Community Detection From Research Articles

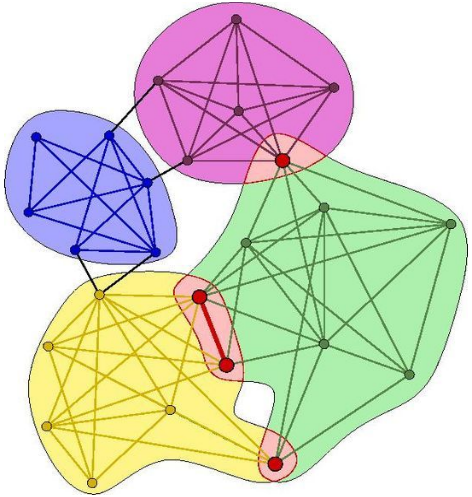
Anupriya Gupta-201505582
Rashi Chauhan-201506527
Munmun Chowdhary-201506593

Contents

Introduction

What is a Community?

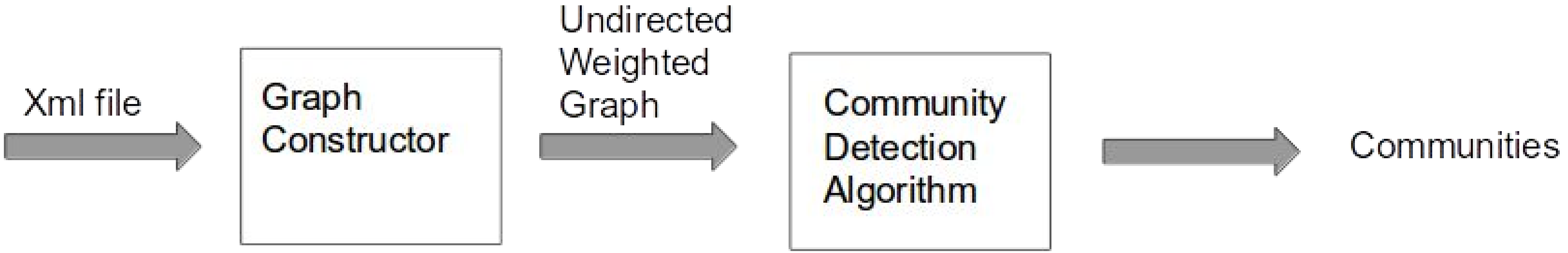
A community is defined as the group of nodes which are densely connected inside the group, while loosely connected with the nodes outside the group i.e group of dense graphs within a sparse graph.



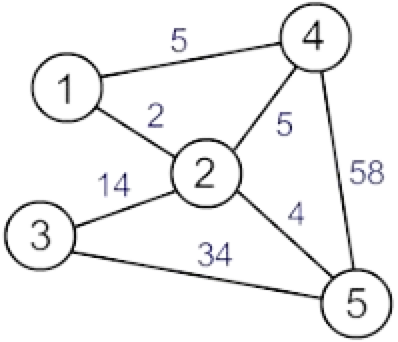
Problem Statement

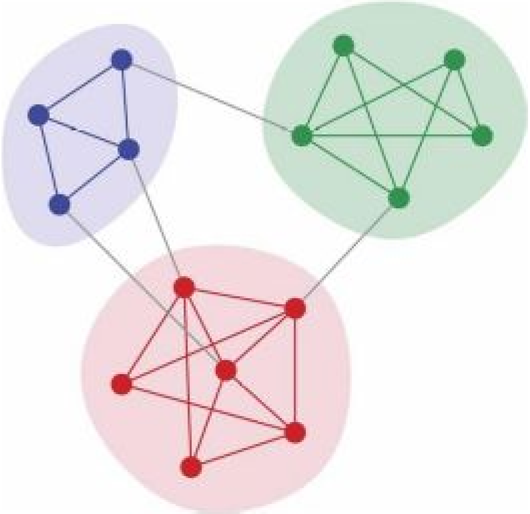
- The task is to come with an algorithm to detect communities within this network of Research articles and their Authors.
- The problem was that the algorithm should scale up for graphs containing millions of nodes.
- The quality measure should be such that it helps to analyze the network.

Solution


Xml file
Graph Constructor
Undirected Weighted Graph
Community Detection Algorithm
Communities


tutorialspoint.com
<?xml version="1.0" encoding="utf-8" ?>
- <Authors>
- <author code="1">
<fname>Zara</fname>
<lname>Ali</lname>
</author>
- <author code="2">
<fname>Priya</fname>
<lname>Sharma</lname>
</author>
- <author code="3">
Show Author File


1
2
3
4
5
5
2
5
14
58
4
34



Description

- Research Articles/ Authors are represented in a form of a network or a graph.
- The nodes represent the participating entities . In our case, entities are authors.
- The edges represent the relation between authors.

Dataset


```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE dblp SYSTEM "dblp.dtd">
<dblp>

[...]

<article key="journals/cacm/Gentry10" mdate="2010-04-26">
<author>Craig Gentry</author>
<title>Computing arbitrary functions of encrypted data.</title>
<pages>97-105</pages>
<year>2010</year>
<volume>53</volume>
<journal>Commun. ACM</journal>
<number>3</number>
<ee>http://doi.acm.org/10.1145/1666420.1666444</ee>
<url>db/journals/cacm/cacm53.html#Gentry10</url>
</article>

[...]

<inproceedings key="conf/focs/Yao82a" mdate="2011-10-19">
<title>Theory and Applications of Trapdoor Functions (Extended Abstract)</title>
<author>Andrew Chi-Chih Yao</author>
<pages>80-91</pages>
<crossref>conf/focs/FOCS23</crossref>
<year>1982</year>
<booktitle>FOCS</booktitle>
<url>db/conf/focs/focs82.html#Yao82a</url>
<ee>http://doi.ieeecomputersociety.org/10.1109/SFCS.1982.45</ee>
</inproceedings>

[...]

<www mdate="2004-03-23" key="homepages/g/OdedGoldreich">
<author>Oded Goldreich</author>
<title>Home Page</title>
<url>http://www.wisdom.weizmann.ac.il/~oded/</url>
</www>
```

Tags and their meanings:

.article – An article from a journal or magazine.

.inproceedings – A paper in a conference or workshop proceedings.

• proceedings – The proceedings volume of a conference or workshop.

• book – An authored monograph or an edited collection of articles.

• incollection – A part or chapter in a monograph.

• phdthesis – A PhD thesis.

• mastersthesis – A Master's thesis. There are only very few Master's theses in dblp.

• www – A web page. It contains all the aliasing of the authors

Parsed Dataset

Titles and Corresponding Authors ID

```
1 Book Review: The Taming of the True, by Neil Tennant.#122362
2 Some lattice attacks on DSA and ECDSA.#80432
3 Anycast Routing Protocol for Forest Monitoring in Rechargeable Wireless Sensor Networks.#99165
4 Design of Energy-Efficient Application-Specific Instruction Set Processors (ASIPs), Tilman Glokler, Heinrich Meyr, Kluwer Academic
  Publishers, Boston, 2004, ISBN 1-4020-7730-0, Hardcover, pp 234, plus XX.#119539
5 Fast Ant Colony Optimization on Runtime Reconfigurable Processor Arrays.#135308
6 Two levels autonomic resource management in virtualized IaaS.#3254
7 The Importance of Digital Libraries in Joint Educational Programmes: A Case Study of a Master of Science Programme Involving
  Organizations in Ghana and the Netherlands.#106455
8 Engineering high-performance legacy codes as CORBA components for problem-solving environments.#111764
9 Parallel color space converters for JPEG image compression.#119436
10 Identification-robust simulation-based inference in joint discrete/continuous models for energy markets.#18461
11 Release of hazardous substances in flood events: Damage model for atmospheric storage tanks.#86299
12 Application of Bayesian nonparametric models to the uncertainty and sensitivity analysis of source term in a BWR severe accident.#86816
13 Constructing Application-Specific Memory Hierarchies on FPGAs.#38860
14 To theme or not to theme: Can theme strength be the music industry's "killer app"?#127430
15 Type II Reverse Engineering [For Good Measure].#64912
16 Agent-based distributed architecture for mobile robot control.#115414
17 Some computer experiments in picture processing for data compaction.#38622
18 Rule-preserved object compression in formal decision contexts using concept lattices.#12461
19 Implementing Discrete-time Fractional-order Controllers.#66547
20 Smart memory architecture and methods.#3306
21 What Makes Measuring Software So Hard?#140865
22 Identification of a modified Wiener-Hammerstein system and its application in electrically stimulated paralyzed skeletal muscle
  modeling.#34075
23 A systolic algorithm for extracting regions from a planar graph.#38607
24 Logical fallacies as informational shortcuts.#63819
25 Nonclassical Mereology and Its Application to Sets.#121105
26 System identification and control design using P.I.M. + software: I. D. Landau.#30160
```

Authors and corresponding IDs(to remove aliasing)

```
A'fza Shafie#762404
A'zraa Afhzan Ab Rahim#1007942
A-Chuan Hsueh#809347
A-Imam Al-Sammak#930816
A-Nasser Ansari#635222
A-Ning Du#1412454
A-Qun Deng#253952
A-Ra Cho#802343
A-Ram Choi#1574426
A-Rang Jeong#241939
A-Reum Bae#1354450
A-Rum Jun#1260724
A-Xing Zhu#1659073
A-Yeon Park#482302
A-Youn Park#888121
A-Young Cho#559266
A-rom So#1346114
A. (Zizo) Farrag#1269616
A. A'Campo-Neuen#605946
A. A. (Louis) Beex#262515
A. A. A. Darwish#736394
A. A. A. Kock#1300450
A. A. A. Nasser#791846
A. A. A. Samat#1470345
A. A. Aaby#1526059
A. A. Abd El-Aziz#953730
A. A. Abd El-Latif#421231
A. A. Abd Elaziz#941750
A. A. Abd-Allah#817312
A. A. Abd-Ellatif#525162
A. A. Abdel Kader#1463744
```

Inproceedings and authors

q1.c × q2.cpp × README × Louvain.py × article_journal_author1.txt × titJouAuth.txt × titJou.txt × titid.txt × auth_id.txt × inproceedings.txt ×

```
1647309|1535536|1536200|774170|178029|177987|231650|1536216|231650|231362|1590204|165381|348925|940600|976552|1536216|172468|166045|653514|
1430154|1600095|452004|1535634|1375567|302980|349007|210152|1536006|574133
15. WLP>>1600400|68287|530026|655446|231027|1286300|744656|878931|1058197|1590460|146461|146269|302905|1461174|1287813|1680990|1014934|
165399|204640|878931|178000
1999 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery>>1670593|774409|103007|1090503|262095|1693007|1368605|
177886|1590336|75918|1600663|1428921|1090503|1590479|326647|348974|230949|1090503|1600429|1264635|206909|103072|1536078|1669791|1461214|
103360|177298|474726|132490|1536077|618974|173850
25 Years CSP>>1373601|349007|326339
25 Years Communicating Sequential Processes>>1679139|512917|14159|1670623|177672|992938|1091110|605331|1590525|1590297|116796|6556|1590525|
1535510|303201|1600123|1536220|172468|1535797|1467185|1507092|326613|655553|177753|1600123|1316339
25 Years GULP>>177612|1536123|114552|103381|1590306|1669818|1461176|124269|1590460|177904|166190|1090598|1590379|103205|1461363|103206|
1669752|102594|178044|178203|1536105|349217|1461176|671734|1473002|1493406|349118|1005414|682488|1281952|231403|173571|1416265|1589927|
114755|1670334|102718|81429
25 Years ISCA: Retrospectives and Reprints>>1431533|1669866|804449|114165|339147|1670095|272050|1293685|209934|1420174|185134|1333878|
479657|1020111|1484428|1600610|103289|1590329|1600659|1049994|626855|1027421|134362|231219|102957|1670095|325730|1000584|1535544|127723|
735650|862885|177811|1670095|325730|304242|260648|1011577|1621415|1087925|880870|272050|1011577|114606|325932|704734|102957|165820|1670408|
231488|185134|1600659|1600610|17769|1621415|1590338|1090927|1670454|804449|114165|983744|1020111|872398|103289|1600659|50964|1090567|
1360040|1585679|209989|1025411|703466|177811|1425566|165403|880870|1087925|1049994|626855|1368621|1353380|1600416|1600659|50964|1090567|
1360040|1585679|209989|1025411|325932|325932|1669841|339147|1670095|231219|177222|851724|231488|44666|1276827|976813|74845|348831|646842|
114740|165903|767716|1488542|1224665|703036|1435253|326495|76525|986559|136152|433312|1590338|1090927|1670454|646842|304242|1364826|177909|
239778|1431533|1669866|1670408|165820|177909|239778|646842|646842|102666|1223553|1679416|337340|1669828|114606|102802|1333878|804449|
114165|103186|103289|1600659|1171911|625643|103353|303092|1091242|1348463|1484301|270857|1011577|231488|1011577|264809|1682702|302841|
338255|1472074|252516|1461068|146299|1090936|1091104|1669841|862885|934312|165820|102965|325461|1333878|1353380|1600416|1425566|165403|
177222|851724|1368621|114740|975357|165903|1261521|1488542|308951|1468285|624724|986559|103353|1091242|934312|165820|325461|102965|646842|
310526|348831|1679416|114606|1683697|273910|774159|264809|646842|862885|1669866|1461068|146299|1090936|325932|1679557|1515835|862885|
1669866|703466
25 Years of Model Checking>>231353|177407|103011|102414|1374122|1600059|1600707|1491245|103046|1669852|673721|165732|102658|515692|326626|
146373|201750|165732|146373|114592|1520208
25th Anniversary of INRIA>>102630|408994|487360|220494|1590198|1254895|1669916|134121|918422|165722|831883|329137|66232|1372471|1549221|
80359|1536200|1669852|990709|182580|189283|166007|326529|637318|1166208|229430|177685|177255|704418|1590362|1167605|106046|347596|618966|
231078|881636|1149845|166146
35 Years of Fuzzy Set Theory>>1674747|165702|1364383|764017|114610|1669772|452461|1674747|849435|795217|593784|326061|1227301|1224856|
603883|409627|494923|337476|262121|418813|984449|127533|1395363|1364267|302934|1176447|340870|1419232|531990|1491468|1348950|669571|488542|
799964|1326483|1590147
3D Research Challenges in Cultural Heritage>>562046|1357221|30070|1353625|868855|33211|210005|1162860|1396603|361439|519518|1377511|425369|
1068396|801697|1003654|167307|40961|708797|534888|1461103|35735|667941|1360209|709788|19951|210005|631060|280817
3D-GIS>>472497|326554|749142|1329081|1287864|1681917|272778|1623438|1565961|1392381|712162|1433820|1071966|852743|988290|1491893|530022|
1274653|1319484|564321|808822|712235|318667|704427|1073878|1582543|1578492|170857|1135466|815679|1256597|1525678|317635|650449|1630375|
621882|391623|341445|862745|1006122|682587|660976|1172218|916390|814181|945348|1560516|618644|687238|1685290|136497|1222518|703930|1412087|
```

Plain Text ▾ Tab Width: 8 ▾ Ln 10, Col 180 ▾ INS

1. We parsed the data for incollections, books, phd-thesis and other tags.

2. Next we constructed graph from the parsed data in which node represents authors and edges represented connections between them,edges were created if they have worked under similar topic.

To construct weighted graph we gave weight according to the area in which they worked together:

Tags	**Weight given**
article	10
inproceedings	5
proceedings	5
incollection	5
book	10
phdthesis	5
masterthesis	5

Graph obtained :

Unweighted

Weighted Graph

```
66108,690898,10
750631,1439968,10
276917,411806,10
392008,1685633,30
936946,1598424,10
303103,1163822,10
436728,1355228,10
44555,437402,5
71914,208015,10
128627,339440,10
327052,1137535,5
114783,206829,10
1439181,1524509,5
737656,1580379,5
1410306,1475832,10
326668,1225827,5
1150896,1180888,10
184130,566043,10
1279361,737685,10
1174373,1494174,5
326712,1556568,5
113379,136161,10
350754,567004,10
232214,1462326,5
1233249,1535823,10
68243,807482,10
443025,634920,10
463021,1504268,5
```

```
1290623 210282
321392 694376
110005 397050
1051038 1217824
1072890 1600508
16624 509668
598751 999757
1369714 1429624
712632 1316315
232035 338753
1684528 607799
112007 562183
1365139 1689326
3922 420058
1065795 1171059
1284664 1331843
1000645 1092117
424589 783977
750141 1279570
1140482 1365253
936946 1088190
1005260 1073225
1222719 1426335
1078286 1138089
1639575 1690963
292510 338337
668129 748611
1565772 1576801
1007872 1600494
1488403 1507207
431077 1506604
1417502 386996
1021214 1692782
1507312 781609
278338 1432005
131035 1466545
```

Algorithms

1. HIERARCHICAL CLUSTERING
   - Divisive approach-Newman Girvan
   - Newman Girvan Improvement
2. MODULARITY MAXIMIZATION
   - Louvain Method
3. VERTEX / PARTITION CLUSTERING
   - Kmeans
     - CountVectorizer
     - Tf-Idf
     - Doc2Vec
   - Label Propagation
4. EFFICIENT METHOD FOR OVERLAPPING COMMUNITIES
   - Link aggregation
   - Iterative scan

Divisive Approach

Focus on edges and vertices that exist between communities. This class tends to be more repeatable, traditional and computationally expensive

Newman Girvan (Shortest Path Betweenness)

The Girvan-Newman method for the detection and analysis of community structure is based on the iterative elimination of edges with the highest number of the shortest paths that go through them.

Edge Betweenness: is the number of shortest paths passing through the endpoints of the edge.

Vertex Betweenness: is the number of shortest paths passing through the vertex

Algorithm:

1. Calculate edge betweenness for every edge in the graph.
2. Remove the edge with highest edge betweenness
3. Calculate edge betweenness for remaining edges
4. Repeat steps 2-4 until all edges are removed

n = # of vertices , m = # of edges

Time Complexity:

$O(nm^2)$: Each Iteration uses a tree structure to calculate edge betweenness of a graph in $O(nm)$. Do this m times,once for each edge.

Output:

```
  Appelbaum,Miro Kraetzl,Yves Crama,Rahul Varshney,Marc Pirlot,Laurent Perron,David L. Olson,Chenhua Li,Masakazu Muramatsu,Martin
  Smiacuted,Stefano Gualandi,Charles E. M. Pearce,Thomas Bruckner,Jeroen Belieumln,Oleg Shcherbina,Arthur Pinkney,Soohan Ahn,Antonis
  Economou,David Yeung,Yanqing Wen,Lihua Chen,Damien Ernst,Radu Ioan Bot,Erhan Kozan,Antonio Rodrigo,Erwin von Wasielewski,John
  Kleppe,Floske Spieksma,Hsing Paul Luh,Gerhard Reinelt,Kok Lay Teo,Jing Liang,William W. Hager,C. S. Lalitha,Liqun Qi,Basil D. Manos,Nora
  Muler,Lukasz Delong,Atsuo Suzuki,Myoung-Ju Park,
2 Volker Sorge,Idriss Bengeloune,Sebastian Winkel,Christian Schulte,
3 Asad M. Ali,Jason Uher,Seraphin B. Calo,Alan T. Sherman,Kari Kostiainen,John D. Fulp,Martin Naedele,Marcus A. Maloof,
4 Huiyuan Zhang,Thang N. Dinh,
5 David Sabourin,John Baldwin,
6 Sachin Kalia,Martin Sturm,
7 Dirk Pfluumlger,Florian Echtler,Eva Geisberger,Ronald Roumlmer,Harald Goumlrl,Bartosz von Rymon-Lipinski,Ekaterina Elts,Edmond
  Kereku,Moritz Grosse-Wentrup,Sabine M. Buckl,Derik Schroumlter,Anton Riedl,Michael Tuumlchler,Polina Kondratieva,Frank Wallhoff,Robert
  Schmohl,Latifa Boursas,Stefan Reifinger,Marie Tromparent,Martin Lacher,Martin Wimmer 0001,Zheng Wang,Danail Traskov,David Bettencourt da
  Cruz,Colin Estermann,Frank Joachim Leitner,Naoufel ben Ahmed Boulila,Ioan Lucian Muntean,Jens Ernst,Christian Rehn,Daniel Stodden,Marco
  Hoffmann,Moritz G. Maaszlig,Stefan Schwaumlrzler,Sanaa Sharafeddine,Roland Haratsch,Rui Liu,Jochen Staudacher,Walid Maalej,Thomas
  Villgrattner,Florian Alexander Kuzmany,Kathrin Lehmann,Rui Chang,Florian Doumltzer,Yan Li,Christine Kiss,Sven L. Lachmund,Sascha
  Schreiber,Matthias Wimmer,Michael Robert Fahrmair,Robert Hanek,Oliver Huumlhn,Oleksandr Pochayevets,Christoph Jung,Sascha
  Kirstan,Joatildeo Barros,Stefan Hinz,Robert Muumlller,Alexandra Kirsch,Stefan Riesner,Iris Gilsdorf,Simone Kaumls,Kay Werthschulte,Martin
  Wagner,Oliver Kutter,Florian Deiszligenboumlck,Jan Robert Stadermann,Jan Bandouch,Tim Bodenmuumlller,Tina Mattes,Alexandru Berlea,Stephan
  A. Reiter,Thomas Setzer,Michael Pramateftakis,Martin Wojtczyk,Matthias Thomae,Joumlrg L. Reiner,Joumlrn David,Gerhard Muumlnz,Robert Josef
  Widhopf-Fenk,Rainer Steffen,Veronika Thurner,Wolfgang Woumlrndl,Michael Kleis,Tjark Weber,Martin Schwaiger,Jens Harald Kruumlger,
8 Martin Vogt,Dennis J. Underwood,Dominik Gront,Kun Zou,F. J. V. Pinto,Marta Murcia,Hao Wang,Marcel L. Verdonk,Gerhard Bringmann,Richard
  Lewis,Sonja Meddeb,T. Lehmann,Oliver Barker,Fredrik Bjoumlrkling,Marcus Elstner,Shantaram Kamath,Ronan Bureau,Ritu Aneja,
9 Roque Alfredo Osornio-Rios,Jesus Rooney Rivera-Guillen,
10 Shinn-Horng Chen,Ali Vahidian Kamyad,Peng Jia,Nedia Aouani,Zhaopeng Ding,Paul Messenger,
11 Ki-Joune Li,Davide Buscaldi,Monika Sester,
12 Jan T. Fischer,Holger Gast,
13 Markus Won,Thorsten Belker,Pascal Costanza,
14 David Kreische,Klaus Donath,Christian Langenbach,Christoph Guumltter,
15 Bora Beran,Michael Piasecki,
16 Daniel Warneke,Michael Stemmer,Mohammad Shadi Al Hakeem,Dirk Kleeblatt,Tobias Achterberg,Alexander Loumlser,Kerstin Buhr,Sandro
  Leuchter,Stephan Frank,Roland Stahn,Aureli Soria-Frisch,Tanja Zseby,Nabil Aly Mohamed Aly Lashin,Bernd-Paul Simon,Gabriele Beate
  Schweikert,Thomas Hoch,Martin Grabmuumlller,Josef Maier,Ralph A. Muumlller,Alain-Georges Vouffo Feudjio,Jan Trowitzsch,Christian
  Petersohn,Jae-In Lee,Soumlren Sonnenburg,Matthias Fluumlgge,
17 Martin Becker 0002,Stephan Baumann,Bernd Loumlchner,Manuel Moumlller,Andreas Jedlitschka,Alexander Geraldy,Gustavo Nery,
18 Pathamadi V. Sankar,A. A. Naqvi,Anna R. Bruss,
19 Anna Astapenko,Mesfin Mulugeta Dinku
```

Modularity

Modularity is the fraction of the edges that fall within the given groups minus the expected fraction if edges were distributed at random

normalization

adjacency matrix

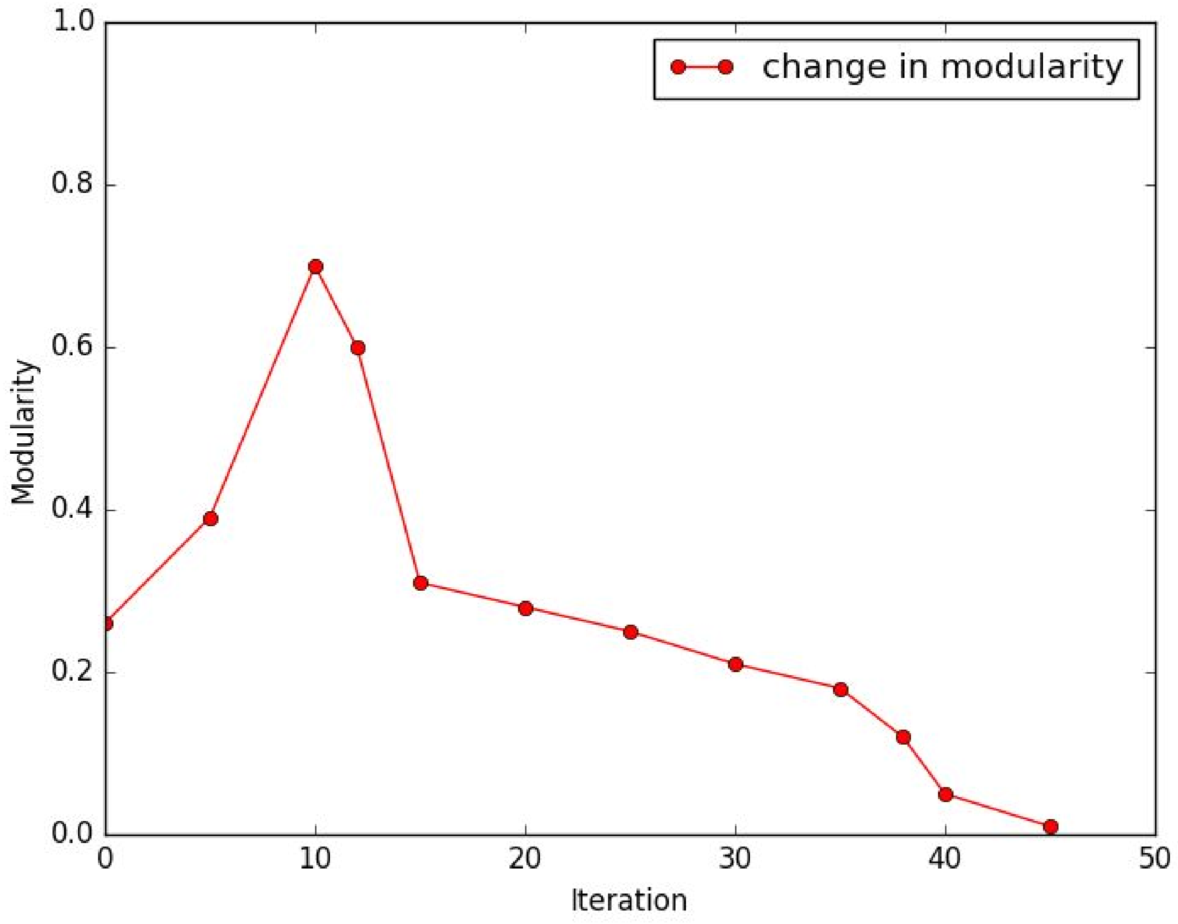
probability a random edge would go between i and j

$$Q = \frac{1}{4m} \sum_{\substack{i,j \\ \text{in same} \\ \text{module}}} \left(A_{ij} - \frac{k_i k_j}{2m} \right)$$

m = # edges in graph
k_i = degree(i)

Consider the case of only 2 modules.
Let $s_i = 1$ if node i is in module 1; -1 if node i is in module 2

$$\begin{aligned} Q &= \frac{1}{4m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) (s_i s_j + 1) \\ &= \frac{1}{4m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) s_i s_j \end{aligned}$$

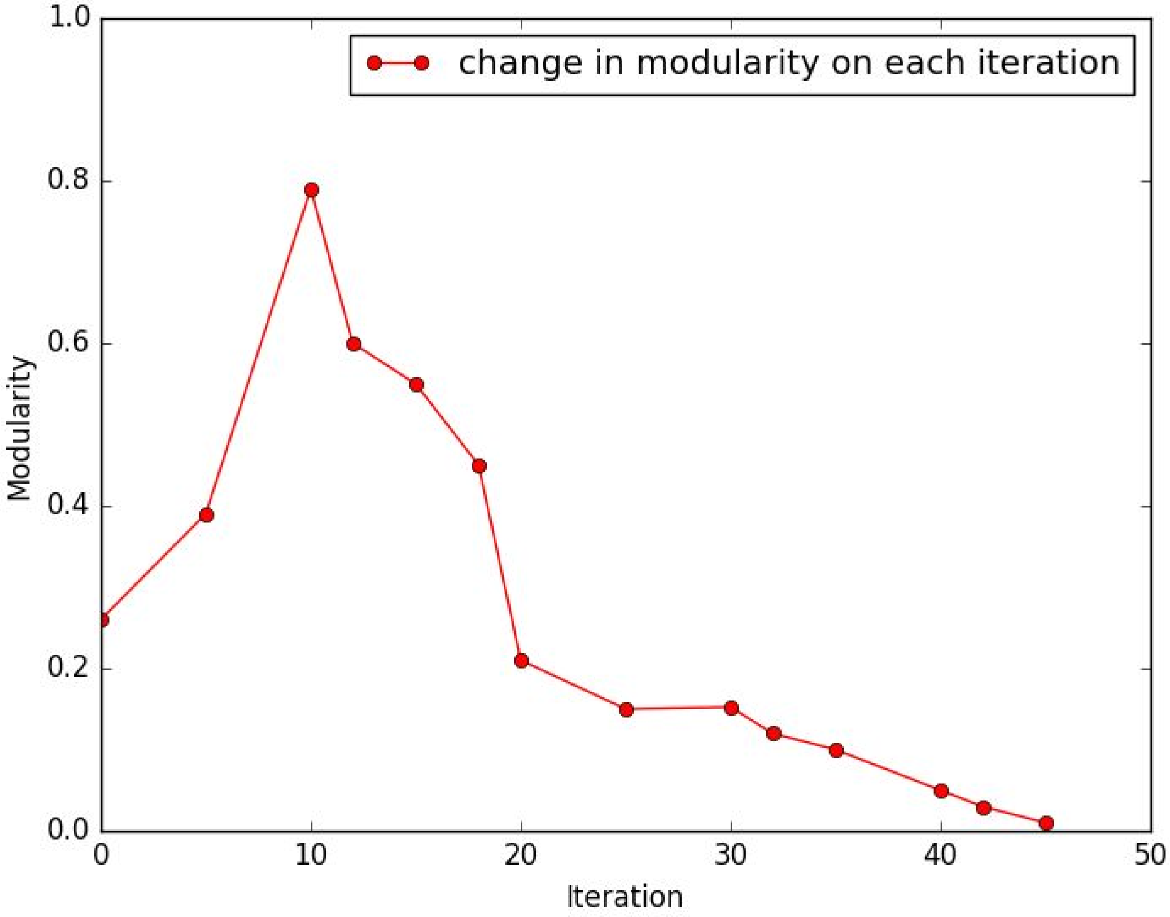

1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
Modularity
Iteration
change in modularity

Newman Girvan Improvement

1. Calculate edge betweenness for every edge in the graph.
2. Remove **all edges** with highest edge betweenness.
3. Recalculate edge betweenness for remaining edges.
4. Repeat 2-4 until graph becames empty.

Time Complexity:

Worst-case time complexity is still $\mathbf{O(nm^2)}$, but in networks with strong community structure the number of calculations could be significantly reduced.


change in modularity on each iteration
Modularity
Iteration
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
50

Results

S.No.	Algorithms	Modularity	Execution Time
1.	Newman Girvan	0.71	300 min 12 sec(on 5000 edges)
2.	Improvement in Newman Girvan	0.78	102 min 45 sec(on 5000 edges)

MODULARITY OPTIMIZATION

Louvain Method

The method consists of two phases.

1.It looks for "small" communities by optimizing modularity in a local way.

2.It aggregates nodes of the same community and builds a new network whose nodes are the communities.

These steps are repeated iteratively until a maximum of modularity is attained.

The output of the program therefore gives several partitions. The partition found after the first step typically consists of many communities of small sizes. At subsequent steps, larger and larger communities are found due to the aggregation mechanism.

Modularity:

$$Q = \sum_{c \in C} \left[\frac{\Sigma_{in}^{c}}{2m} - \frac{(\Sigma_{tot}^{c})^2}{4m^2} \right],$$

where
Σc_{in} is the sum of the weights from all internal edges of community c, calculated as $\sum w(u,v), \forall u,v \in c$ and $e(u,v) \in E$,$\Sigma c,tot$ is the sum of the weights from edges incident to any vertex in Community c, calculated as $\sum w(u,v), \forall u \in c$ or $v \in c$ and $e(u,v) \in E$, and m is the normalization factor obtained by summing the edge weights across the entire graph.

Gain in modularity:

$$\begin{aligned}
\Delta Q_{u \to c} &= \left[\frac{\Sigma_{in}^{c} + w_{u \to c}}{2m} - \left(\frac{\Sigma_{tot}^{c} + w(u)}{2m} \right)^2 \right] \\
&\quad - \left[\frac{\Sigma_{in}^{c}}{2m} - \left(\frac{\Sigma_{tot}^{c}}{2m} \right)^2 - \left(\frac{w(u)}{2m} \right)^2 \right] \\
&= \frac{w_{u \to c}}{2m} - \frac{\Sigma_{tot}^{c} * w(u)}{2m^2}
\end{aligned}$$

$w(u)$ is the sum of the weights of the edges incident to vertex u, and $w_{u \to c} = \sum_{v \in c} w_{u,v}$ is the sum of the weights of the edges from vertex u to vertices in community c

LOUVAIN ALGORITHM

Input: $G=(V,E)$: graph representation.
Output: C: community sets at each level;
Q: modularity at each level.
Var: $\hat{c}$: vertex u's best candidate community set.
Loop ***outer***
- $C \leftarrow \{\{u\}\}, \forall u \in V$;
- $\Sigma_{in}^{c} \leftarrow \sum w_{u,v},\ e(u,v) \in E,\ u \in c$ and $v \in c$;
- $\Sigma_{tot}^{c} \leftarrow \sum w_{u,v},\ e(u,v) \in E,\ u \in c$ or $v \in c$;
- *// Phase 1.*
- **Loop** ***inner***
  - **for** $u \in V$ *and* $u \in c$ **do**
    - *// Find the best community for vertex u.*
    - $\hat{c} \leftarrow \underset{\forall c',\ \exists e(u,v) \in E,\ v \in c'}{\operatorname{argmax}} \Delta Q_{u \rightarrow c'}$;
    - **if** $\Delta Q_{u \rightarrow \hat{c}} > 0$ **then**
      - *// Update Σ_{tot} and Σ_{in}.*
      - $\Sigma_{tot}^{\hat{c}} \leftarrow \Sigma_{tot}^{\hat{c}} + w(u)$; $\Sigma_{in}^{\hat{c}} \leftarrow \Sigma_{in}^{\hat{c}} + w_{u \rightarrow \hat{c}}$;
      - $\Sigma_{tot}^{c} \leftarrow \Sigma_{tot}^{c} - w(u)$; $\Sigma_{in}^{c} \leftarrow \Sigma_{in}^{c} - w_{u \rightarrow c}$;
      - *// Update the community information.*
      - $\hat{c} \leftarrow \hat{c} \cup \{u\}$; $c \leftarrow c - \{u\}$;
  - **if** *No vertex moves to a new community* **then**
    - **exit** ***inner*** **Loop**;
- *// Calculate community set and modularity.*
- $Q \leftarrow 0$;
- **for** $c \in C$ **do**
  - $Q \leftarrow Q + \frac{\Sigma_{in}^{c}}{2m} - (\frac{\Sigma_{tot}^{c}}{2m})^2$;
- $C' \leftarrow \{c\}, \forall c \in C$; print C' and Q ;
- *// Phase 2: Rebuild Graph.*
- $V' \leftarrow C'$;
- $E' \leftarrow \{e(c,c')\},\ \exists e(u,v) \in E,\ u \in c,\ v \in c'$;
- $w_{c,c'} \leftarrow \sum w_{u,v},\ \forall e(u,v) \in E,\ u \in c,\ v \in c'$;
- **if** *No community changes* **then**
  - **exit** ***outer*** **Loop**;
- $V \leftarrow V'$; $E \leftarrow E'$;

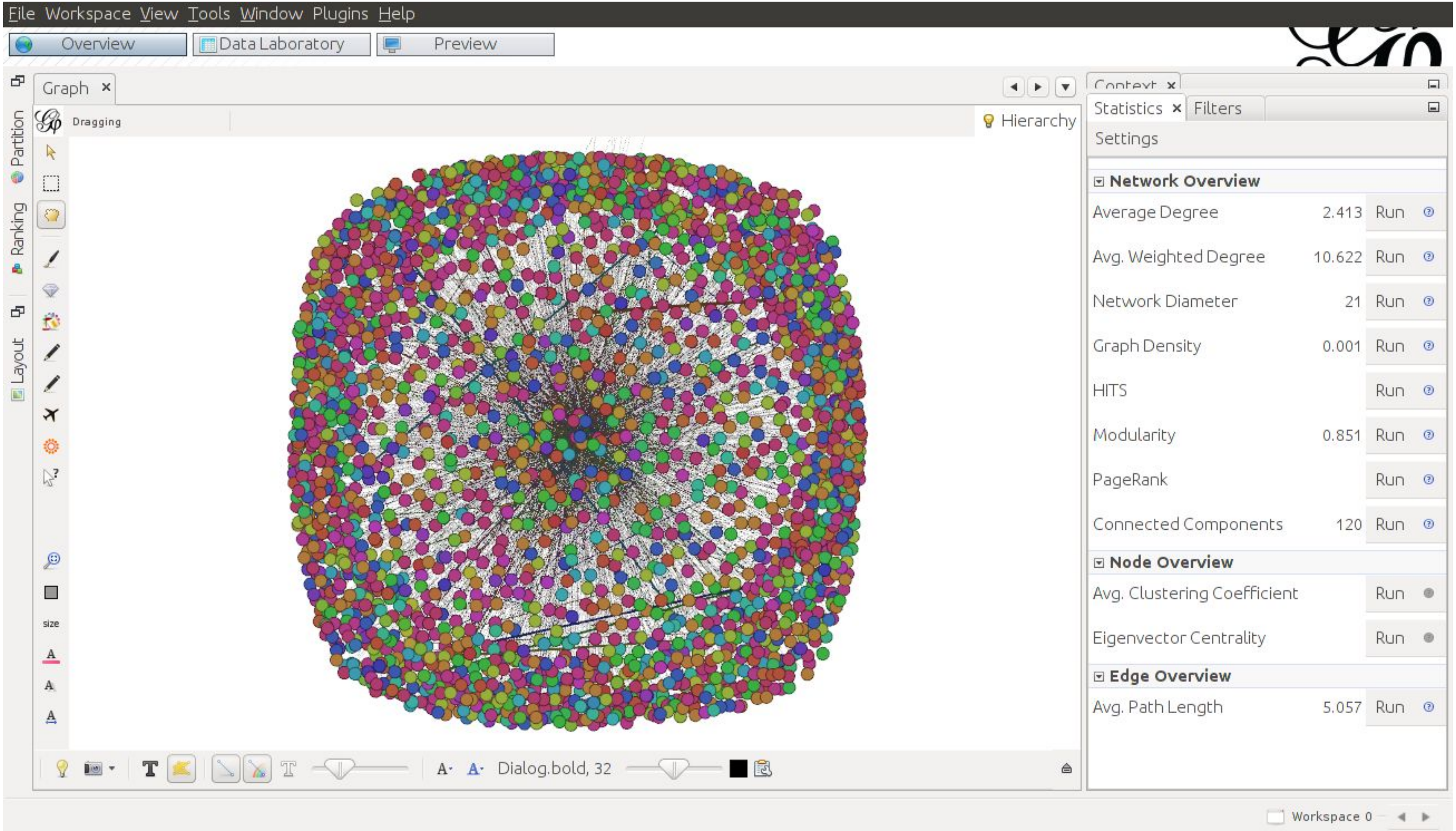
Castro, Kwanho Kim, Vincent Melfi, Gillian R. Hayes, Zhexue Huang, Alexey Tsymbal, An-Lei Hu, Qiyao Wang, Anupam Basu, Lauren Katzen, Adrian Demaid, Ernesto Compatangelo, Dewang Chen, J. Dong, Kun Yue, Yaojin Lin, Roderic Leigh, Bert Bongers, Kuo-Yuan Kao, Amy K. Hurst, Hannes Perkmann, Hongjun Lu, Hou-Yi Li, Yasutoshi Makino,

Joachim Lepping, Claus Pahl, Ralf Stadelhofer, Jiadao Li, Lena Wiese, Stefan Haustein, Timm Euler, Paul Lokuciejewski, Heike Hunneshagen, Heiko Falk, Nanette Bauer, Karsten Klein, Martin Lang 0005, Haiseung Yoo, Thorsten Camps, Frank Weichert, Claudia Reuter, Dirk D uuml;ding, Maria Kuhl, Morteza Monemizadeh, Baiyi Song, Uta Pankoke-Babatz, Carsten Witt, Thomas K ouml;nigsmann, Matthias Hebbel, Alexander Klemm, Gerrit Bleumer, Tahir Ejaz, Ingo L uuml;ck, Konstantinos Kotsokalis, Stephan Lehmke, Gerrit Rothmaier, Jens Niehaus, Kamol Limtanyakul, Christian Thyssen, Claudia Gsottberger, Christoph Jan Richter, Aleksandra Sowa, Martin Scholz, Harald Gebhard, Peter Schramm, Tobias Marschall, Carsten Gutwenger, David Fiedler, Mohammad Yasser al-Nahlaoui, Stefan Schmermbeck, Klaus Julisch, Patrick Ren eacute; Steve Piastowski, G uuml;nter Graw, Georgios Lajios, Ingo Dahm, Torben Weibert, Jens Busch, Mark Jung, Boris Naujoks, D ouml;rte K. Rappe, Gila Brandt-Herrmann, Maria Kandyba-Chimani, Xiaolei Shi, Roman Klinger, Dominik G ouml;ddeke, J ouml;rg Pleumann, Daniel Chernuchin, J uuml;rgen Kemper, Thomas Beielstein, Christian Brockmann, Marius Otte, Jens Wagner, Katharina Hilker,

Toni Cortes, Mikael H ouml;gqvist,
Maria Sorea, Zhendong Ma, Pierre Bayerl,

Heiko Maus, Joachim Bayer, Tobias Schuele, Oliver Wirjadi, Peter Zeile, Jens Brandt, Sven Schwarz, Michael P. Haustein, Gerrit Hanselmann, Ingmar Fliege, Ramon Serna Oliver, Achim Ebert, Mark M uuml;ller, Ivan Martinovic, Thomas Patzke, Martin Becker 0002, Jan Olaf Blech, Georg Buscher, Klaudia Hergula, Joachim Thees, Marcus Trapp, Philipp Dopichaj, Jens Heidrich, Andreas M. Weiner, Mart iacute;n Soto, Ingo Ginkel, Armin Stahl, R uuml;diger Ebendt, Sebastian Thelen, Manuel M ouml;ller, Andreas Jedlitschka, Mesut Ipek, Stephan Baumann, Raik Brinkmann, Frank Michel, Ulrike Becker-Kornstaedt, Chenxi Qiu, Markus Nick, Matthias Gro szlig;, Gabriele Bleser, Thomas Kilb, Patric Keller, Dirk Hamann, Christoph K ouml;gl, Joost van Beusekom, Akin Tanatmis, Ina Schaefer, Mario Trapp, Younis O. Hijazi, Christoph Garth, Erwin Sitompul, Kizito Ssamula Mukasa, Christian Mathis, Mohammed Bani Younis, Ansgar Lamersdorf, Thomas Kollig, Tanvir M. E. Hussain, R uuml;diger Grammes, Gustavo Nery, Jochen M uuml;ller 0002, Jean-Francois Girard, Alexander Geraldy, Bernd G. Freimut, Helge Sch auml;fer, Jorge Rafael Vel aacute;squez Flores, Tobias Schmidt-Samoa, Michael M uuml;nchhofen, Jochem H uuml;llen, Holger Diekmann, Jan Schaefer, Achim Reuther, Max Thalmaier, Hao Jiang, Sandra Zilles, Florian Gerhardt, Robert Kolter, Maja Ruby, Jens Knodel, Sascha H. Schmitt, Eduard Deines, Tim Braun, Daniel Burkhart, Leonardo Ribeiro, Stefan Agne, Daniel G ouml;rlich, Torsten Bierz, Matthias Priebe, Christian Webel, Leo Sauermann, Eros Comunello, Lars Geyer, Thorsten Keuler, Siana Halim, Norbert Schmitz, Armin Hust, Robert Eschbach, Eric Ras, Ge Zhang, Philipp Schaible, Christian Denger, Thomas Kuhn, Markus Bon, Adrian Ulges, Heinz Ulbricht, Petra Malik, Norbert G ouml;b, Alexis Ocampo, Gerrit Meixner, Jernej Kov eacute;se, Steffen Wolf, Robert Kalckl ouml;sch, Jos eacute; de Agiuar Moraes Filho, Markus Hillenbrand, Andreas Morgenstern, Bernd L ouml;chner, Benedikte Elbel, Oliver R uuml;bel, Duoli Qiu, Bernd Reuther, Isabel John, Michael Schlemmer, Harald Holz,

Tanja Paulitz, Roland Steidle,


File Workspace View Tools Window Plugins Help
Overview
Data Laboratory
Preview
Graph
Dragging
Hierarchy
Partition
Ranking
Layout
Context
Statistics
Filters
Settings
Network Overview
Average Degree 2.413 Run
Avg. Weighted Degree 10.622 Run
Network Diameter 21 Run
Graph Density 0.001 Run
HITS Run
Modularity 0.851 Run
PageRank Run
Connected Components 120 Run
Node Overview
Avg. Clustering Coefficient Run
Eigenvector Centrality Run
Edge Overview
Avg. Path Length 5.057 Run
size
Dialog.bold, 32
Workspace 0

Results

S. No.	Algorithm	Modularity	Execution Time (sec)
1.	Newman Girvan	0.71	300 min 12 sec(on 5000 edges)
2.	Improvement in Newman Girvan	0.78	102 min 42 sec(on 5000 edges)
3.	Louvain Method	0.851	15 min 32 sec(on 5000 edges)

Vertex Clustering

- Creates a clusters based on the value of a vertex attribute.
- Vertices having the same attribute will correspond to the same cluster.
- Embeds the Graph into vector space in order to use conventional data clustering methods such as k-means

Kmeans

1. The KMeans algorithm clusters data by trying to separate samples in n groups of equal variance.
2. It requires the number of clusters to be specified.
3. It scales well to large number of samples.
4. Kmeans trying to minimize the distance from the points to the cluster.

Kmeans Clustering trying to solve:

$$\arg\min_{\mathbf{c}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in c_i} d(\mathbf{x}, \mu_i) = \arg\min_{\mathbf{c}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in c_i} \|\mathbf{x} - \mu_i\|_2^2$$

Kmeans Algorithm

1. Initialize the center of the clusters	$\mu_i =$ some value $, i = 1, \ldots, k$
2. Attribute the closest cluster to each data point	$\mathbf{c}_i = \{j : d(\mathbf{x}_j, \mu_i) \leq d(\mathbf{x}_j, \mu_l), l \neq i, j = 1, \ldots, n\}$
3. Set the position of each cluster to the mean of all data points belonging to that cluster	$\mu_i = \frac{1}{\|c_i\|} \sum_{j \in c_i} \mathbf{x}_j, \forall i$
4. Repeat steps 2-3 until convergence	
Notation	$\|\mathbf{c}\| =$ number of elements in $\mathbf{c}$

Silhouette Analysis of Kmeans Clustering

1. The Silhouette Coefficient is calculated using the mean intra-cluster distance (a) and the mean nearest-cluster distance (b) for each sample.
2. The best value is 1 and the worst value is -1.
3. Values near 0 indicate overlapping clusters.
4. Negative values generally indicate that a sample has been assigned to the wrong cluster, as a different cluster is more similar.

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

Which can be also written as:

$$s(i) = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i) \\ 0, & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i) \end{cases}$$

Algorithms

1. Count Vectorizer + Kmeans
2. TF-IDF + Kmeans
3. Doc2Vec + Kmeans

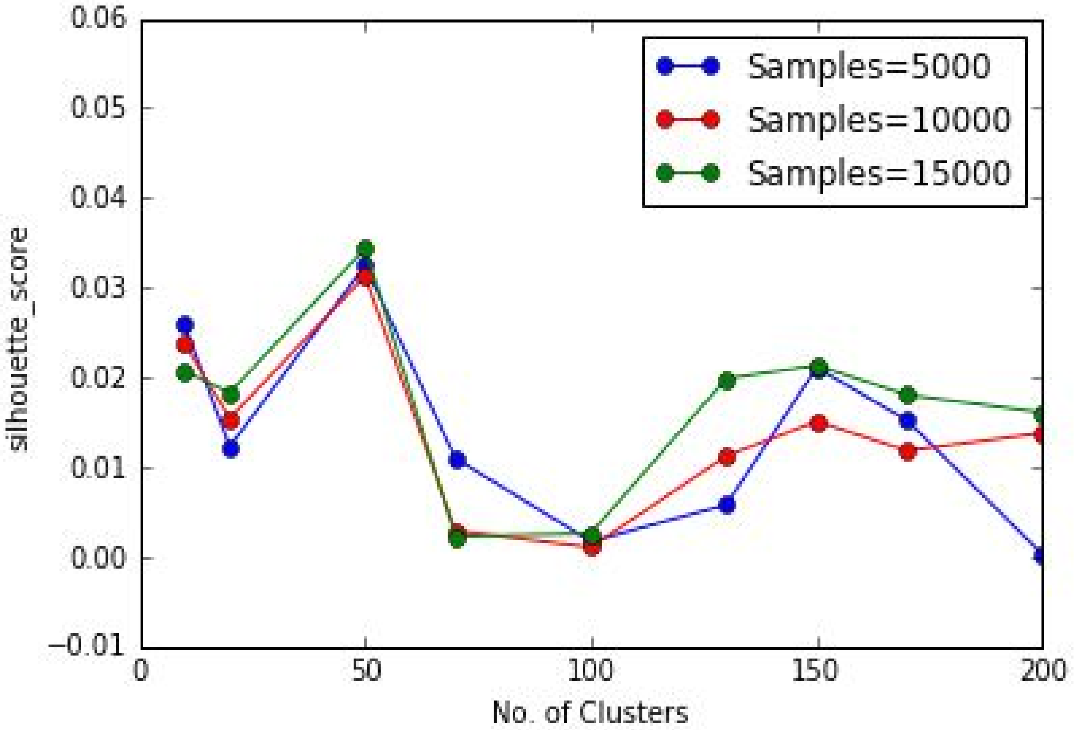
Input Vector based on CountVectorizer

1. Convert a collection of Articles to a matrix of token counts
2. Implementation produces a sparse representation of the counts
3. If you do not provide an a-priori dictionary and you do not use an analyzer that does some kind of feature selection then the number of features will be equal to the vocabulary size found by analyzing the data

Results-CountVectorizer

Cluster Size	Sample Size (Best Sample size)	Silhouette_Score
10	5000	0.025877574447
20	15000	0.0182702724563
50	15000	0.0343891621179
70	5000	0.0109729691681
100	15000	0.00263053653512
130	15000	0.0198805833876
150	15000	0.0213326772462
170	15000	0.0180481606536
200	15000	0.0161160082673

Results-CountVectorizer



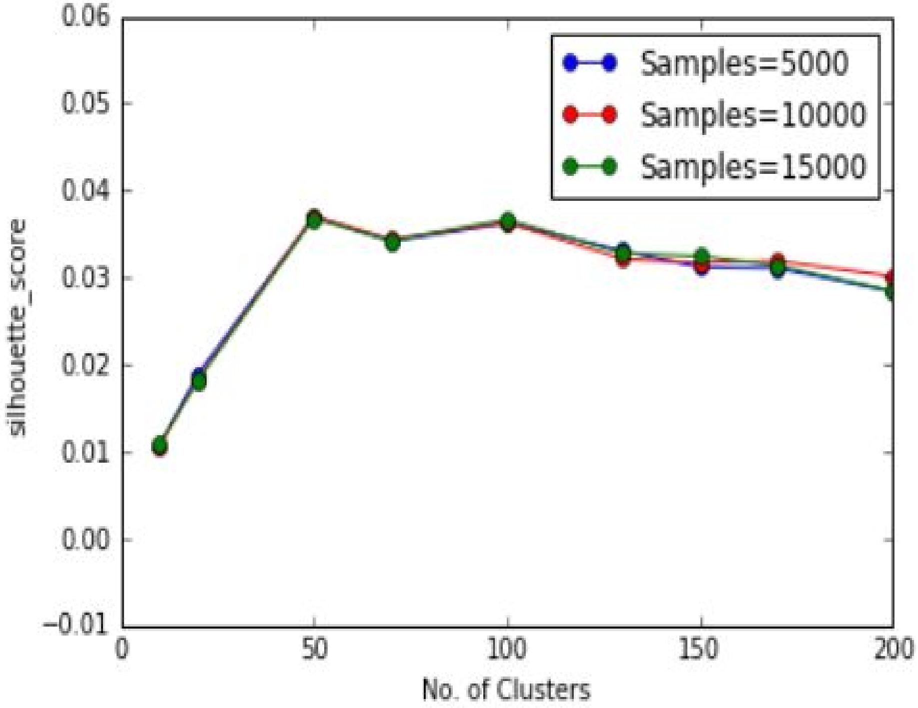
Input Vector based on TF-IDF

1. This method as input takes in a set of word Articles and outputs a vector with integer components.
2. It identifies the vocabulary of the entire set of Articles that are fed as input to it. The dimension of the vector is equal to the size of the vocabulary. (It can be reduced to the number of our convenience.)
3. In each article, the unique words that are present are identified, and in the vector, the component values indicate the frequency of their occurrence in that article.

Results-TF-IDF

Cluster Size	Sample Size (Best Sample size)	Silhouette_Score
10	5000	0.010734032515
20	5000	0.0189018479368
50	10000	0.0370563180451
70	10000	0.0343865015456
100	15000	0.0366813813101
130	5000	0.0330725535286
150	15000	0.0324619651998
170	10000	0.0319264441636
200	10000	0.0301362632732

Result-TF-IDF



Input Vector based on DOC2VEC

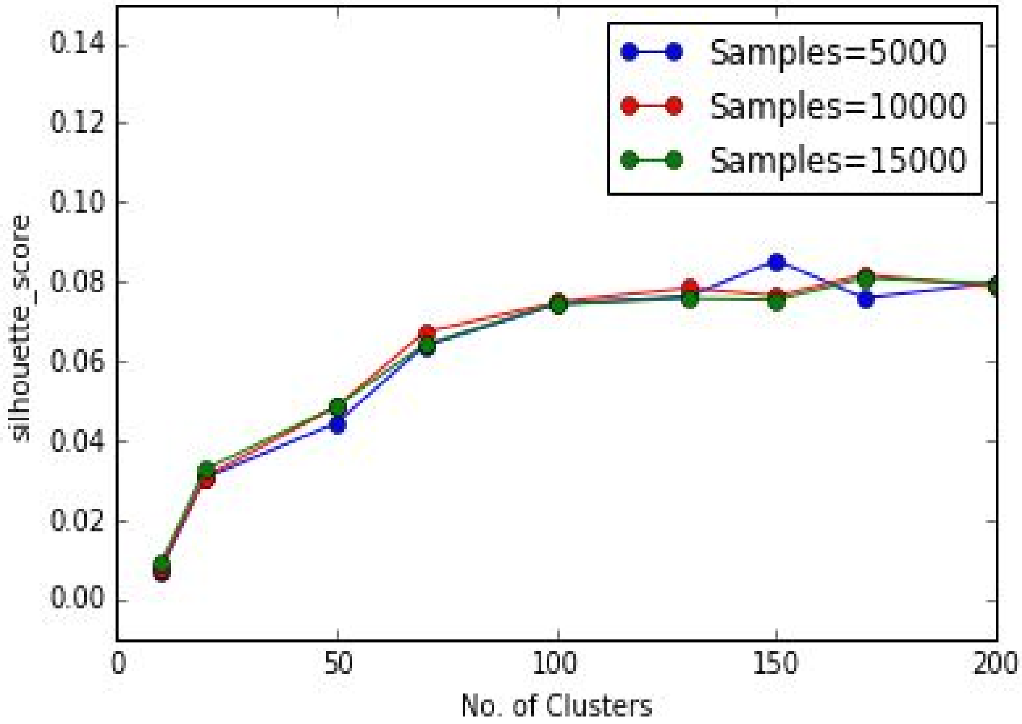
1. We provided all the articles as inputs to the gensim Doc2Vec Model.
2. We obtained document embeddings for each article we used as input.
3. Using the document embedding, we could represent each input instance a vector with real numbers.

Doc2vec runs through the sentences iterator twice: once to build the vocab, and once to train the model on the input data, **learning a vector representation for each word and for each label** in the dataset

Results-Doc2Vec

Cluster Size	Sample Size (Best Sample size)	Silhouette_Score
10	15000	0.00904671
20	15000	0.0327203
50	15000	0.0486096
70	10000	0.0672753
100	10000	0.074809
130	10000	0.0784348
150	5000	0.0853631
170	10000	0.081752
200	15000	0.0797882

Results-Doc2Vec



Results/Comparisions

S.NO	InputVector/ Algorithms	Sample Size	Optimal Cluster size	Silhouette Score
1.	CountVectorizer + KMeans	15000	50	0.0343891
2.	TF-IDF + KMeans	15000	50	0.0370563
3.	Doc2Vec + KMeans	15000	150	0.0853631

Label Propagation Algorithm

The idea of a basic label propagation algorithm (LPA) is very simple, that is, letting each node in the same community as most of its neighbors. The specific algorithm flow is: initialize, each node carries a unique label; then update the label of the node so that its label is the same as the label of most of its neighbors, if there are multiple random selection. Iteration until the label of each node no longer changes.

This requires an overall time of O(m + n) time.
#m no of edges
#n no of nodes

Label Propagation

The process has 5 steps:

1. Initialize the labels at all nodes in the network. For a given node x, C_x (0) = x.

2. Set t = 1.

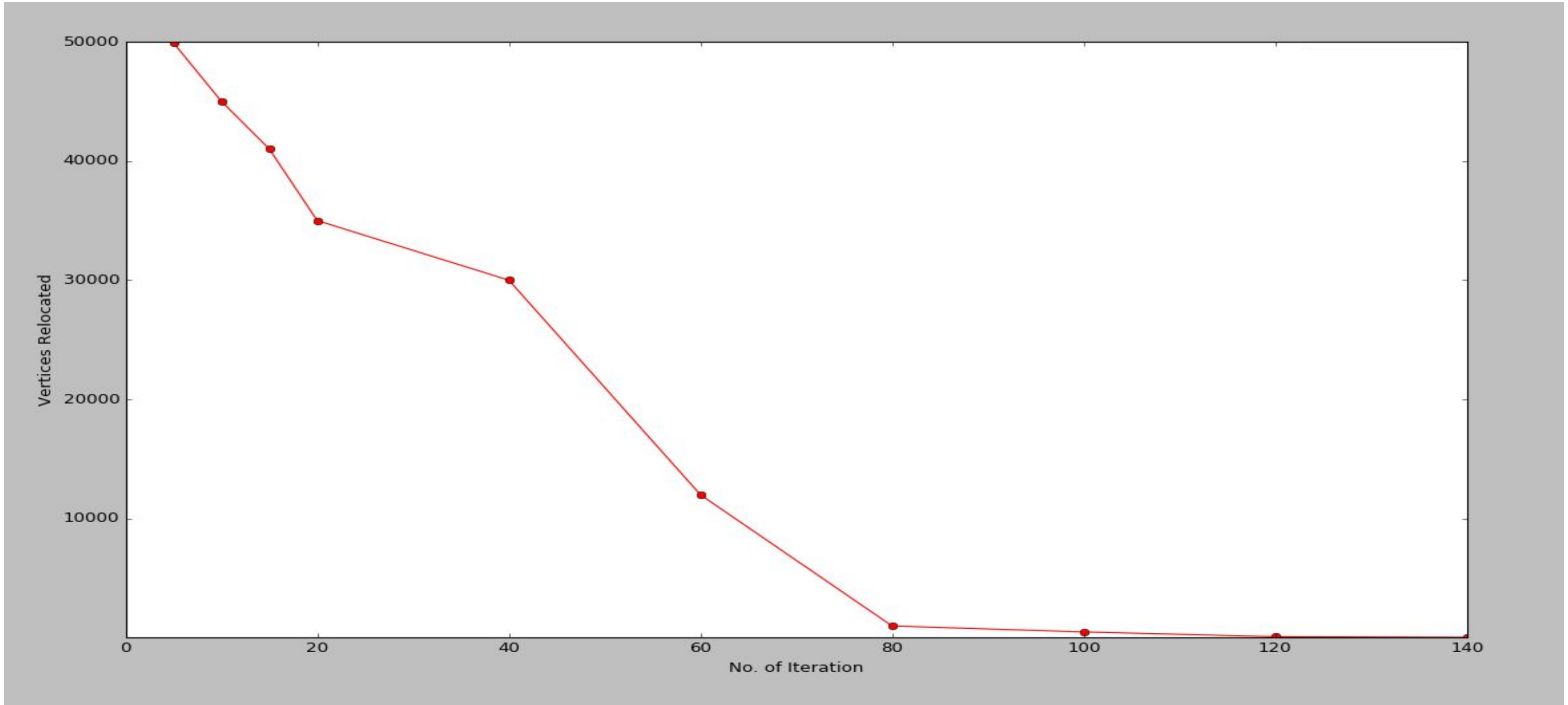
3. Arrange the nodes in the network in a random order and set it to X.

4. For each $x \in X$ chosen in that specific order, let $C_x(t) = f(C_{xi1}(t - 1), ..., C_{xim}(t - 1), C_{xi(m+1)}(t - 1), ..., C_{xik}(t - 1))$. f here returns the label occurring with the highest frequency among neighbours.

5. If every node has a label that the maximum number of their neighbours have, then stop the algorithm. Else, set t = t + 1 and go to (3).

Results

Label Propagation Graph of Iteration vs Vertices Relocated



Results:

Time - 2 minutes , Data-dblp.dtd , Iterations-20

```
[1005416, 1470953]
[1167790, 385864]
[851830, 1674817]
[39719, 37464]
[1671487, 443029, 742960]
[1449576, 1446704]
[5859, 668755]
[49463, 1397211]
[1077125, 491044]
[143955, 140200]
[559837, 1518480]
[338791, 647787]
[292770, 1170738]
[1328284, 1670070]
[531397, 478088]
[170700, 1095886]
[849237, 979272]
[839675, 845076]
[1139259, 1232014]
[381529, 443425]
[1301512, 1304425]
[1218283, 1684658]
[71484, 188386]
[1595952, 1639148]
```

Efficient Approach to find Overlapping Communities

Overlapping Subgraph Algorithm
(Link Aggregate + Iterative Scan)

The algorithm for finding locally optimal subgraphs,consists of two parts:

1. **Initialization:** LA, which creates seed clusters

2. **Improvement**: IS, which repeatedly scans the vertices in order to improve the current clusters until one arrives at a locally optimal collection of clusters.

Page Rank

1. PageRank computes a ranking of the nodes in the graph G based on the structure of the incoming links.
2. The PageRank algorithm was designed for directed graphs but this algorithm does not check if the input graph is directed and will execute on undirected graphs by converting each edge in the directed graph to two edges.

$$\phi_p(v) = c\sum_{u,v} \frac{\phi_p(v)}{\deg^-(v)} + \frac{1-c}{n}$$

Where, n is the no. of nodes in a graph ; $\deg^-(v)$ is the out degree of a vertex v ; c is the decay factor ; $\phi_p(v)$ is the Page rank of a vertex v.

Cluster Density Metric

Cluster density tells how intense the relation within the cluster nodes is, relative to that outside the cluster nodes.

For undirected simple graphs, the **graph density** is defined as:

$$D = \frac{2|E|}{|V|\,(|V|-1)}$$

E is the number of edges and V is the number of vertices in the graph. The maximum number of edges for an undirected graph is ½ $|V|\,(|V|-1)$, so the maximal density is 1 and the minimal density is 0

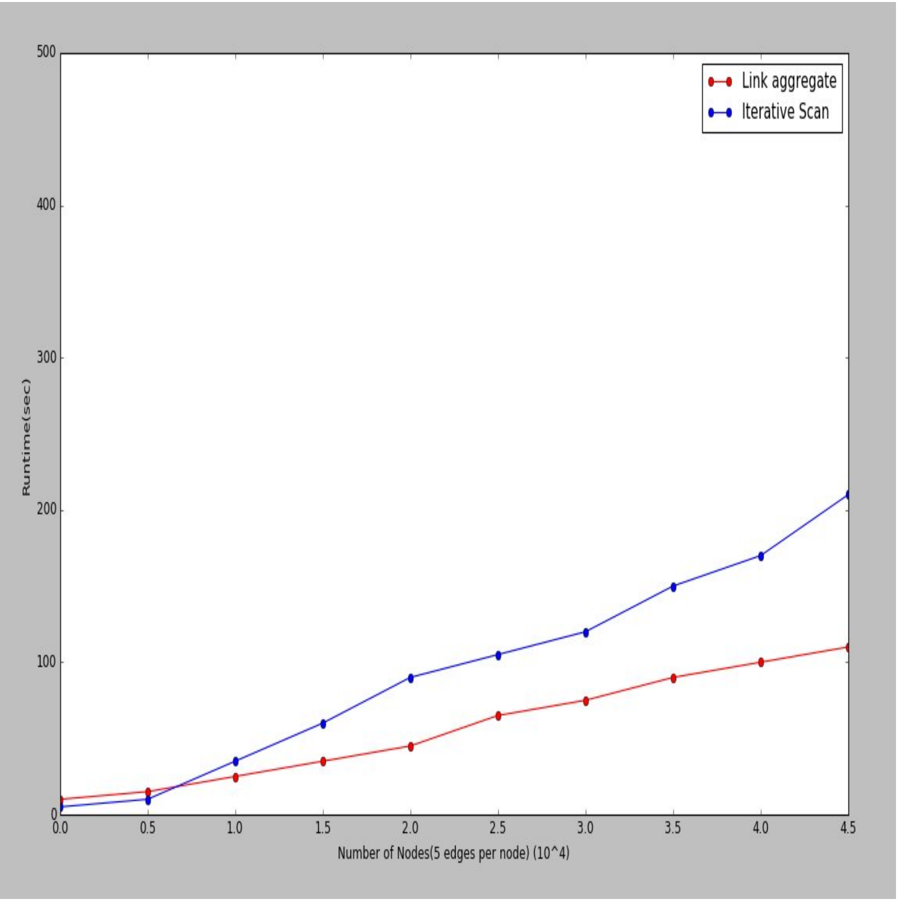
procedure $\mathsf{LA}(G = (V, E), W)$
$C \leftarrow \emptyset$;
Order the vertices $v_1, v_2, \ldots, v_{|V|}$;
for $i = 1$ to $|V|$ **do**
  $added \leftarrow$ false;
  for all $D_j \in C$ **do**
    if $W(D_j \cup v_i) > W(D_j)$ **then**
      $D_j \leftarrow D_j \cup v_i$; $added \leftarrow$ true;
  if $added =$ false **then**
    $C \leftarrow C \cup \{\{v_i\}\}$;
return C;

LINK AGGREGATE

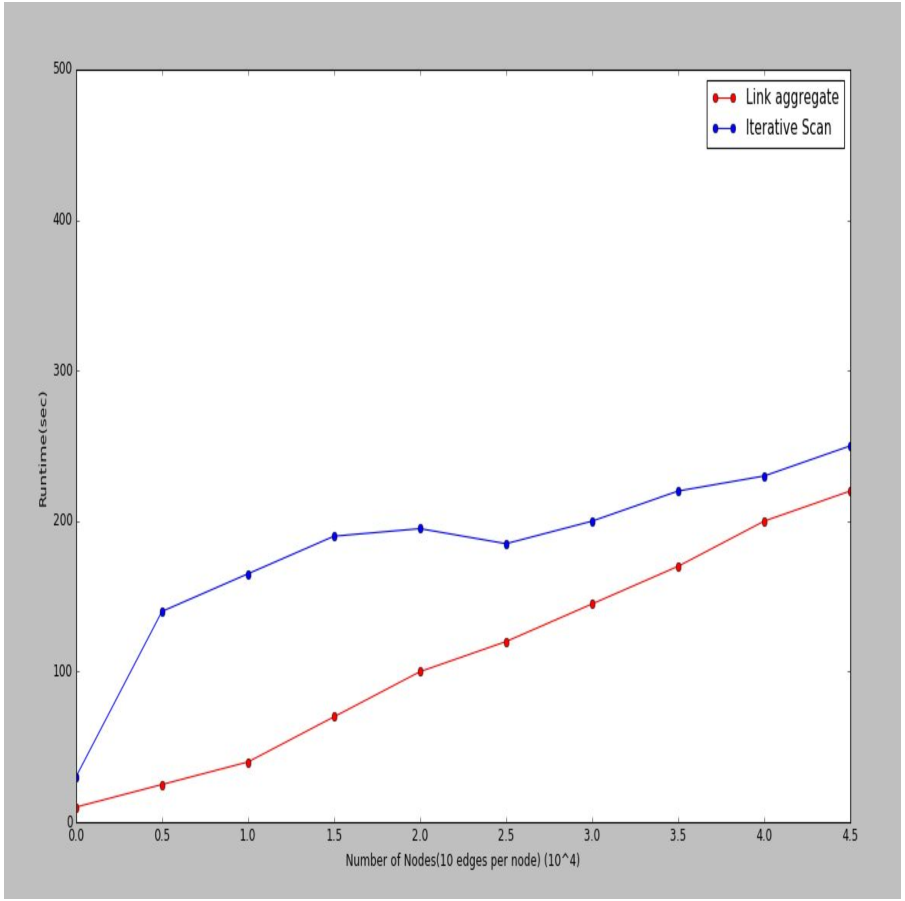
Iterative Scan

```
procedure IS²(seed, G, W)
C ← seed; w ← W(C);
increased ← true;
while increased do
    N ← C;
    for all v ∈ C do
        N ← N ∪ adj(v);
    for all v ∈ N do
        if v ∈ C then
            C' ← C \ {v};
        else
            C' ← C ∪ {v};
        if W(C') > W(C) then
            C ← C';
    if W(C) = w then
        increased ← false;
    else
        w ← W(C);
return C;
```

Results

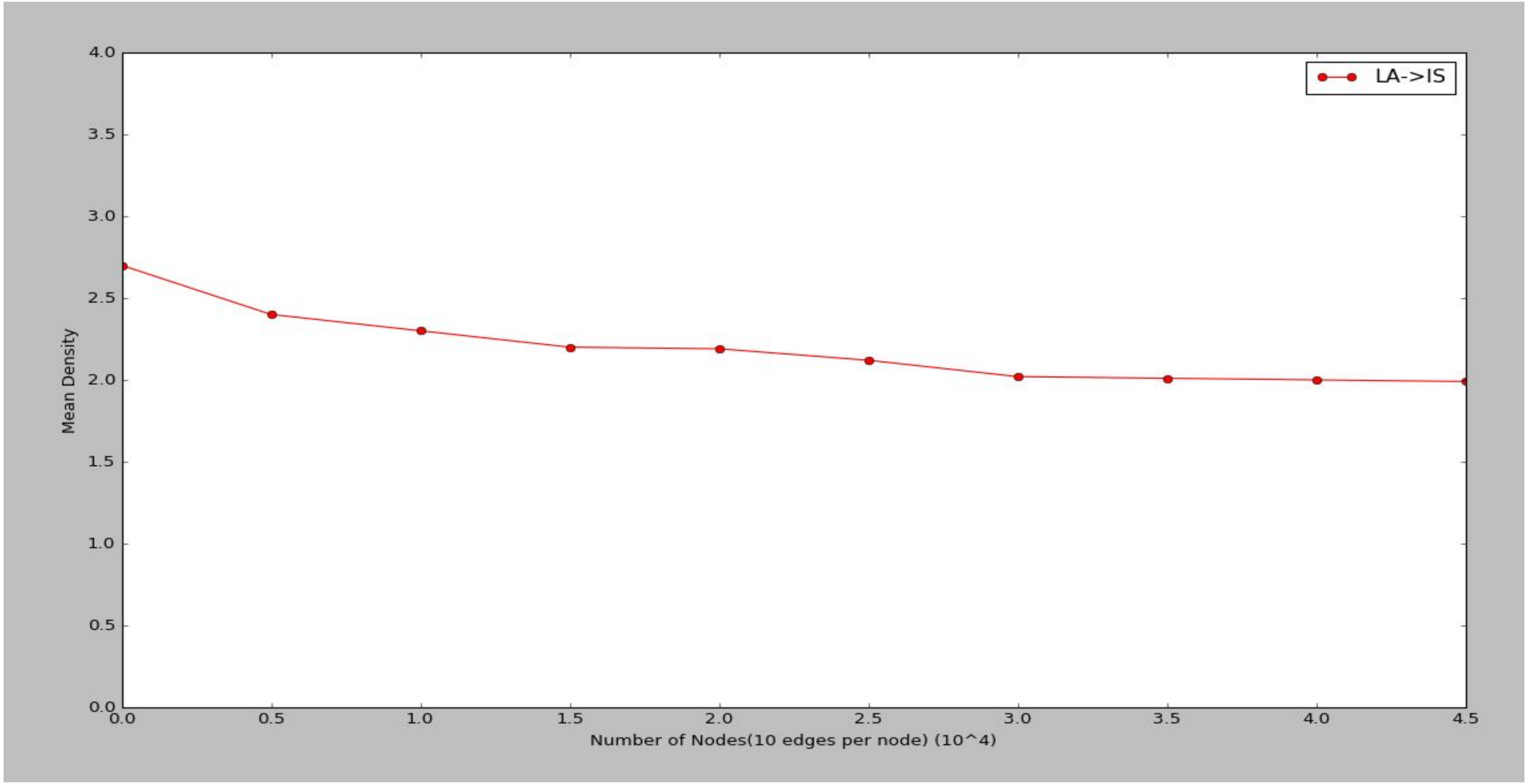


Link aggregate
Iterative Scan
Runtime(sec)
Number of Nodes(5 edges per node) (10^4)
0
100
200
300
400
500
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5



Link aggregate
Iterative Scan
Runtime(sec)
Number of Nodes(10 edges per node) (10^4)
0
100
200
300
400
500
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5

Results



LA->IS
Mean Density
Number of Nodes(10 edges per node) (10^4)
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5

Conclusion

Algorithm	Time Complexity
Newman Girvan	O(VE2)
Improvement in Newman Girvan	O(VE2) (reduce number of computation)
Louvain	O(V log V)
Kmeans	O(CV)
LA->IS	O(|C||E|+|V|)
LP	O(|V|+|E|)

Where E #No. of Edges
V #No. Vertices
C #No. of Clusters
N #No .of Nodes

Newman Girvan and Louvain are based on modularity so if we want to get more modular graph then we may stick to these algorithm out of which Louvain is better as it is more efficient.

KMeans and Label propagation are having different aspect KMeans involves unsupervised learning , we need to train and test the data. So for large network if data is dynamic then this won’t help much but if the data is static then clusters obtained is very accurate.

The advantage of the LPA algorithm is simple, fast approaching linear time, and 5 iterations can make 95% of the node tags stable. The disadvantage is that the algorithm results are unstable, and the results that may be performed multiple times are different.basic label propagation algorithm is sometimes formed too large ("monster") of the community.

Link Aggregation:

As shown in the above table in terms of time complexity “Label Propagation algorithm” is better than others. But to detect overlapping communities which is the requirement in different cases “Link Aggregation==>Iteration Scan” algorithm is better.

So based on our requirement of the desired communities we may switch to different algorithms.

Git Hub Repository

https://github.com/ras1234/Community_Detection

References

1. http://esatjournals.net/ijret/2013v02/i14/IJRET20130214017.pdf
2. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6859714
3. https://arxiv.org/pdf/cond-mat/0309508.pdf
4. https://arxiv.org/pdf/0709.2938.pdf
5. http://www.pnas.org/content/99/12/7821.full.pdf
6. http://ieeexplore.ieee.org/document/6859714/
7. http://ieeexplore.ieee.org/abstract/document/7161493/