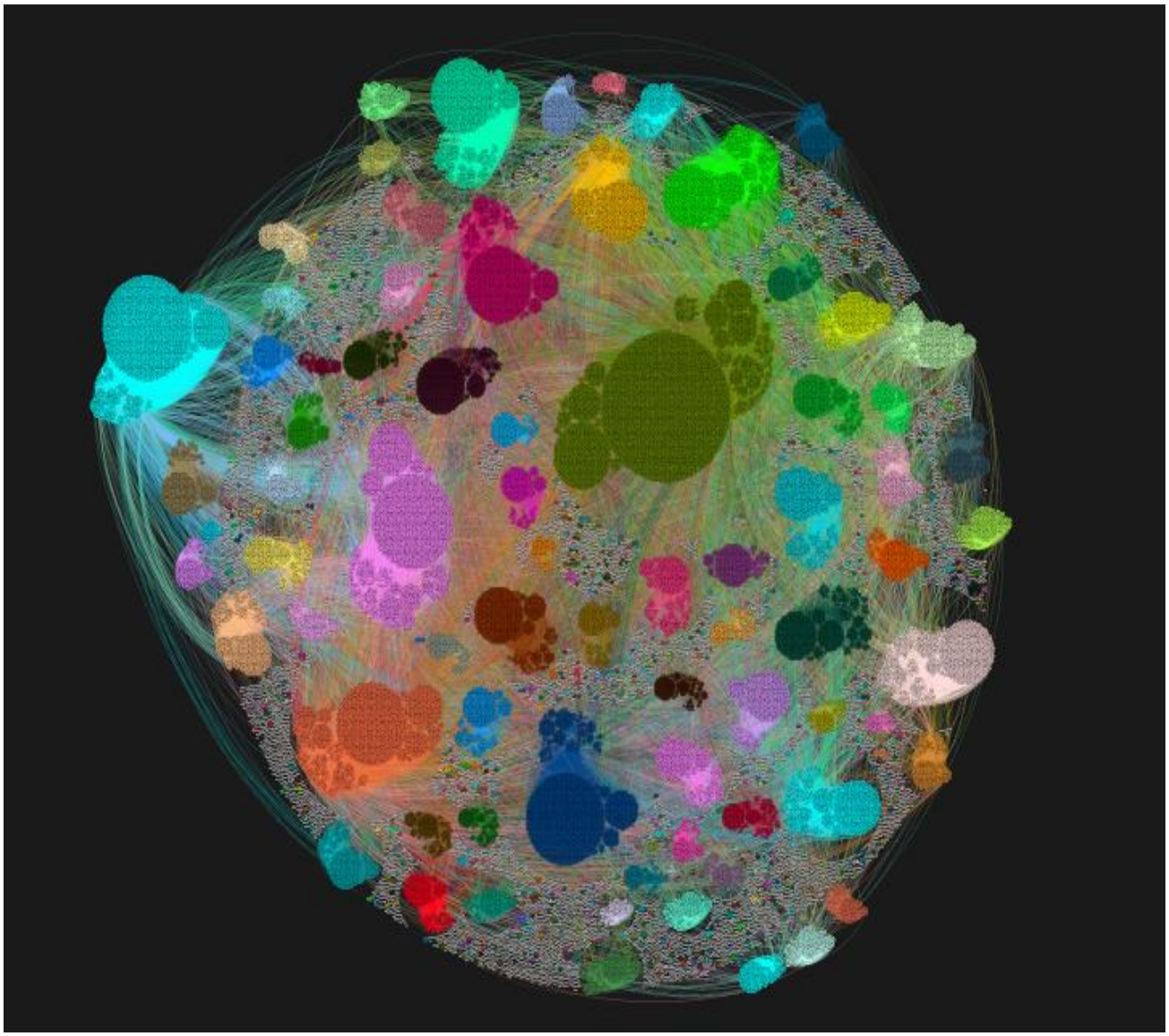
n = تعداد رئوس، m = تعداد یال‌ها، K = بیشینه درجات رئوس، α = ضریب توانی، T= تعداد دفعات اجرای الگوریتم

نام روش	زمان اجرا	تعداد انجمن‌ها
Louvain	44 s	10709
LPA	8 s	21656
Infomap	2 min	16271

مقایسه زمان اجرا و تعداد انجمن‌های تشخیص داده شده توسط سه روش Louvain، Label Propagation و Infomap

همانطور که در جدول بالا مشاهده می‌فرمایید، کمترین زمان اجرای الگوریتم‌های تشخیص انجمن، مربوط به الگوریتم label propagation async می‌باشد، بعد از آن روش Louvain و در نهایت روش Infomap است. تعداد انجمن‌های تشخیص داده شده به وسیله‌ی lpa نیز از دو روش دیگر نیز بیشتر می‌باشد و سپس Infomap و در نهایت Louvain قرار دارد.

ترسیم گراف:



تصویری که در بالا مشاهده می‌فرمایید با استفاده از ابزار Gephi و لی‌اوت "circle pack layout" می‌باشد. ابتدا مقداری modularity برای گراف محاسبه شده، سپس نودها براساس هر انجمنی در آن قرار می‌گیرند، رنگ‌آمیزی شده‌اند. تعداد انجمن‌ها "10637" عدد می‌باشد. در ترسیم بالا به علت محدودیت در رنگ‌آمیزی هر 10637 انجمن، تنها 1200 عدد از آن‌ها که بزرگترین انجمن‌ها نیز می‌باشند، رنگ‌آمیزی شده‌اند و انجمن‌های کوچکتر طوسی رنگ می‌باشند.

مراجع:

1. https://snap.stanford.edu/snappy
2. V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *J. Stat. Mech.*, October2008.
3. https://en.wikipedia.org/wiki/Louvain_method
4. M. Rosvall, C. T. Bergstrom, Maps of information flow reveal community structure in complex networks, Proceedings of the National Academy of Sciences Usa (2007) 1118–1123.
5. Raghavan, U. N., Albert, R., Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks, Physical review E, Vol. 76, p. 036106
6. Gregory, S. (2010). Finding overlapping communities in networks by label propagation, New Journal of Physics, Vol. 12, p. 103018.
7. J. Xie, B. K. Szymanski and X. Liu. SLPA: Uncovering Overlapping Communities in Social Networks via A Speaker-listener Interaction Dynamic Process. In *Proc. of ICDM 2011 Workshop*, 2011
8. J. Xie and B. K. Szymanski. Towards linear time overlapping community detection in social networks. In PAKDD, pages 25-36, 2012
9. Coscia,M., Rossetti, G., Giannotti, F., Pedreschi, D. (2012). Demon: a local-first discovery method for overlapping communities, in Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 615-623.