

COMMUNITY DETECTION IN COMPLEX NETWORKS
(KARMAŞIK AĞLARDA KOMÜN TARAMA)

by
Keziban ORMAN, B.S.

Thesis

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE

Date of Submission : May 14, 2010

Date of Defense Examination : June 8, 2010

Supervisor : Dr. Vincent Labatut

Committee Members : Asst. Prof. Dr. S. Murat Egi
Asst. Prof. Dr. S. Emre Alptekin

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude to Dr. Vincent Labatut for his invaluable guidance and helping during the preparation of this dissertation. I would like to mention his patience; giving me inspiration when I was stuck at dead-ends.

I would like to thank to my parents and especially my sister Dr. Güneş Orman for her incredible patience and her endless support. I have very much benefited from the discussions with my dear teacher Prof. Hocine Cherifi in the Computer Engineering Department, and Chantal Cherifi. I thank them all. I would like to thank all my teachers who guided me to improve myself with their great contributions.

Keziban ORMAN

June, 2010

TABLE OF CONTENTS

Acknowledgement.....	ii
Table of Contents	iii
List of Symbols	vii
List of Figures	x
List of Tables.....	xiv
Abstract	xv
Résumé.....	xix
Özet	xxiii
1 Introduction	1
2 Complex Networks.....	5
2.1 Graph Modeling	5
2.2 Notion of Complex Network.....	6
2.3 Definitions.....	8
2.3.1 Nodes and Links.....	8
2.3.1.1 Label and Direction.....	8
2.3.1.2 Degree and Strength	9
2.3.1.3 Hub, Authority & Bottleneck and Bridge	10
2.3.2 Node Sequences	11
2.3.2.1 Walk, Trail and Path.....	11
2.3.2.2 Distance	12
2.3.3 Node Sets.....	13
2.3.3.1 Dyad and Triad.....	13

2.3.3.2	Clique, k-core and k-plex	14
2.3.3.3	Component	15
2.3.3.4	Community	16
2.4	Properties.....	18
2.4.1	Homophily and Heterophily	18
2.4.2	Transitivity	18
2.4.3	Small World	18
2.4.4	Hierarchy	19
2.4.5	Community Structure	20
2.4.6	Network Resilience	20
2.4.7	Spectral Properties.....	21
2.5	Measures.....	22
2.5.1	Counts and Density	22
2.5.2	Distance-Based Measures	23
2.5.3	Node Centrality	24
2.5.4	Degree-Based measures	26
2.5.5	Transitivity Coefficient	28
2.5.6	Modularity	29
2.6	Summary	30
3	Community Detection	31
3.1	Main Algorithms	32
3.1.1	Divisive Algorithms	33
3.1.1.1	Girvan and Newman’s Edgebetweenness	33
3.1.1.2	Radicchi <i>et al.</i> ’s Radetal.....	34
3.1.2	Modularity Based Optimization Algorithms.....	35
3.1.2.1	Newman’s Fast Greedy	35
3.1.2.2	Blondel <i>et al.</i> ’s Louvain	37
3.1.2.3	Reichardt and Bornholdt’s Spinglass	38
3.1.3	Spectral Algorithms.....	40
3.1.3.1	Newman’s Leading Eigenvector	40
3.1.3.2	Donetti and Muñoz’s Commfind.....	42
3.1.4	Dynamic Algorithms	42

3.1.4.1	Van Dongen’s Markov Cluster	43
3.1.4.2	Latapy and Pons’ Walktrap	44
3.1.4.3	Rosvall and Bergstrom’s Infomap.....	46
3.1.5	Other Methods.....	49
3.1.5.1	Raghavan <i>et al.</i> ’s Label Propagation.....	49
3.1.5.2	Rosvall and Bergstrom’s Infomod	51
3.2	Comparing Algorithms.....	53
3.2.1	Algorithms Properties	53
3.2.2	Normalized Mutual Information	54
3.2.3	Artificially Generated Networks	55
4	Random Network Models	57
4.1	Models without Community Structure	57
4.1.1	Erdős–Rényi Model.....	57
4.1.2	Configuration Model	59
4.1.3	Barabási-Albert Model	61
4.1.4	Evolutionary Preferential Attachment Model	62
4.2	Models with Community Structure	64
4.2.1	Girvan-Newman Model.....	65
4.2.2	Lancichinetti <i>et al.</i> Model	66
4.3	Summary	68
5	Generation of Realistic Networks	70
5.1	Original LFR	70
5.1.1	Maximal and Average Degrees	72
5.1.2	Network Size	73
5.1.3	Mixing Coefficient	75
5.1.4	Other Parameters	78
5.1.5	Conclusion.....	78
5.2	Modified LFR.....	79
5.2.1	Proposed Modification	79
5.2.2	Average Degree	80
5.2.3	Network Size	81

5.2.4	Mixing Coefficient	84
5.2.5	General Comparison.....	87
5.2.6	Conclusion.....	89
6	Community Detection Algorithms Comparison	91
6.1	LFR-CM Results	92
6.1.1	Divisive Algorithm Performance	93
6.1.2	Modularity Based Optimization Algorithms Performance	94
6.1.3	Spectral Algorithms Performance	95
6.1.4	Dynamic Algorithms Performance.....	96
6.1.5	Other Algorithms Performance	98
6.1.6	Joint Effects of Parameters.....	99
6.1.7	Summary	100
6.2	Compared Results	101
6.2.1	Divisive Algorithm Performance Change	101
6.2.2	Modularity Based Optimization Algorithms Performance Change	103
6.2.3	Spectral Algorithms Performance Change	105
6.2.4	Dynamic Algorithms Performance Change	106
6.2.5	Other Algorithms Performance Change	108
6.2.6	Summary	110
7	Conclusion.....	113
	References	117
	Biographical Sketch	123

LIST OF SYMBOLS

CN	Complex Network
n	Node Number
m	Link Number
$G_{n,m}$	Graph Representation with n nodes m links
US	United States
β	Community Size Distribution Exponent
$A(i, j)$	Adjacency Matrix of a Network
L_{ij}	Laplacian Matrix of a Network
d	Network Density
p	Probability
$C_b(t)$	Betweenness Centrality
$C_c(t)$	Closeness Centrality
$C_k(t)$	Degree Centrality
$C_e(t)$	Eigenvector Centrality
C_c	Network Centralization
C	Transitivity Coefficient
Q	Network Modularity
WWW	World Wide Web
NP	Non-deterministic Polynomial
EB	Edge Betweenness Algorithm
RA	Radetal Algorithm
FG	Fastgreedy Algorithm
LV	Louvain Algorithm
SG	Spinglass Algorithm
H	Hamiltonian

LEV	Leading Eigenvector Algorithm
CF	Commfind Algorithm
MCL	Markov Cluster Algorithm
WT	Walktrap Algorithm
INP	Infomap Algorithm
LP	Label Propagation Algorithm
IND	Infomod Algorithm
I	Mutual information
MDL	Minimum description length
NMI	Normalized mutual information
$G_{n,p}^{ER}$	Erdos Renyi Network with n node p probability
$\langle k \rangle$	Average Degree
l	Average Distance
CM	Configuration Model
$G_{D,n}^{CM}$	Configuration Model Network with n node D degree sequence
k_{max}	Maximum Degree of Network
γ	Degree Distribution Exponent
ER	Erdos Renyi Model
BA	Barabasi Albert Model
$G_{n,k}^{BA}$	Barabasi Albert Network with n nodes and k average degree
EV	Evolutionary Preferential Attachment Model
GN	Girvan Newman Model
LFR	Lancichinetti, Fortunato and Radicchi model
μ	Mixing coefficient
p_{in}	Probability of having a link between the nodes in same community
p_{out}	Probability of having a link between the nodes in different community
n_c^{max}	Node Number of the Biggest Community
LFR-CM	Original LFR Model with Configuration Model Network
LFR-BA	Modified LFR Model with Barabasi Albert Model Network

LFR-EV	Modified LFR Model with Evolutionary Preferential Attachment Model Network
ε	selection pressure in EV model
b	points scored for unilateral cooperation in EV model

LIST OF FIGURES

Figure 2.1 A scientific network based on citation patterns.	6
Figure 2.2 Friendship network of children in a US school.	8
Figure 2.3 Node degree difference between directed and undirected networks	9
Figure 2.4 Hub and bottleneck representation	10
Figure 2.5 A walk from node 0 to node 7 is shown with red line.	11
Figure 2.6 A trail from node 7 to node 7 (left), a trail from node 0 to node 7 (right)....	12
Figure 2.7 A path between node 0 and node 13 is shown with red line.....	12
Figure 2.8 Three networks with different properties (undirected, directed and weighted)	13
Figure 2.9 A network with 8 nodes and including a clique with 5 nodes.	14
Figure 2.10 A network with 8 nodes and including a 2-clique with 5 nodes.....	14
Figure 2.11 A network with 4 components. Node 6 is isolated.	15
Figure 2.12 A dendrogram with several hierarchical levels.....	16
Figure 2.13 Different community structures.	17
Figure 2.14 An example of hierarchical structured network.....	19
Figure 2.15 Networks with (left) and without (right) a community structure.....	20
Figure 2.16 The diameter of network is shown by red line.....	23
Figure 2.17 Two networks with different centralizations.	26
Figure 3.1 Visualization of the steps of LV algorithm.....	38
Figure 3.2 Representation of creation of flow matrix.	43
Figure 3.3 An example of WT execution and its result community structure.	45
Figure 3.4 Detecting communities by compressing the description of information flows on networks.	47
Figure 3.5 Detecting communities by compressing the description of information flows on networks.	48

Figure 3.6 Nodes are updated one by one as we move from left to right.	50
Figure 3.7 Basic framework for detecting communities as a communication process. .	51
Figure 4.1 ER networks which are created with 20 nodes. The network (left) is created with $p = 1/10$ and the other (right) is created with $p = 1/2$	59
Figure 4.2 A network created by CM with $n = 50$. Degree sequence is created by exponential degree distribution.	61
Figure 4.3 A network created by BA model with $n = 200$ (left), cumulative distribution of degree of same network (right).....	62
Figure 4.4 A network created by EV model. $n = 50, b = 1.5, \varepsilon = 0.99$	64
Figure 4.5 A schematic representation of a GN model generated network.	65
Figure 4.6 Two networks created by LFR model. Creation parameters; $n = 100, \langle k \rangle = 5, k_{max} = 15, \gamma = 3, \beta = 2$. The network (left) is created with $\mu = 0.05$ and the other (right) is created with $\mu = 0.95$	67
Figure 5.1 The change of average distance with average degrees for all network sizes in LFR-CM model. Here $\gamma = 3, \beta = 2$ and $\mu = 0.5$	73
Figure 5.2 Average distance (left) and degree correlation (right) changes with network sizes in LFR-CM model. Here $\gamma = 3, \beta = 2, \langle k \rangle = 30$ and $\mu = 0.5$	74
Figure 5.3 The change of transitivity with network sizes in LFR-CM model. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$ and from top to bottom each line corresponds different μ starting with 0.05 and ending 0.95 with 0.05 step.	75
Figure 5.4 The change of average distance (top left), degree correlation (top right) and transitivity (bottom center) with mixing coefficient on all network sizes in LFR-CM model. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$	77
Figure 5.5 The change of average distance with average degrees for all network sizes in LFR-BA (left) and LFR-EV (right) model. Here $\beta = 2$ and $\mu = 0.5$	81
Figure 5.6 The change of average distance (top left) , degree correlation (top right) and transitivity (bottom center) with network sizes in LFR-BA model. Here $\beta = 2, \langle k \rangle = 30$ and $\mu = 0.5$	82
Figure 5.7 The change of average distance (top left) , degree correlation (top right) and transitivity (bottom center) with network sizes in LFR-EV model. Here $\beta = 2, \langle k \rangle = 30$ and $\mu = 0.5$	83

Figure 5.8 The change of average distance (top left), degree correlation (top right) and transitivity (bottom center) with mixing coefficient on all network sizes in LFR-BA model. Here $\beta = 2$ and $\langle k \rangle = 30$.	85
Figure 5.9 The change of average distance (top left), degree correlation (top right) and transitivity (bottom center) with mixing coefficient on all network sizes in LFR-EV model. Here $\beta = 2$ and $\langle k \rangle = 30$.	86
Figure 5.10 Comparison of average distance (top left), degree correlation (top right) and transitivity (bottom center) on three network models. Here $\beta = 2$, $\langle k \rangle = 30$ and $n = 5000$.	89
Figure 6.1 RA performance on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$.	94
Figure 6.2 Comparison of optimization algorithms performance on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$.	95
Figure 6.3 Comparison of spectral algorithms performance on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$.	96
Figure 6.4 Comparison of dynamic algorithms performances on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$.	97
Figure 6.5 Comparison of other algorithms performances on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$.	99
Figure 6.6 SG (left) and WT (right) performance with joint influence of the mixing coefficient and the average on LFR-CM networks. Here $\gamma = 3$, $\beta = 2$ and $n = 5000$.	100
Figure 6.7 RA performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$.	102
Figure 6.8 FG (top left), LV (top right) and SG (bottom center) performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$	104
Figure 6.9 CF (left) and LEV (right) performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV network. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$	106
Figure 6.10 MCL (top left), WT (top right) and INP (bottom center) performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$	108

Figure 6.11 LP (left) and IND (right) performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV networks. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$	109
Figure 6.12 Comparison of the eleven algorithm's performance on 1000 (left column) and 5000 (right column) nodes LFR-BA (top), LFR-CM (center) and LFR-EV (bottom) networks. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$	112

LIST OF TABLES

Table 2.1 Real-world networks properties values	30
Table 3.1 Algorithms comparison	54
Table 4.1 Payoff of EV model	63
Table 4.2 Summary of the generation models without community structure	68
Table 4.3 Summary of the generation model with community structure	69
Table 5.1 LFR-CM parameter values	71
Table 5.2 Pearson Correlation of Parameters on Properties for $n = 1000$	71
Table 5.3 Pearson Correlation values of parameters on properties of LFR-BA and LFR-EV networks for $n = 1000$	80
Table 6.1 Values of the algorithms parameters	91
Table 6.2 Community sizes and mixing thresholds in function of the networks size and average degrees	92

ABSTRACT

Complex networks have become very popular during the last decade. They allow modeling a given system by representing its components and their relationships with nodes and links, respectively. This powerful tool has been attracting the interest of searchers not only from a theoretical, but also from an applicative side, because of their wide range of applicability. Complex network analysis helps to surface the hidden properties of the studied systems which cannot be discovered at first sight. One of the most prominent sub-domains in complex network analysis is community detection, which consists in searching cohesive subgroups in complex networks. This popular research topic has applications in many fields such as biology, social sciences, physics, etc. For example, one can search related web services on a network which is created by web services inputs relations, while others can search genes related to metastasis and cancer development using a network created from protein interactions. This wide range of applications led to many different algorithms based on different ideas such as graph partitioning, hierarchical clustering, networks spectral properties, optimization techniques or dynamic methods.

Until now, algorithms performing community detection have usually been tested on both real-world and artificial networks. The former are costly and difficult to obtain, whereas the latter were most of the time not realistic enough to be used for this purpose. In this context, being able to generate networks with realistic properties is crucial for the reliability of algorithms testing. Recently, Lancichinetti *et al.* designed a method to produce realistic networks, with a community structure and power-law distributed degrees and community sizes. This model first applies the configuration model to create an initial scale-free network, and then it rewires some of its links in order to make the community structure appear. Lancichinetti *et al.* model (LFR-CM) gives direct control on some network properties such as network size, average degree or

community structure through some generation parameters. However, other realistic properties such as degree correlation and transitivity are not directly controlled and are missing from the generated networks.

In this work, we study the effect of the realism of artificially generated networks on community detection algorithms. We first propose a modification of the Lancichinetti *et al* model to improve the realism of the networks it generates, using two variants of the preferential attachment model: Barabási and Albert’s original preferential attachment model (LFR-BA) and Poncela *et al.*’s evolutionary preferential attachment model (LFR-EV). We replaced the configuration model with those ones and keep the same rewiring process. We analyze the transitivity, average distance and degree correlation properties of the modified networks and compare them to those generated with the original approach. According to our results, the mixing coefficient has the strongest effect on those properties, among all generation parameters. It is noted μ and represents how isolated the communities are, topologically speaking: the lower it is, and the more separated the communities are. Average distance and transitivity decrease non-linearly for all three models along μ increase. ANOVA and Tukey’s test show a significantly smaller average distance ($p < 0.05$) and greater degree correlation ($p < 0.05$) for our modified models than for the original one, which means they are more realistic for these properties. Concerning the transitivity property, the original model exhibits higher values only for $\mu < 0.3$, and the decrease is much more important than for the modified models, leading to transitivity values not significantly different for $\mu > 0.6$ ($p > 0.5$). In other terms, the original model generates more realistic networks in terms of transitivity only for very small μ values, and all three models are either more realistic or equivalently realistic after the 0.3 limit. Globally speaking, both modified models are more robust relatively to changes in μ , i.e. the properties measured on the generated networks are much more stable, especially for the degree correlation. In summary, our modification allows producing networks which are more realistic and/or more robust to changes in μ . This last point is very important in our context, since when one applies a community detection algorithm on networks with different μ , he wants to assess the effect of μ on the algorithm performances, and not some mixed effect involving other network properties.

We then apply eleven different community detection algorithms, representative of five different categories: divisive (Radet al (RA)), modularity-based optimization (Fastgreedy (FG), Louvain (LV), Spinglass (SG)), spectral (Leading Eigenvector (LEV), Commfind (CF)), dynamic (Markov Cluster (MCL), Walktrap (WT), Infomap (INP)) and others (Infomod (IND), Label Propagation (LP)) on networks generated with the original and modified models. The algorithms performances are assessed using the normalized mutual information measure. We first compare in details the algorithms performances on the original model. Like for the properties, the mixing coefficient is again the most effective parameter on the algorithms performances. When expressed in function of μ , most of these performances take the form of a reverse sigmoid (LV, SG, MCL, WT, INP, IND), each algorithm being characterized by a different slope and inflection point. For a few algorithms, the performance decrease more or less linearly (FG, CF). For the rest of them, the decrease is non-linear but no trend seems to emerge (LEV, RA, LP). ANOVA does not allow to distinguish most performances for networks whose communities are clearly separated (small μ), especially when considering algorithms from the same category. However, significant differences ($p > 0.05$) start to appear when μ reaches a so-called mixing threshold, which corresponds to an equilibrium point between the proportion of inter- and intra-community links. Besides pure performance, algorithms can also be compared in terms of robustness by observing the dispersion of their results on equivalent networks. Based on these criteria, we can conclude dynamic algorithms exhibit better result, followed by some modularity-based algorithms (SG and LV). Spectral algorithms are not robust. Moreover, we can order the tested algorithms: $LEV < CF < FG < RA < IND, LP < LV, SG < MCL, WT < INP$. Using these results as a reference, we apply the same algorithms on networks generated with both modified models, in order to assess the effect of realism on their performances. According to our results, when considering algorithms categories, the relative performances are not affected much by the realism of the considered networks, although we observe statistically significant differences (all $p < 0.01$ for ANOVA and Tukey's tests for all pairs of models) on each algorithms performances on three different collections. We can state that except for RA and LP, all algorithms perform worse on LFR-BA-generated networks (all $p < 0.01$ for ANOVA and Tukey's test). Their performances start decreasing for lower μ values than when

applied on networks generated with LFR-CM or LFR-EV. There are also some order changes on algorithms individual performances for all three models. While LP, MCL and IND performances are significantly lower on LFR-EV than on LFR-CM networks ($p < 0.01$ for ANOVA and Tukey's tests), spinglass algorithm performs better ($p < 0.01$). We also observe that RA, LEV and CF are more robust on networks generated with the modified models, having smaller dispersion and more stable plots.

Our study does not highlight big changes on algorithms performances, but it nevertheless shows the generative model significantly affects these performances. The main difference between the compared models being their level of realism and robustness to changes in μ , we conclude network realism regarding average distance, degree correlation and transitivity has definitely an effect on current community detection performance algorithms: the more realistic the network topology and the more difficult for them to identify the actual community structure.

Keywords: Community Detection, Random Networks, Network Realism, Network Generation, Network Properties, Algorithms Comparison.

RESUME

Au cours de la dernière décennie, les réseaux complexes sont devenus très populaires en tant qu'outils de modélisation. Ils permettent de modéliser un système donné en représentant ses composants par des nœuds et leurs interactions par des liens. Ce puissant outil suscite l'intérêt de la communauté scientifique non seulement d'un point de vue purement théorique, mais aussi relativement aux domaines d'application les plus variés, offrant un large éventail d'applications possibles. L'un des sous-domaines les plus importants dans l'analyse de réseau complexe est la détection de communautés, qui consiste à rechercher des sous-groupes de nœuds cohésifs dans les réseaux complexes. Ce type d'approche est appliqué dans de nombreux domaines comme la biologie, les sciences sociales et de la physique. Le travail réalisé dans ces champs très divers a conduit à de nombreux algorithmes de détection de communautés, basés sur des principes très différents tels que le partitionnement de graphe, la classification hiérarchique, l'utilisation des propriétés spectrales du réseau, de techniques d'optimisation ou de méthodes dynamiques. Ces algorithmes sont habituellement testés sur des réseaux modélisant des systèmes réels, mais aussi sur des réseaux générés artificiellement. Les premiers sont coûteux et difficiles à obtenir, alors que les derniers ne sont la plupart du temps pas réalistes. Dans ce contexte, être capable de générer des réseaux avec des propriétés réalistes est crucial pour la fiabilité des résultats des tests comparatifs. Récemment, une étape décisive a été franchie par Lancichinetti *et al.*, qui ont conçu une méthode pour produire des réseaux réalistes possédant une structure de communauté (LFR-CM), et dont les distributions du degré des nœuds et de la taille des communautés suivent une loi de puissance. Ce modèle applique d'abord le *configuration model* pour créer un réseau initial *scale-free*, puis en redirige les liens afin de faire apparaître la structure de communauté désirée. Le modèle de Lancichinetti *et al.* permet de contrôler directement, au moyen de paramètres, certaines propriétés du réseau telles que sa taille, son degré moyen ou la structure des communautés.

Cependant, d'autres propriétés réalistes telles que degré de corrélation et la transitivity sont absentes. Dans ce travail, nous étudions l'effet de réalisme des réseaux générés artificiellement sur des algorithmes de détection de la communauté.

Nous proposons tout d'abord une modification sur le modèle de Lancichinetti *et al.* pour améliorer le réalisme des réseaux qu'il génère, en utilisant deux variantes du modèle d'attachement préférentiel; le modèle original de Barabási et Albert (LFR-BA) et une modification évolutionniste de Poncela *et al.* (LFR-EV). Nous avons remplacé le configuration model par ceux-ci, mais sans changer le processus redirection des liens dans le processus de Lancichinetti *et al.*. Nous analysons les propriétés de transitivity, distance moyenne et corrélation du degré dans les réseaux générés avec l'approche originale et dans ceux générés avec nos modifications. Selon nos résultats, le coefficient de mélange est, parmi les paramètres du modèle générateur, celui dont l'influence est la plus forte sur ces propriétés. Il est à noter qu'une valeur faible de μ (proche de 0) correspond à des communautés clairement séparées, alors qu'une valeur élevée proche de 1) correspond à des communautés extrêmement mélangées. Notre étude montre que la distance moyenne et la transitivity diminuent de façon non-linéaire pour les trois modèles quand μ augmente. L'ANOVA et le test de Tukey montrent que les modèles modifiés produisent des réseaux dont la distance moyenne est significativement plus faible en moyenne ($p < 0.05$), et la corrélation de degré significativement plus grande ($p < 0.05$). Ceci signifie que les modèles modifiés produisent des réseaux plus réalistes, du moins pour ces propriétés-là. En ce qui concerne la propriété de transitivity, les valeurs du modèle original sont significativement supérieures seulement pour $\mu < 0.3$, et de plus leur diminution est beaucoup plus importante que pour les modèles modifiés. Quand μ dépasse des valeurs modérées ($\mu > 0.6$), les trois modèles ne sont pas significativement différents ($p > 0.5$). En d'autres termes, le modèle original ne génère des réseaux plus réalistes en termes de transitivity que pour des valeurs de μ très petites, et les modèles modifiés sont soit plus réalistes, soit autant réaliste pour des valeurs modérées ou élevées de μ . Il faut aussi souligner que les modèles modifiés sont globalement plus robustes relativement à l'évolution de μ , c'est à dire les propriétés mesurées sur les réseaux générés sont beaucoup plus stables, en particulier pour la corrélation du degré. En

résumé, notre modification permet de produire des réseaux qui sont plus réalistes et/ou plus robustes aux changements de μ . Ce dernier point est très important dans notre contexte, car quand on applique un algorithme de détection de la communauté sur des réseaux générés avec différentes valeurs de μ , on veut évaluer l'effet de μ sur les performances des algorithmes, et non pas l'effet joint de μ et d'un ensemble d'autres facteurs liés aux propriétés du réseau.

Nous appliquons ensuite onze algorithmes de détection de communautés, représentant cinq types d'approches différentes: divisive (Radetal(RA)), par optimisation de la modularité (Fastgreedy (FG), Louvain (LV), Spinglass (SG)), spectrale (Leading Eigenvector (LEV), Commfind (CF)), dynamique (Markov Cluster (MCL), Walktrap (WT), Infomap (INP)) et autres (Infomod (IND), Label Propagation (LP)), sur les réseaux générés par les modèles originaux et modifiés. Les performances des algorithmes sont évaluées en utilisant la mesure d'information mutuelle normalisée. Nous commençons par comparer en détail les performances des algorithmes sur le modèle original. Tout comme pour les propriétés, le coefficient de mélange est de nouveau le paramètre le plus important en termes d'effet sur les performances des algorithmes. Lorsqu'elle est exprimée en fonction de μ , la performance prend la forme d'une sigmoïde inversée pour la plupart des algorithmes (LV, SG, MCL, WT, INP, IND). Chaque algorithme se caractérise par une pente et point d'inflexion différente. Pour quelques algorithmes, la baisse des performances est plus ou moins linéaire (FG, CF). Pour les autres, la diminution n'est pas linéaire, mais aucune tendance régulière ne semble se dégager (LEV, RA, LP). L'ANOVA ne permet pas de distinguer la plupart des performances pour les réseaux dont les communautés sont nettement séparées (μ petit), en particulier lorsqu'on considère des algorithmes de la même catégorie. Toutefois, des différences significatives ($p > 0.05$) commencent à apparaître lorsque μ atteint un seuil de mélange, qui correspond à un point d'équilibre entre la proportion de liens inter- et intra-communautaires. Outre la performance pure, les algorithmes peuvent également être comparés en termes de robustesse, en étudiant la dispersion de leurs résultats sur des réseaux équivalents (en termes de propriétés). Sur la base de ces critères, nous pouvons conclure que les algorithmes dynamiques sont les plus efficaces, suivis de quelques algorithmes à base de modularité (SG et LV). Les algorithmes

spectraux ne sont pas robustes. Il nous est possible d'ordonner les algorithmes: $LEV < CF < FG < RA < IND$, $LP < LV$, $SG < MCL$, $WT < INP$.

En utilisant ces résultats comme référence, nous appliquons ensuite les mêmes algorithmes sur les réseaux générés avec les deux modèles modifiés, afin d'évaluer l'effet du réalisme sur la détection de communauté. Selon nos résultats, lors de l'examen des catégories d'algorithmes, les performances relatives ne sont que peu influencées par le réalisme des réseaux en question, même si les différences observées sont statistiquement significatives ($p < 0.01$ pour l'ANOVA et les tests de Tukey), pour chaque algorithmes et sur les trois collections. Nous pouvons affirmer que, à part pour RA et LP, tous les algorithmes sont moins bons sur les réseaux générés par LFR-BA que sur les autres ($p < 0.01$). La valeur du coefficient de mélange à partir de laquelle les performances commencent à diminuer est plus basse pour LFR-BA. Pris individuellement, on observe des changements dans l'ordre des algorithmes sur les trois modèles. Alors que LP, MCL et IND sont moins bons pour LFR-EV que pour LFR-CM ($p < 0.01$), l'algorithme SG est lui significativement plus performant ($p < 0.01$). Nous observons également que les algorithmes comme RA, LEV et CF sont plus robustes sur les collections modifiées, ayant moins de dispersion et des performances plus régulières.

Notre étude ne relève pas d'énormes changements dans les performances des algorithmes appliqués aux réseaux générés avec les différents modèles. Toutefois, il est clair que leurs performances diffèrent significativement. Les principales différences entre les modèles concernent le niveau de réalisme des réseaux générés, et la stabilité de leurs propriétés topologiques relativement aux variations du coefficient de mélange. Nous pouvons donc conclure qu'en termes de distance moyenne, transitivité et corrélation du degré, le réalisme du réseau analysé a une influence sur les performances des algorithmes actuels de détection de communautés : plus les propriétés topologiques sont réalistes et plus l'identification des communautés réelles est difficile.

Mots clés : Détection de Communauté, Réseaux Aléatoires, Réalisme des Réseaux, Génération de Réseaux, Propriétés des Réseaux, Comparaison d'Algorithmes.

ÖZET

Etkileşim halindeki nesneleri düğümler ve aralarındaki ilişkiyi de bağlantılar kullanarak modelleyen karmaşık ağlar geçtiğimiz on yılda çok ilgi çekmeye başladı. Bu etkin araç, geniş çerçevedeki kullanılabilirliğiyle, yalnızca akademik dünyada değil aynı zamanda birçok uygulama alanında da yer buldu. Çünkü karmaşık ağ analizi, uygulandığı sistemin ilk bakışta belirlenemeyecek özelliklerinin ortaya çıkarılmasını mümkün kılar. Birbirleriyle daha sıkı ilişki halinde olan grupları saptamayı kapsayan komün arama, karmaşık ağ analizinin önemli bir alt dalı olarak karşımıza çıkmakta. Bu popüler araştırma konusu biyoloji, sosyal bilimler, fizik gibi birçok temel bilimde uygulanmaktadır. Komün arama ile örneğin, web servis girdileri kullanarak oluşturulmuş bir ağda, birbirleriyle bağlantılı web servisler saptanabilir ya da protein etkileşimleriyle oluşturulan bir ağda, metastaz ya da kanser yayılması saptanabilir. Geniş uygulama alanı, komün arama algoritmalarının çok çeşitli olmasına neden olmaktadır. Birçok farklı algoritma, farklı alanlardaki başka uygulamaların uyarlaması olarak karşımıza çıkmaktadır. Örneğin; graf bölümleme, hiyerarşik kümeleme, grafların spektral özelliklerini kullanma, optimizasyon yöntemleri ya da dinamik metotlar bunlardan bazılarıdır. Günümüze kadar bu algoritmalar çoğunlukla gerçek ağlar ve kısmen de yapay ağlar üzerinde test edilmiştir. Gerçek bir ağı inşa etmek hem zaman hem de emek açısından çok pahalıya mal olmaktayken, yapay ağlar da çoğunlukla gerçeklikten uzaktır. Bu nedenle, gerçek ağ özelliklerini taklit edebilen yapay ağlar, algoritma testleri için çok kullanışlı olmaktadır. Kısa bir süre önce, Lancichinetti *et al.* gerçek ağların birçok önemli özelliğini kapsayan bir model ortaya koymuştur (LFR-CM). Bu model, içerisinde komün bulunan, derece ve komün büyüklükleri power-law dağılım gösteren ağlar üretmektedir. LFR-CM model, yapılandırma modelini kullanarak serbest ölçekli bir ağ yaratmakta ve ardından bazı linklerin yerlerini değiştirerek komünleri oluşturmaktadır. LFR-CM model, ağ büyüklüğü, ortalama derece ya da komün yapısı gibi bazı ağ özellikleri bakımından, oluşturma parametreleri

aracılığıyla direkt kontrol sağlarken, derece korelasyonu ya da geçişkenlik gibi bazı önemli özellikleri kontrol edememektedir.

Bu çalışmada, yapay karmaşık ağların gerçeklik düzeylerinin komün bulma algoritmaları üzerindeki etkilerini araştırdık. Bunun için Lancichinetti *et al.* modelin ürettiği ağların gerçeklik düzeyini artıracak bir değişiklik yaptık, öncelikli bağlanma modelinin iki sürümünü kullandık: Barabási ve Albert'in öncelikli bağlanma modeli (LFR-BA) ve Poncela *et al.*'ın gelişimli öncelikli bağlanma modeli (LFR-EV). Link yer değiştirme sürecini sabit tutarak, yapılandırma modelini, yerine bu modelleri kullandık. Değiştirilmiş ve orijinal ağlar üzerinde geçişkenlik, ortalama uzaklık ve derece korelasyonu özelliklerini analiz ettik ve karşılaştırdık. Bulduğumuz sonuçlara göre, karıştırma katsayısı bu özellikler üzerindeki en etkili parametre oldu. μ şeklinde not edilen bu parametre, komünlerin ne kadar belirgin olduğunu tayin etmektedir. Bu parametrenin değeri ne kadar düşükse, komünler birbirinden o kadar ayrıktır, ne kadar yüksekse o kadar karışmıştır. Ortalama uzaklık ve geçişkenlik, her üç model için μ artışı ile doğrusal olmayan şekilde azalır. ANOVA ve Tukey testleri, değiştirilmiş modeller ile orijinal model karşılaştırıldığında, fark edilir küçüklükte ortalama uzaklık ($p < 0.05$) ve büyüklükte derece korelasyonu ($p < 0.05$) sonucu vermektedir. Bu sonuçlar, yapılan değişimler sonucu daha gerçekçi ağlar oluştuğunu göstermektedir. Geçişkenlik özelliği incelendiğinde ise, orijinal modelin sadece $\mu < 0.3$ değerleri için daha yüksek sonuçlar verdiği görülmüştür. Orijinal modeldeki geçişkenlik düşüşü, değiştirilmiş modele göre çok daha fazla göze çarpmaktadır. Değiştirilmiş modellerde $\mu > 0.6$ için geçişkenlik değeri pek fazla değişmemektedir ($p > 0.5$). Başka bir deyişle, orijinal model yalnızca küçük μ değerleri için, geçişkenlik anlamında, daha gerçekçi ağlar oluşturmaktadır. Genel konuşmak gerekirse, her iki değiştirilmiş model de μ değişimlerine karşın daha karardır. Ölçülen özellikler, özellikle derece korelasyonu, daha sabittir. Özetle, yaptığımız değişiklikler daha gerçekçi ve/veya μ değişimlerine karşın daha kararlı ağların oluşmasını sağlamıştır. Bu ikinci nokta, bizim çalışmamızda çok önemlidir çünkü komün arama algoritmalarının performans değerlendirmesi yapılırken, μ değişimlerine karşın performans değişimleri göz önüne alınmalıdır. Burada herhangi başka bir özellik ile karışmış bir etki olması, değerlendirmeyi sağlıklı kılar.

Daha sonra, beş farklı kategoriden on iki adet komün bulma algoritmasını hem orijinal hem değiştirilmiş modellerle üretilen ağlar üzerinde deneyip performanslarını gözlemledik. Bu algoritmalar kategorileriyle birlikte şunlardır: Bölümleme (Radetal (RA)), modülarite optimizasyonu (Fastgreedy (FG), Louvain (LV), Spinglass (SG)), spektral (Leading Eigenvector (LEV), Commfind (CF)), dinamik (Markov Cluster (MCL), Walktrap (WT), Infomap (INP)) ve diğer Infomod (IND), Label Propagation (LP)). Algoritma performansları normalleştirilmiş ortak bilgi ölçütü ile saptandı. Elde ettiğimiz sonuçlara göre, algoritmaların kategori ya da bireysel sıralamaları iyileştirmeden çok fazla etkilenmedi. Ancak, bazı algoritmaların daha iyi çalışırken bazılarının daha başarısız olduğunu gördük. Öncelikle, algoritmaların original model üzerindeki performanslarını detaylıca inceledik. Ağ özelliklerinde olduğu gibi, algoritma performanslarında da en etkili parameter karıştırma katsayısı oldu. μ fonksiyonu bakımından açıklanacak olursa, birçok algoritmanın performansı farklı eğim ve düşme noktalarına sahip ters sigmoid (LV, SG, MCL, WT, INP, IND) şeklinde azaldı. Bazı algoritmaların (FG, CF) performansı az ya da çok doğrusal azaldı. Geri kalan ise (LEV, RA, LP), doğrusal olmayan şekilde azaldı. ANOVA, komünler iyi tanımlanmışken (düşük μ değerleri için), özellikle aynı kategorideki algoritmaları ayrıştıracak sonuçlar vermedi. Ancak, μ değeri limite vardığında yani komün içi ve dışı link oranı dengeye geldiğinde, dikkate değer farklar ortaya çıkmaya başladı ($p > 0.05$). Algoritmalar, komün bulma performansları dışında, aynı parametrelerle oluşturulmuş ağlar üzerinde göstermiş oldukları sonuç farklılıklarına göre belirlenen kararlılıklarına göre de değerlendirilebilir. Bu ölçüt temel alındığında, dinamik algoritmaların diğer algoritmalarından daha kararlı davrandıklarını söyleyebiliriz. Bazı modülarite optimizasyon algoritmaları da (SG ve LV) yine kararlı sonuçlar vermektedir. Fakat spektral algoritmalar kararlı sonuçlar verememektedir. Test ettiğimiz algoritmaları şu şekilde sıralayabiliriz: $LEV < CF < FG < RA < IND, LP < LV, SG < MCL, WT < INP$. Elde ettiğimiz sonuçları referans alarak, ağların gerçeklik düzeylerinin etkilerini ortaya çıkarabilmek için aynı algoritmaları değiştirilmiş modellerle oluşturulan ağlar üstünde çalıştırdık. Algoritmaların, bu üç koleksiyondaki uygulamalarında, teker teker performans sonuçları istatistiksel olarak göze çarpan değişimler verse bile (tüm model çiftleri için tüm ANOVA ve Tukey test sonuçları $p < 0.01$), algoritma kategori sıralamasında ağ gerçekçiliğinin çok fazla etkisi olmadığını gördük. Genel olarak RA

ve LP hariç diğer tüm algoritmaların LFR-BA koleksiyonunda daha kötü çalıştığını gözlemledik (tüm ANOVA ve Tukey test sonuçları $p < 0.01$). Performanslar, LFR-BA koleksiyonunda, LFR-CM ve LFR-EV koleksiyonlarından daha küçük μ değerlerinde düşüş gösterdi. Ayrıca, bazı algoritmaların bireysel performansları her üç koleksiyonda farklı sıralamalar gösterdi. LP, MCL ve IND performansları LFR-EV koleksiyonunda, LFR-CM koleksiyonundan çok daha düşüktü (ANOVA ve Tukey's test sonuçları $p < 0.01$). Ayrıca, RA, LEV ve CF'nin değiştirilmiş modellerle üretilen ağlarda daha kararlı çalıştığını gördük.

Yaptığımız çalışma, algoritma performanslarında dramatik değişimler oluşturmada da, kimi önemli değişimlere katkıda bulundu. Yapılan iyileştirmelerin en önemli etkisi, ağların özelliklerin μ değişimlerine göre daha kararlı olmasıdır. Daha kararlı geçişkenlik, derece korelasyonu ve ortalama uzaklık özelliklerine sahip ağlarda çalıştırılan algoritmaların da daha kararlı çalıştığını gözlemledik. Ağ kararlılığının ayrıca, algoritmalar için komün yapısını bulmayı zorlaştırdığını da bu çalışmada göstermiş olduk.

Anahtar Sözcükler: Komün Arama, Rastsal Ağlar, Ağ Gerçekçiliği, Ağ Üretimi, Ağ Özellikleri, Algoritma Karşılaştırma.

1 INTRODUCTION

Complex networks constitute a powerful modeling tool, able to represent most real-world systems. The objects composing the system are represented under the form of nodes while their interactions correspond to links. Thanks to this versatility, they have become very popular during the last decades, attracting the attention of scientists from many different domains. They are now used to model and analyze complex systems in areas like physics, biology, social sciences or computer science [1, 2]. Among the various approaches used to study complex networks properties, community detection has become one of the most popular ones. A community is a cohesive subset of nodes with denser inner links, relatively to the rest of the network [3]. Because of the spread of complex networks, the community detection problem has been studied in many different areas too, resulting in tens of algorithms based on a whole range of principles: hierarchical clustering, optimization methods, graph partitioning, spectral properties, etc. [4].

Until recently, those algorithms were tested on a few real-world and/or artificial networks. Limiting these tests to real-world networks can be considered as an issue for several reasons. First, building such networks is a costly and difficult task, and determining reference communities can only be done by experts. This leads to small networks, where actual communities are not always defined objectively, or even known. Second, a complex network is characterized by various properties like its average degree, degree distribution, shortest average path, etc. By definition, it is not possible to control these features in a real-world network. This means the algorithm is tested only on a very specific and limited set of features. Hopefully, artificial networks allow overcoming these limitations. First, by using some generative model, it is possible to automatically build many of them. Moreover, since they are generated, objective reference communities are defined. Finally, depending on the selected

generative model, various properties can be directly controlled. The only point of concern with artificial networks is their level of realism. Indeed, for algorithm testing to be relevant, the generated networks must exhibit realistic properties. For this purpose, generative models are generally defined in order to mimic known real-world networks properties. Of course, current knowledge regarding these properties may not be exhaustive, and we can consequently never be completely assured the generated networks are perfectly realistic. For this reason, tests on artificial networks should be seen as complementary to tests on real-world networks.

In terms of artificial networks, there are a few models to generate networks with communities with various levels of realism. Girvan-Newman created a small sized network with all nodes has roughly the same degree; and all communities have the same size [5]. More recently, a different approach appeared, based on rewiring [6, 7]. The method by Lancichinetti *et al.* [7] is based on the configuration model and creates power-law distributed community sizes by rewiring. This model exhibits many real-world network properties with controlling some parameters. Nevertheless, it also has some noticeable limitations regarding the low transitivity and close to zero degree correlation measured in the generated networks [8]. According to Newman [1], real-world networks usually have a clearly non-zero degree correlation, and their transitivity, or clustering coefficient, is relatively high.

Interestingly, improvement on the realistic aspect of the generated networks has a noticeable effect on most community detection algorithms. A performance drop was observed when authors switched from equal-sized communities to heterogeneous distributions [9, 10]. This allowed showing one algorithm may not perform on the same level depending on the size of the considered communities. The introduction of a power-law degree distribution also made the benchmarks more discriminatory, allowing to highlight differences between algorithms whose performances were considered similar before [7]. The fact Lancichinetti *et al.*'s method, which is the most realistic one to date, still has room for improvement naturally raises two questions, which we will try to answer in this work: 1) how is it possible to produce more realistic networks, and 2) will this have an effect on community detection algorithms.

In this work, we study the effect of the realism level of the network generation model on community detection algorithms performances. We focus on undirected, unweighted networks with non-overlapping communities, because almost all existing community detection algorithms are dedicated to this type of networks. We first propose two modifications of Lancichinetti *et al.*'s model in order to improve the realism level of the generated networks. We then create a collection of networks with different network sizes, average degrees and a whole range of values for other parameters of interest, using both the original Lancichinetti *et al.* model and our own modified versions. We study and compare the properties of the generated networks in order to assess their level of realism. In the second part of our work, we select eleven different community detection algorithms and apply them to the three networks collections. We then compare their results in order to evaluate the effects of the networks realism on their relative and absolute performances.

The rest of the document is organized as follows. In section 2, we explain the fundamental concepts related to Complex Networks. We describe how they are used as modeling tools and give the associated terminology. We list the properties most widely used to characterize them, and the associated measures used to quantify these properties. Importantly, we indicate how real-world networks generally behave for each property, which allows us to precisely define the notion of *network realism*, which is central to this work. Section 3 includes detailed information on the community detection task. We individually review the most widely used algorithms and classify them into different categories depending on the principle they are based on. We explain how these algorithms can be compared in terms of performance, which problems exist and how they can be tackled by using artificially generated networks. In section 4, we review the main random models used for network generation, and compare them from a theoretical point of view. In section 5, we study empirically the model by Lancichinetti *et al.* and describe the limitations we observed in terms of the realism of the generated networks. We describe the modifications we propose to solve these problems, and then proceed with an empirical study of the resulting modified models. In section 6, we present the results obtained when we applied the community detection algorithms on the networks generated in the previous section. We compare and comment these results

for the original Lancichinetti *et al.*'s model and for our own modifications, in terms of community detection algorithms partitioning performance. In section 7, we conclude our study by commenting the obtained results. We also propose future works which will be complementary of this study.

2 COMPLEX NETWORKS

2.1 Graph Modeling

In many application fields, different instruments or objects interact or are correlated, forming a system. An efficient way to study a system and understand its dynamics is to model it. Modeling a system consists in representing it in a simplified and abstract way, leading to an easier comprehension. A popular modeling method is to represent its objects and their relationships by a graph, allowing using a set of generic methods for its study. In a graph, objects are represented by nodes and relationships between them are represented by links. They are used as a modeling tool in many different domains such as physics, genetic, sociology, computer science, etc. For example, to explain the molecules attractions on each other in physics, to discover unknown relationships between any two people who shares common friends in sociology, to analysis Web in information technology, to study the statistical properties of metabolic networks in biology or to find the optimal flight from Paris to New York in daily life [1, 11]. Because graphs are used in different areas, many different vocabularies exist in the literature to refer to the same concepts. For example, a graph can be named “network”, the objects in interaction are called “vertex”, “node”, or “actor” and the relationships between them are called “edge”, “arc”, “link”, “tie”, “bond” or “connection”. Generally, graph representation which is $G_{n,m}$ such that n is node number and m is link number is used for mathematical networks models. In this work, “network” will be used for the model, “node” will be used for the objects and “link” will be used for the relationship between them.

2.2 Notion of Complex Network

Modern age real systems are no longer small. They contain millions of actors and very different type of relationships. For example, a co-authorship network of biology scientists has 1 520 251 nodes and 11 803 064 links [12] and the Altavista web search engine has 203 540 046 nodes and 2 130 000 000 links [13]. This results in so-called complex networks (CN), characterized by a huge number of nodes, possibly different types of relationships, and non-trivial topological features. The properties of nodes or links depend on the system in question. For example, the links may be directed, weighted or the nodes can have a size, color or many other different attributes which represents a property of real system object. We see an example of directed and weighted network with 3024 directed and weighted links which show citations between journals on Figure 2.1 [14]. The network image is simplified to show groups rather than single journals.

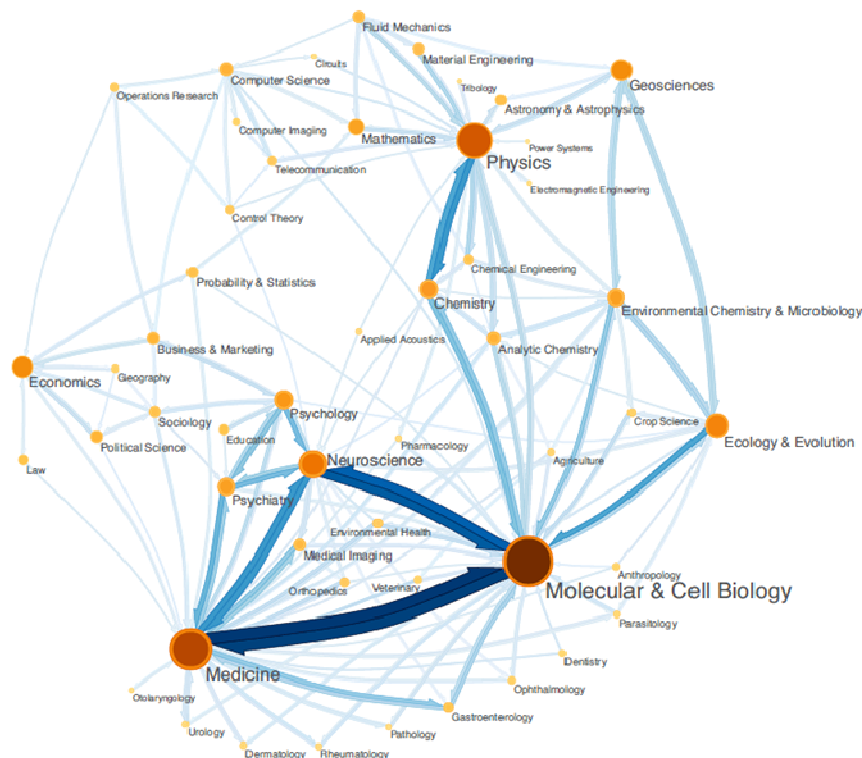


Figure 2.1 A scientific network based on citation patterns.

Another CN example is given on Figure 2.2 [1]. Relationships of this network are determined by asking the participants and are directed. Nodes are color-coded according to race, as marked, and the split from left to right in the figure is clearly primarily along lines of race. The split from top to bottom is between middle school and high school, i.e., between younger and older children.

A CN can also have different type of linking between the nodes. For example, an egocentric CN can contain both links representing the human relationships between the ego and its neighbors, friends or family and links representing the ego's professional or educational background as the companies which he worked or his university etc. However, in this work, we will focus only on networks with single-typed attribute-less nodes and unweighted, undirected, single-typed links. To analyze CNs, it is not appropriate to apply classical network measures, neither computationally nor interpretatively. First, the exact processing of those measures will be too expensive. Second, they reflect low level information which may be useless when some synthetic information is necessary. Consequently, specific tools were designed to extract useful information from CNs. Here, getting a synthetic (useful) information often means considering only a part of the information conveyed by the CN. For example, all nodes may not be important in a 1 million nodes. One may be interested only in identifying a few central nodes. At that point, statistics are a good tool to infer general properties from a lot of observations. Many tools have been dedicated to CN coming from this field, but other approaches exist too (genetic algorithm, etc.). Whatever the tool, analyzing CN firstly relies on some basic concepts coming from graph theory. There are many different concepts which help with statistical inference. Those concepts form the foundation of more complicated analyses.

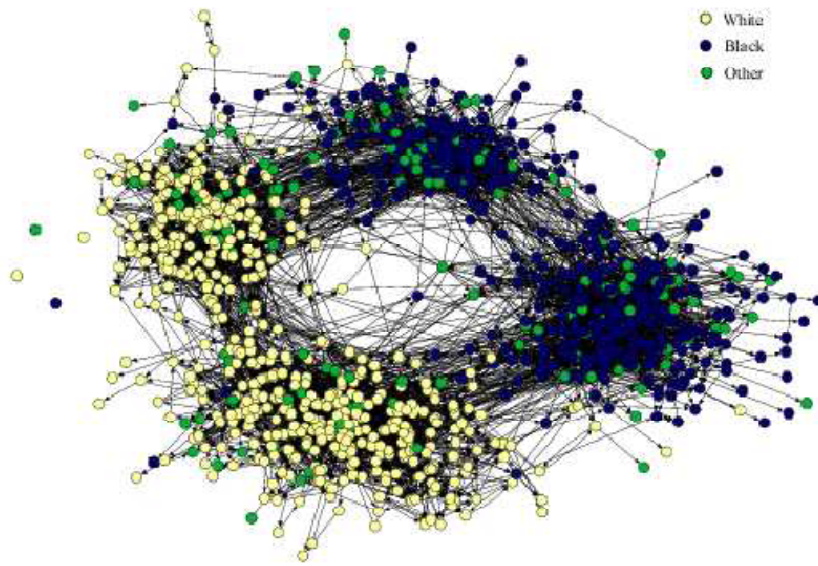


Figure 2.2 Friendship network of children in a US school.

In the rest of this section, we first define the most basic graph-related concepts, then present which properties allow characterizing a graph, and finally explain how these can be measured and quantified.

2.3 Definitions

Several more complex concepts can be built based on nodes or links, i.e. the simplest graph concepts. These concepts are then used as a base for various analysis methods, by considering for instance their values, counts or distribution.

2.3.1 Nodes and Links

2.3.1.1 Label and Direction

As mentioned before, some systems have nodes or links attributes. In the networks, these are classically represented under the form of *labels*. For example, in a social network, one might need to consider the gender or age of the considered population. In

the corresponding to network, this translates into labels associated to each node. Links labels are more common, and generally take the form of numerical values called *weights*, expressing the strength of the corresponding to relationships. Another important link attribute is the *direction*, which introduces a distinction between the connected nodes: one is the tail, the other is the head, and the link represents an asymmetrical relationship from the tail towards the head. Depending on the links, a network is said to be (un)weighted and (un)directed.

2.3.1.2 Degree and Strength

The number of neighbors of a node corresponds to its degree. In another words, the total number of links directly associated to a node is the degree of that node [15, 16]. We will note the degree of a node k in this document. For a directed network, one can distinguish in- and out-degree. These are defined by the direction of the links which represent the relationships. The in-degree of a node is the total number of other nodes which have links directed towards it; and its out-degree is the total number of other nodes to which it directs its own links. One can see the difference of degrees in directed and undirected networks on Figure 2.3. The node 3 on the left-side undirected network has degree 3. The node with same number on the right-side directed network has in-degree 2 and out-degree 3. The nodes which have no degrees are called *isolated*.

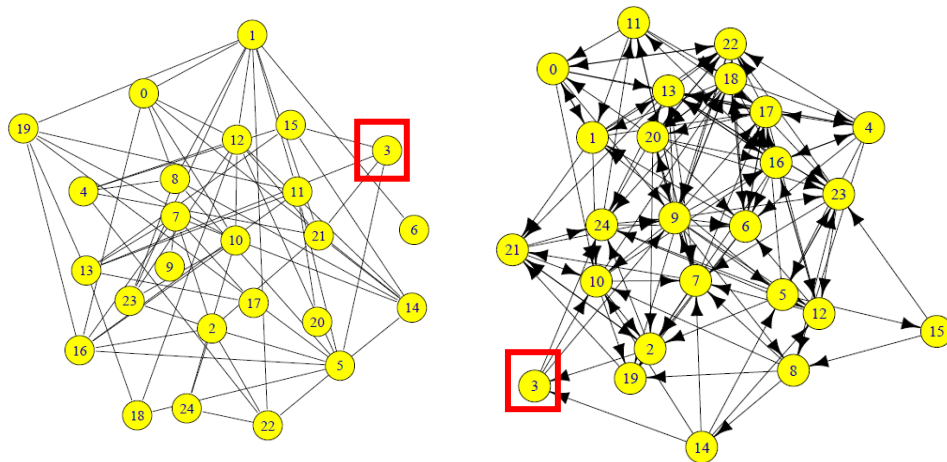


Figure 2.3 Node degree difference between directed and undirected networks

The sum of the weights of the links attached to considered node is its *strength*. Strength is a more generalized definition of degree which takes into account the weights.

2.3.1.3 Hub, Authority & Bottleneck and Bridge

In a directed network, we can identify two types of important nodes: hubs and authorities. *Hub* and *authority* are the node which have respectively more in- and out-degree than the mean in- and out-degree of network [17]. These two concepts refer to the same definition for undirected networks. *Bottleneck* is the node which controls the flow from one part to another part of the network whose deletion increases the number of unconnected sub networks [18]. *Bridge* is the link which is connected to bottlenecks [18]. The removal of the bridges also causes to part the network into unconnected sub networks. The green node in Figure 2.4 is a hub and a bottleneck, the dark blue one is not hub but is a bottleneck, the orange one is a hub but not a bottleneck and the red one is neither a hub nor a bottleneck.

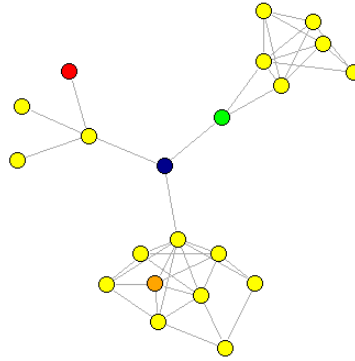


Figure 2.4 Hub and bottleneck representation

2.3.2 Node Sequences

2.3.2.1 Walk, Trail and Path

A *walk* from node i to node j is any sequence of adjacent links that begins on a link attached to node i and ends with a link attached to node j [15]. Nodes and links can appear several times in a walk, as long as these constraints are respected. Process of the monetary exchange is an example where a dollar bill travels from one person to another with no limitations. One can possess the same bill several times and give it to the same people. That is why; a link can be used again and again during the exchange process.

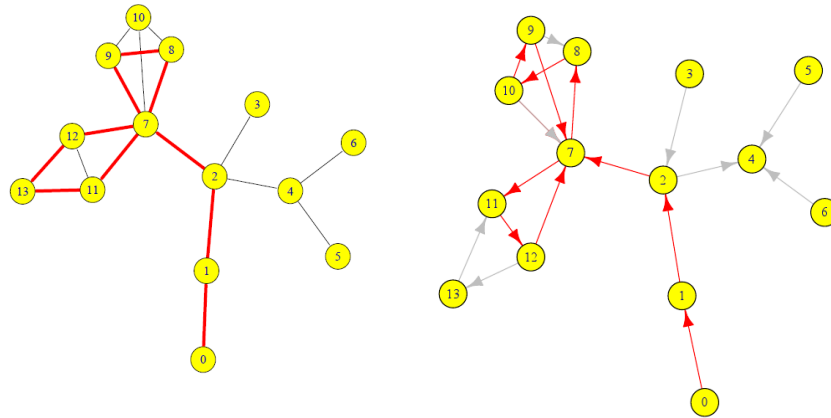


Figure 2.5 A walk from node 0 to node 7 is shown with red line.

A *trail* is any walk between two nodes that includes a given link no more than once (the same node can appear several times, though). The gossip process, where a story is moving through an informal network, can be a good example for a trail. It never travels between the same pair of nodes, but can reach the same node multiple times by following the trails.

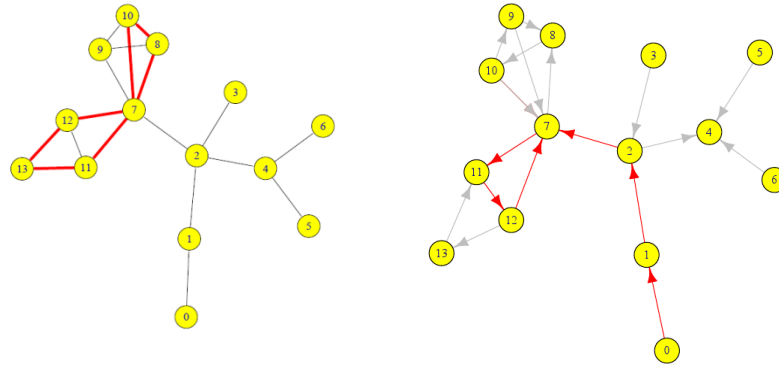


Figure 2.6 A trail from node 7 to node 7 (left), a trail from node 0 to node 7 (right)

A *path* is a walk in which each link and node appear at most once [15]. A viral infection, which activates effective immunological response, can be a good example for a path because every node can be infected at most once. A *cycle* is a closed path, i.e. a path beginning and ending with the same node.

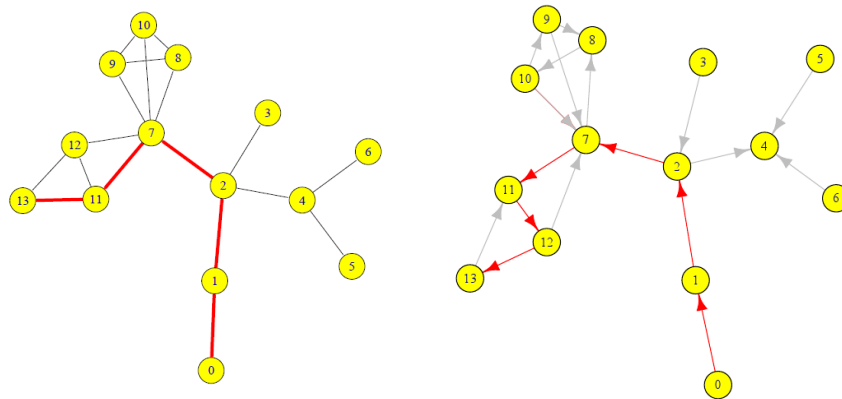


Figure 2.7 A path between node 0 and node 13 is shown with red line.

2.3.2.2 Distance

In an unweighted network, the *length* of a path corresponds to the number of links it includes. The *distance* between two nodes is the length of the shortest path existing

between them [15]. In a weighted network, the length of a path is processed by summing the weights of the links it includes, resulting in a slightly different definition of the distance. When there is no path between the two considered nodes (which can be the case when the network has several components, or in directed networks), their distance is theoretically infinite. However, in many situations, this infinite value causes problems, and this distance is then considered to be zero (cf. distance-based measures).

We see an unweighted undirected network with 8 nodes on the left side, a weighted network with same number of nodes on center and a directed network with same number of node on right side of Figure 2.8. The distance between any two nodes of those three networks is different than each other because of their direction (non-direction) or weights.

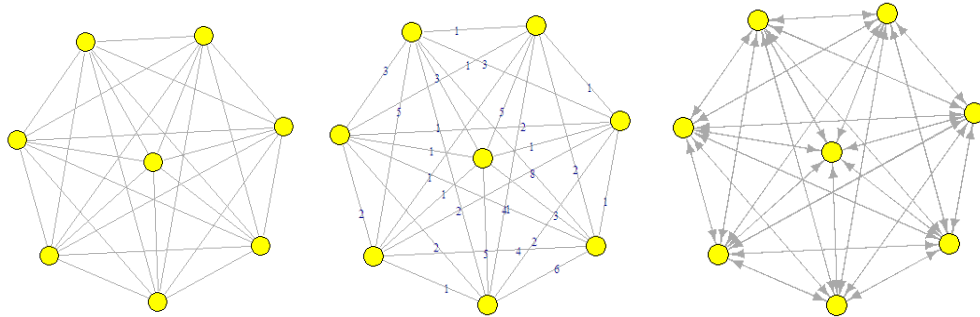


Figure 2.8 Three networks with different properties (undirected, directed and weighted)

2.3.3 Node Sets

2.3.3.1 Dyad and Triad

A *dyad* is any combination of two nodes. There can be a link, mutual links or no link in a dyad. It is the smallest node set and concerns mutual relationships.

A *triad* is any subset of three nodes. For a directed graph, one can find sixteen possible types of triads but for an undirected graph, there are only 8 types of triads. Triads are used to analyze network transitivity, which will be explained later on this document.

2.3.3.2 Clique, k-core and k-plex

A *clique* is a maximally complete subset of nodes. Each node is linked to all the other nodes from the clique [15]. Its size corresponds to the number of nodes it contains. Finding all cliques of a given size in a network is NP-hard.

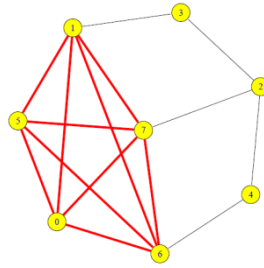


Figure 2.9 A network with 8 nodes and including a clique with 5 nodes.

An *n-clique* is a maximal subset of nodes such that the distance between each pair of nodes is n . If n is equal to one, then the structure is a clique.

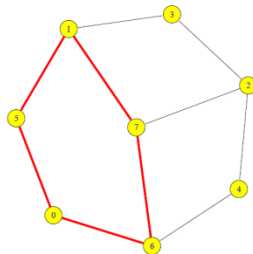


Figure 2.10 A network with 8 nodes and including a 2-clique with 5 nodes.

A *k*-plex is a subset of n nodes in which each node is connected to at least $n - k$ nodes also belonging to this subset. If k is equal to one, each node is connected to all other nodes and we have a clique.

A *k*-core is a subset of nodes whose every node is connected to at least k nodes from the same subset [15].

2.3.3.3 Component

A *component* is a sub-network in which any node is reachable from any other one by a walk [16]. Put concisely, it is a maximal connected subgraph. For an undirected network, a component is a set of connected nodes with no links with other nodes from the same network. But for directed networks, it is less straightforward. A component is said to be *strongly connected* if there is a directed walk between each pair of nodes. It is called *weakly connected* if there is at least an undirected walk between each pair of nodes. A network with only one component is said to be *connected* [1, 16]. An isolated node (i.e. a node with a degree zero) is a component of its own.

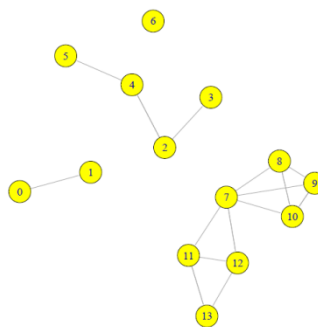


Figure 2.11 A network with 4 components. Node 6 is isolated.

2.3.3.4 Community

Like components, *communities* are cohesive subsets of nodes. The difference is they are generally connected; only the links are denser inside the communities than between them. Dense networks do not have communities. Only sparse networks have community structures [4]. Communities have been studied by many scientists from various domains, and for this reason several names are used to refer to them, like *modules*, *partitions*, *clusters* or *cohesive subgroups*. Several alternate definitions also exist. The *local* definition of the community concept is based on the aggregation of simple substructures like cliques, cores, n-cliques, k-cores, k-plex and their neighborhood [4]. It allows a node to be a part of several communities at the same time (i.e. communities can overlap) but on the other hand, it can be costly to implement on large networks. Communities can also be defined *globally*, i.e. with respect to the network as a whole. Radicchi *et. al* [19] distinguish *strong* and *weak* communities. A strong community is a subgraph such that the internal degree of each node is greater than its external degree, and a weak community is a subgraph such that its total internal degree exceeds its total external degree. Of course, strong communities include weak communities. According to global definition, the nodes belong to only one community which corresponds that communities do not overlap. These type of communities are in general hierarchical. Every high level community can be partitioned to low level ones. The nodes in the sub communities have denser inner links. This hierarchical structure can be represented under the form of so-called *dendrograms*, as shown on Figure 2.12. This dendrogram is cut on second top level. Different colors show different communities.

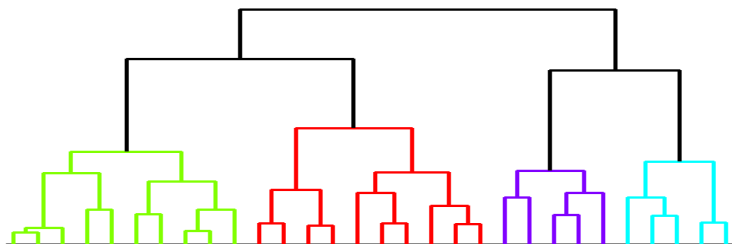


Figure 2.12 A dendrogram with several hierarchical levels.

A community can also be defined basing on *similarity* of the nodes. Nodes belonging to the same community must be more structural similar to each other than the other nodes. In other words, the distance between the nodes in the same community should be shorter than the nodes in the rest of network.

For example, Zachary Karate Club which is represented in Figure 2.13 [4] by image (a) is a social network of friendships between 34 members of a karate club at a US university in the 1970s and it contains 4 communities which show the close friendships of the members [20]. The different colors of image correspond to different communities. Another famous networks including communities is a collaboration network between scientists working at the Santa Fe Institute [5] with 6 communities. This network is represented on Figure 2.13 with image (b); the colors indicate high level communities. And another example is bottlenose dolphins' network with 2 communities [21] which is represented on image (c) is Lusseau's network of bottlenose dolphins. The colors label the communities.

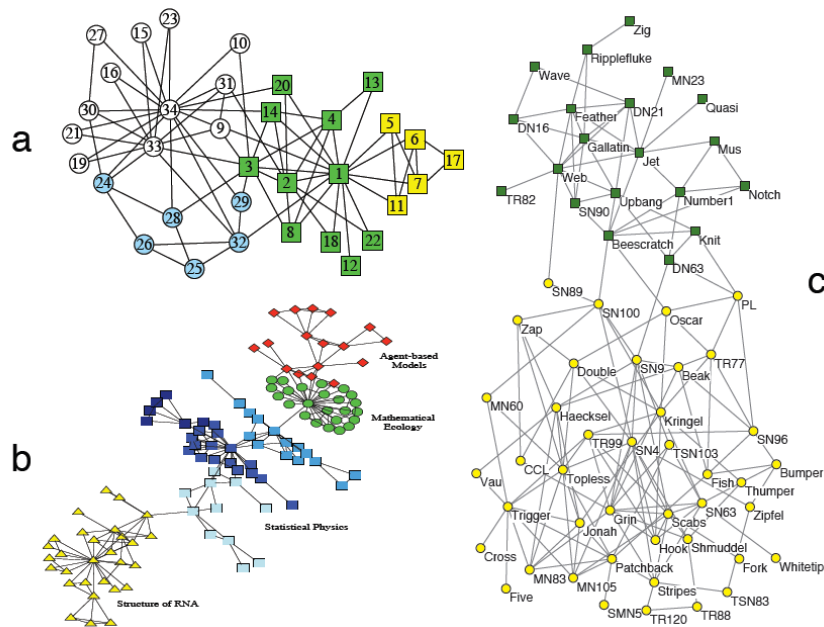


Figure 2.13 Different community structures.

2.4 Properties

Each network can be characterized by some topological properties. We will mention here the most widely spread in the literature. They can be used to describe a given network and discuss how they can be interpreted relatively to the modeled system, or to compare different networks.

2.4.1 Homophily and Heterophily

Homophily (resp. *heterophily*) is the tendency for nodes to be connected to other nodes that are like (resp. unlike) them in some way. Sharing common attributes or characteristics emphasize or causes the relationship between the nodes. They preferentially connect to the other nodes which are similar (resp. different). This similarity (resp. difference) can be due to the topology of network or the attributes of the nodes. For example; in a friendship network, the people who are at same age are higher connected to each other than the rest of the network. In some resource, it is called *assortative mixing* [1].

2.4.2 Transitivity

Transitivity corresponds to the proportion of triangles (three connected nodes) among all possible triads. This property is also called clustering, but this creates a conflict with other meanings of this word, that is why we will use only transitivity here. The more transitive a network is and the more probable it is to observe a link between two nodes with a common neighbor. In other terms, if node A is connected to node B and node B to node C , then there is a higher probability that node A is also connected to node C [1, 11, 16].

2.4.3 Small World

A network is said to have the small world property if the average path length between nodes is small. Milgram's famous six degrees of separation experiment [22] refers to

the idea that everyone is at most six steps away from any other person on earth, so that a chain of, "a friend of a friend" statements can be made to connect any two people in six steps or less. This property can be interpreted as propagation efficiency: spreading on the network remains relatively fast even if the network grows. The empirical studies are shown that many real networks have shortcuts among the nodes which mean they have small world property. In many real-world networks, for a fixed average degree, the average path length increases non linearly with network size [1]. Although their size increases, they still keep having small world property.

2.4.4 Hierarchy

Nodes can be connected with a hierarchical order so the links among the nodes have a hierarchical topology [15]. While we observe less links between the nodes of different levels, we see denser links between the nodes at same level. Some real-world networks have hierarchy; for example, the network which represents the relationship among the employees in a company will carry hierarchy properties. A hierarchical network is represented on Figure 2.14. Each small structure of five nodes, as red framed group, is at first linked to each other and then the groups are linked to other groups and so on.

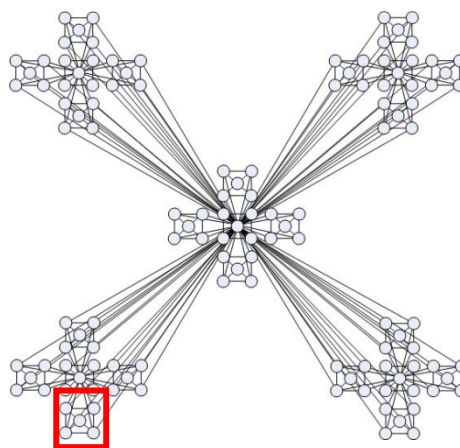


Figure 2.14 An example of hierarchical structured network.

2.4.5 Community Structure

Community structure refers to network topological structure with communities. Some CN have a community structure which refers to the grouping of the nodes. But some others have a homogenous distribution of links, without any community structure. Figure 2.15 represent two networks with and without community structures. Network on left side has a community structure with 4 communities colored with blue, red, green and yellow. Network on right side has no community structure.

It is a common feature in biological and social networks [1]. When the community structure is present, the community size distribution seems to follow a power-law distribution with a parameter β ranging from 1 to 2 [1].

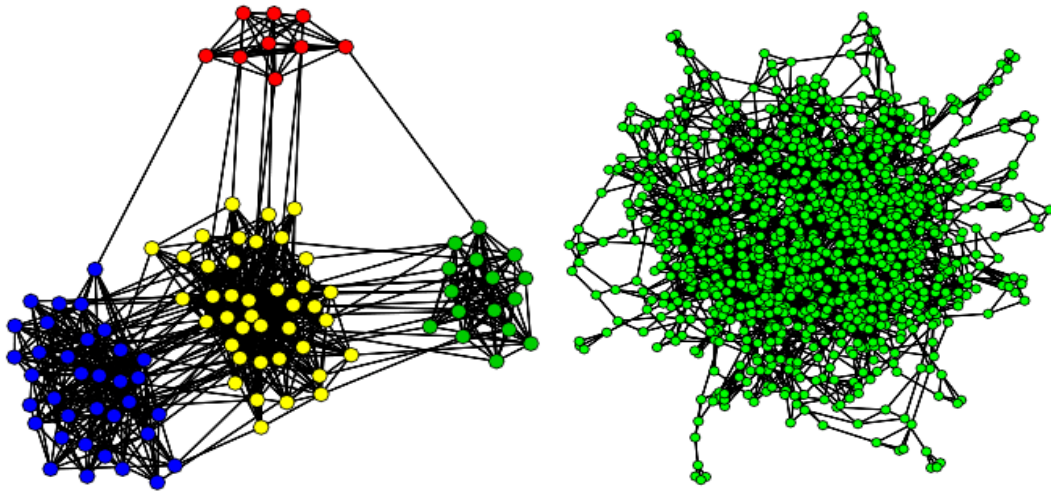


Figure 2.15 Networks with (left) and without (right) a community structure

2.4.6 Network Resilience

Resilience is the robustness of the network to the removal of its nodes, i.e. whether or not it keeps its properties even when its structure is modified. Networks vary in their level of resilience to nodes or link removal [1]. For resilient networks, removing a few

nodes does not affect information flow. The network connectivity, i.e. the existence of paths between pairs of nodes, stays high when some of them, or even nodes, are removed. The typical length of these paths will not change either. But for some more sensitive networks, removing nodes or links can have important effects. This is especially the case when bottlenecks are present. Nodes pairs will become disconnected and communication between them through the network will be impossible anymore.

This property can be very important in epidemiology, where removing nodes in a contact network might correspond for example to vaccinating some persons against a disease. Because vaccination not only prevents the vaccinated individuals from catching the disease but may also destroy paths between other individuals by which the disease might have spread, it can have a wider effect than one might at first think, and careful consideration of the efficacy of different vaccination strategies could lead to substantial advantages for public health.

2.4.7 Spectral Properties

The information of a CN can be described by some mathematical expressions, vectors, matrices. The *adjacency matrix* is one of the most common way of representing how the nodes are connected:

$$A(i, j) = \begin{cases} 1 & \text{if there is a link between nodes } i \text{ and } j \\ 0 & \text{if not} \end{cases} \quad (2.1)$$

When the network is weighted, these weights are used instead of the binary values.

The *Laplacian matrix* is another form of representation of a network and it is also widely used. It is defined as:

$$L_{ij} = \begin{cases} \text{degree of node } i & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and node } i \text{ is neighbor with node } j \\ 0 & \text{otherwise} \end{cases} \quad (2.2)$$

The set of *eigenvalues* of these matrixes is called the *spectrum* of that network. It can be used to find a partition of the network, and detect central nodes [11].

2.5 Measures

Different measures are used to quantify the previously described properties. One can distinguish node level (or local) measures and network level (or global) ones. Node level measures are mainly related to centrality.

2.5.1 Counts and Density

The total number of nodes in a CN is generally referred to as its *size*, and noted n . However, some authors also use the same word to mean the total number of links, usually noted m . CNs can have very large number of nodes with huge number of links.

Density is the general level of linkage among the nodes, calculated as the ratio of the number of existing links to the number of possible links [18]. Its formulation is written as below:

$$d = \frac{m}{n(n-1)} \quad (2.3)$$

For a *complete* network (all nodes are connected to all other nodes), the density is 1. A network is called *sparse* if the number of links is of the same order as the number of nodes ($n \approx km$) and else it is called *dense*.

2.5.2 Distance-Based Measures

The *diameter* of a network is the maximal distance one can find between two of its nodes [16] . Diameter gives the upper length of the distances. It is important information to determine the range of the distances and to understand the topology of the network.

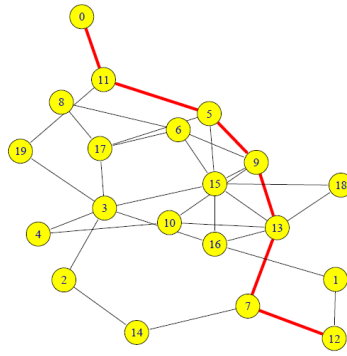


Figure 2.16 The diameter of network is shown by red line.

The *average distance* is the mean value of all distances over all pairs of nodes [15] .

$$l = \frac{1}{n(n-1)} \sum_{i \neq j} d_{ij} \quad (2.4)$$

Where d_{ij} is the distance between nodes i and j . If there are unconnected nodes, this equation diverges. In order to overcome this problem, only connected nodes can be included in the sum. But, this causes a distortion for networks with many unconnected nodes. It shows a small average distance as the networks with high connectivity [15].

2.5.3 Node Centrality

Different measures exist to determine the structural importance of nodes in a network: some are designed for directed networks, some are for undirected networks. Note it is also possible to use undirected measures for directed networks, by either ignoring the links directions or considering separately in- and out-degrees.

The *betweenness centrality* is defined as the number of shortest paths running through the node of interest [23].

$$C_b(t) = \sum_{i < j} \frac{\sigma_{ij}(t)}{\sigma_{ij}} \quad (2.5)$$

Where t is the node of interest, σ_{ij} is the total number of shortest paths from node i to node j , and $\sigma_{ij}(t)$ is the number of shortest paths from i to j running through t . A more central node has more importance on the information flow. Removing nodes with high betweenness centrality may corrupt the information flow and cause an important information loss or slow down its spread.

The *closeness centrality* of a node is the inverse of the sum of the distances between this node and all the other nodes:

$$C_c(t) = \frac{1}{\sum_i d_g(t, i)} \quad (2.6)$$

Where t is the node of interest and $d_g(t, i)$ is the distance from node i to node t . This measure shows how close a node is to the all other nodes. If a node has a high closeness centrality, it is able to get and to spread information efficiently [23].

The *degree centrality* of a node is the proportion of its actual degree to its maximal possible degree.

$$C_k(t) = \frac{k_t}{n-1} \quad (2.7)$$

Where t is the node of interest, k_t is its degree and n is the total number of nodes. More degree they have, more powerful they are [23]. Degree centrality is useful to determine locally central nodes.

Eigenvector centrality is a relative scoring. It is found by the first eigenvector of the adjacency matrix of the network. The centrality of each node is proportional to the sum of the centrality of its neighbors.

$$C_e(t) = \frac{1}{\lambda} \sum_i A_{ti} C_e(i) \quad (2.8)$$

Where t is the node of interest, λ is a constant and A_{ti} is the value of cell located at the t^{th} row and i^{th} column in the adjacency matrix of the considered CN. The nodes which have high eigenvectors are connected to the nodes which are connected to many others (and so on).

Centrality of nodes can have large or small variance depending on the linkage of network. *Centralization* of a networks is the variance of computed node centralities is [23].

$$C_C = \frac{\sum_n (C_C^* - C_C(n))}{\max \sum_n (C_C^* - C_C(n))} \quad (2.9)$$

Where n is any node, C_c^* is the highest value of the selected centrality measure and $C_c(n)$ is the centrality of the node n with selected centrality measure. Network centralization value is 0 if all nodes have equal centrality, like in a ring network for instance, and 1 if one node is much more central than the others, as in a star network. These types of networks are called *highly centralized*.

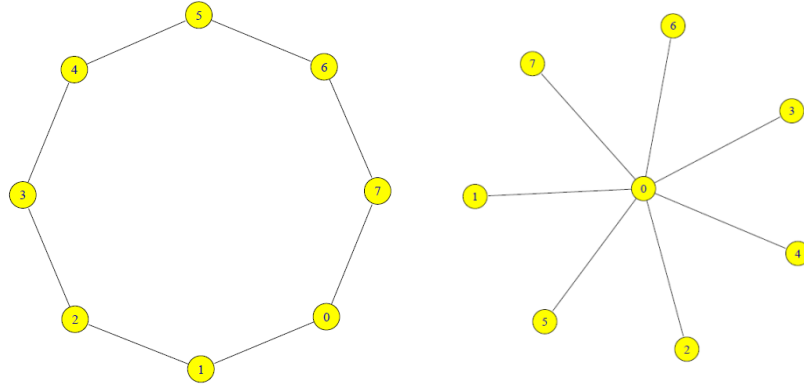


Figure 2.17 Two networks with different centralizations.

We represent networks with different centralizations on Figure 2.17. The nodes of the network on left side have equal degree centrality and this network is not centralized. But the network on right side is highly centralized because one node is very central while the others are not. If these highly central nodes are removed or damaged, the network quickly fragments into unconnected sub-networks (components). The central nodes act as a hub or router. One may define different centralization for each of one of the centrality measures explained before.

2.5.4 Degree-Based measures

Average degree is the mean degree processed over the entire networks node. [15]. It depends on the considered system, the number of links and nodes. The average degree for an undirected, unweighted network can simply be formulized as below;

$$\langle k \rangle = \frac{1}{n} \sum_i k_i \quad (2.10)$$

Here, $\langle k \rangle$ is average degree of network, k_i symbolizes the i^{th} node's degree and n is the total node number. In a real-world network, the average and maximal degrees generally depend on the number of nodes it contains.

Degree correlation allows measuring a special case of homophily (resp.heterophily). For many real-world CN, there is a correlation between the linkage of the nodes and their degrees. Real-world networks usually show a non-zero degree correlation. If it is positive, the network is said to have assortatively mixed degrees, whereas if it is negative, it is disassortatively mixed [1]. According to Newman, social networks tend to be assortatively mixed, while other kinds of networks are generally disassortatively mixed.

Degree distribution over a whole network carries important information and helps understanding its structure and dynamics [1, 11, 16]. For example, it can give an idea about the position of a new node which will be added to current CN as where it will be added, to which nodes it will be linked but more importantly, it helps us to understand general statistical behavior of network. In most real-world networks, this distribution follows either a power or an exponential law. In other terms, the probability for a node to have a degree k is either $p_k \approx k^{-\gamma}$ or $p_k \approx e^{-k/K}$ [1]. Networks with a power-law degree distribution are the most common ones. They are called *scale-free*, because their degree distribution does not depend on their size (some other properties may, though). Experimental studies showed that the γ coefficient usually ranges from 2 to 3 [11, 16]. It is known that for values of γ smaller than 3.48, there is a high probability the network contains one giant component and several small ones or even only one component (the network being completely connected). For a real-world scale-free network, it is estimated to be $\langle k \rangle \approx k_{max}^{-\gamma+2}$ and $k_{max} \approx n^{1/(\gamma-1)}$ [1] respectively.

2.5.5 Transitivity Coefficient

The *transitivity coefficient* is used for calculating the transitivity of the network [1]. It is also called *clustering coefficient*. It is the fraction of triangles. In another words, it shows the presence of loop of order three. Thus, one may result if that network is highly clustered or not. It can be calculated by the following equation:

$$C = \frac{3 * \text{number of triangles in the network}}{\text{number of connected triples of nodes}} \quad (2.11)$$

Transitivity coefficient shows also how close a node and its neighbors are to being a clique. For weighted or directed networks, there are different versions of transitivity coefficient. One may calculate not only the loops of order three but also of other orders as the fraction of squares. Furthermore, one may consider the link transitivity which is formularized as below:

$$C_{ij}^{(g)} = \frac{z_{ij}^{(g)} + 1}{s_{ij}^{(g)}} \quad (2.12)$$

Where, $z_{ij}^{(g)}$ is the number of cyclic structures of order g the link (i, j) belongs to, and $s_{ij}^{(g)}$ is the number of possible cyclic structures of order g that can be built given the degrees of the nodes. Many real-world networks are highly transitive. This is one of the important properties of them.

2.5.6 Modularity

Modularity is the measure which has been presented by Newman and Girvan to assess the quality of a network partition. Its maximal value is measured for the exact community structure of the network. Newman and Girvan first define what could be called a community contingency matrix, whose elements p_{ij} represent the fraction of total links from a node in community i towards a node in community j . The fraction of links inside community i is therefore p_{ii} . Moreover, since we are considering undirected networks, we have $p_{ij} = p_{ji}$ and the matrix is symmetric.

Let p_{i+} and p_{+j} be the sums over row i and column j , respectively. If the network has no community structure, or if the considered communities are not defined accordingly to the network structure, than one can suppose the links are randomly distributed. Under this hypothesis, the expected fraction of links inside community i can be estimated as the probability for a link to start in community i , which is p_{i+} , multiplied by the probability to end in community i , which is p_{+i} . The matrix being symmetric, we have $p_{i+}p_{+i} = (p_{i+})^2$. The modularity measure is defined as the difference between the observed and expected fractions of links in each community, summed over all communities:

$$Q = \sum_i p_{ii} - \sum_i (p_{i+})^2 \quad (2.13)$$

When the communities are not better than a random partition, or when the network does not exhibit any community structure, Q is negative or null. Its superior limit is 1, but it can be reached only if the network has a perfect community structure and if the communities have been perfectly detected. For real-world networks, modularity values between 0.3 and 0.7 are considered high.

The modularity measure is known to have some flaws. For example, it is sensitive to community size [24] and it is possible to find partitions of Poisson random networks with relatively high modularity values [25]. There is now more general modularity measure for directed or weighted networks [26-28].

2.6 Summary

We have explained some important definitions, properties and measures of CN in this section. The most common real-world networks properties have been studied before [1, 11]. We research the effect of network realism on community detection algorithms in this study. The phrase *network realism* corresponds to the level of the having those most common real-world properties. These properties and their realistic values are listed on Table 2.1.

Table 2.1 Real-world networks properties values

PROPERTY NAME	VALUE
Transitivity	[0.1; 0.8]
Degree Correlation	significantly different from zero
Average Distance	small world
Power-law degree distribution	with exponent [2; 3]
Power-law community size distribution	with exponent [1; 2]

3 COMMUNITY DETECTION

In this work, we focus on the notion of community structure, which is one of the properties described in previous section. We encounter the community structure in many CN from different areas. For example, in information technology domain, WWW is designed as a directed CN such that nodes represent web pages and links represent hyperlinks between pages [29]. Communities correspond to the web pages on similar topic. Another example is protein protein networks in biology and bioinformatics domain [30]. Nodes represent protein molecules in the cells and links represent the interactions of those molecules. The molecules interact very frequently with each other if they belong to metastatic cells, which are invasive than the normal cells. Communities correspond to proteins having the same or similar functions. Most communities associated to metastasis and cancer.

Finding community structure is crucial because identifying the community structure in a network can help to classify the nodes according to their structural position. Moreover, by using the boundary and center nodes of the communities, we may obtain knowledge about the information flow or critical objects. The nodes between different communities have important role for information exchange and lead the relationships or central nodes can be more important for the management of the community. Community detection is a NP-hard problem. It means to find the exact community structure can be reduced a *non deterministic polynomial time* problem. More clearly, we can produce any algorithm to detect the exact community structure which can run for small sized network with an acceptable time but for large networks, although it finds them in polynomial time, it is still too slow for applying in real-world. That is why; there exist many approximation solutions to community detection problem [31-33]. Most of those algorithms use a fitness function which optimizes a quality measure. In general, network modularity is used as this measure.

Many different scientists from different disciplines pay attention to the problem of community structure detection. This problem is discussed with many different points of view. It can be expressed as a graph partitioning problem, mining of the CN data, spectral analysis of the networks, an optimization problem, a statistical problem and so on. Most of those approaches have taken into account the structural position of the nodes but there are also a few approaches using nodal attributes. In this section, at first we explain some properties of community structure and at second we describe the community detection methods by classifying them according to their main idea. At last part, we explain how one should compare the algorithms by mentioning comparison measures.

3.1 Main Algorithms

We discuss the community detection algorithms by explaining their characteristic, properties, limits and advantages at this part. In fact, one algorithm can belong to more than one class with different point of views. For example, many algorithms create a hierarchical structure for communities but to decide the hierarchy, they take advantage of various similarity/distance measures or adopt the techniques of different sciences like physical spin model, etc. We have tried to affect the algorithms to the classes by their principal idea of detecting communities. As a result, we grouped them in five different classes.

The first class contains divisive algorithms which are derived from older classic clustering approaches. The third class contains modularity-based algorithms, most of them using modularity as an optimization criterion. The fourth class contains the algorithms using the networks spectral properties. The fifth class contains dynamic algorithms, which are in general using random walk. Finally, the last class contains all the other algorithms which could not be placed in one of the previous classes.

3.1.1 Divisive Algorithms

Until Girvan-Newman algorithm [5], traditional methods were used but they were not compatible for real, huge complex networks. The main idea under the modern divisive algorithms is to find the inter-community links and remove them in order to separate the communities. As it is evident, the crucial point is to decide if the links are at the position of between-communities. One may think that modern divisive algorithms are as same as the traditional ones. But, while traditional algorithms take into account the nodes structural similarity by their links, modern ones uses inter-community links to remove. Although those links can lie between similar nodes, they can be the separator of the communities. We will explain the most popular Girvan-Newman algorithm and its variation Radicchi algorithm[19] at this section.

3.1.1.1 Girvan and Newman's Edgebetweenness

Edgebetweenness searches the links lying between the communities in order to remove and so this finds the communities[5]. The algorithm uses the centrality measure to find the most between links. It is called *edgebetweenness* (EB).

EB has modified the definition of node betweenness and adopts it to the links. Thus the betweenness of a link is basically defined as the number of shortest paths between pairs of nodes that run along it. If a CN contains communities, by its definition, there must be fewer links between the communities and the betweenness of those links must be higher. By removing these links, one separate group from one another and so reveal the underlying community structure of the graph. The algorithm has 4 following steps:

1. Calculate the betweenness for all links in the network.
2. Remove the link with the highest betweenness.
3. Recalculate betweenness for all links affected by the removal.
4. Repeat from step 2 until no links remain.

EB calculates betweenness of all links at each time and this makes the algorithms to run very slow. The complexity of calculating the betweenness is $O(mn)$ where m is total number of links and n is total number of nodes. In worst case, the complexity of EB is $O(m^2n)$ and for sparse networks it costs $O(n^3)$. Of course, after each time step, the number of links and nodes to consider will decrease, especially for strong community structured networks. But EB is still very slow for large networks like web or internet. Another faster variation of EB may be proposed in a form of calculating all betweenness once at the beginning of the algorithm. But it removing the links can cause the change on it.

3.1.1.2 Radicchi *et al.*'s Radetal

Radicchi *et al.* underlined in [19] that hierarchical community detection algorithms which build the dendrogram find subnetworks that are candidates to be considered communities. Thus, one needs other criterions to determine if the subnetwork is a community and if not, the corresponding branch in the dendrogram should not be represented. That is why; they defined strong and weak communities (in section Community Definition). Authors also remarked that if one splits a network randomly in two parts, one very large and the other very small, large subnetwork will respect to strong or weak definition but small one will not. This is valid for random networks too that it is known that they do not carry community structure. So, Radicchi *et al.* propose that separation should create communities only if both two parts respects to strong or weak definition of community. And they improved EB procedure as follows:

1. Choose a definition of community.
2. Compute the link betweenness for all links and remove those with the highest score.
3. If the removal does not split the network go to point 2.
4. If the removal splits the network, test if at least two of the resulting subnetworks respect the definition. If they do, draw the corresponding part of the dendrogram.
5. Iterate the procedure (going back to point 2) for all the subnetworks until no links are left

After overcoming the bad placements on the dendrogram with this solution, Radicchi *et al.*, proposed a new measure for detecting inter-community links to accelerate the algorithm. Link betweenness is the cause of EB's high complexity. That is why; Radicchi *et. al* replaced the measure of link betweenness with link transitivity coefficient [19]. Radetal (RA) algorithm has user parameters g which corresponds to if the transitivity will be calculated on triangle or on square. If the link transitivity is high, link betweenness is low and vice versa without a perfect correlation. The complexity of the algorithm is $O(m^4/n^2)$ where m is link number and n is node number. For sparse networks, this is $O(n^2)$.

The Radetal algorithm seem to fix some of the problems of EB and propose a new and faster measure to find the inter-community links to remove but one still needs a quality assessment to decide the level of dendrogram for best community structure.

3.1.2 Modularity Based Optimization Algorithms

Modularity (Q) which is found by Newman and Girvan [3] attracts the attention of many scientists as a quality assessment and becomes an essential element of many algorithms in this area in spite of all its drawbacks which are mentioned in measure section. An exhaustive optimization is impossible. Although to find the true maximum value of the Q is impossible in polynomial time, there are several algorithms which can approximate the maximum Q in a reasonable time. We will explain some of those algorithms which maximize Q with greedy, simulated annealing or other optimization techniques at this section.

3.1.2.1 Newman's Fast Greedy

Newman remarked in [31] as well as Radicchi *et al.* did [19] that EB algorithm finds always a community structure in given CN even that network does not have any natural such division. Moreover EB is quite slow for large sized networks. Because of all these reason, he oriented to a solution with Q . By using Q , one can distinguish the

qualification of partitioning and avoid inexistent partitioning but the true optimization of Q is very costly. To carry out an exhaustive search of all possible divisions for the optimal value of Q would therefore take at least an exponential amount of time, and is in practice infeasible for systems larger than 20 or 30 nodes. That is why; he choice greedy optimization to apply.

Fast greedy algorithm (FG) of Newman starts with a state in which each node is in their own community and the algorithm repeatedly join communities together in pairs. To choose the join at each step, FG uses greatest increase (or smallest decrease) in Q . Joining a pair of communities which have no inter-link does not gain ground that is why; only the communities having links between them are merged. As a result, FG produces a dendrogram which contains the communities in a hierarchical order and this decides the best partitioning to cut as the level which produces maximum Q . There are maximum $n-1$ steps to create whole dendrogram and at each step, join operation takes $O(n + m)$ time complexity so the complexity of FG is $O(n(n + m))$ and for sparse networks this is $O(n^2)$. This is relatively fast algorithm and searches natural community structure of the networks. However, greedy optimization technique takes into account the local maxima of the criteria measure. This may cause the loose of global maxima. In another word, using local maxima for Q does not assure the true maxima for the partitioning and this may cause to find lack community structure.

A faster variation of FG is proposed by Clauset *et al.* [33]. FG uses adjacency matrix in optimization and repeatedly merges pairs of rows and columns as the corresponding communities are merged. But adjacency matrix contains mostly 0 for sparse networks and FG wastes of time and memory space. Thus, rather than using whole adjacency matrix, they created a matrix containing the change of Q for the merge of each pair of communities and applied greedy optimization on it. Finally, they accelerated the algorithm to complexity $O(n \log^2 n)$.

3.1.2.2 Blondel *et al.*'s Louvain

Blondel *et al.* developed an optimization algorithm too [32]. This algorithm which we call Louvain (LV) is limited not by the computational time but by the storage capacity for very large networks. The authors tested it on 118 million nodes network and observed it worked very fast.

LV has two phases which are repeated iteratively. For a CN with n nodes, LV starts with all of them in their own community. So, there are as many communities as there are nodes. At LV's first phase, for each node, the modularity gain of changing its community with one of its neighbor's community is calculated. Then, the node is placed in the community for which this gain is maximum, but only if this gain is positive. If no positive gain is possible, node stays in its original community. LV applies this procedure repeatedly and sequentially for all nodes until no further improvement can be achieved. Thus, first phase is done.

LV continues with its second phase which considers building a new network whose nodes are the communities which have been found during the first phase. The inner links of the communities are represented by self loops of the new network's nodes and the sums of the outer-links are drawn as links between neighbors. The creation of the network has completed, first phase is reapplied on this new network. The time consuming phase is clearly first phase. The authors are called the combination of these two phases as *pass*. So, passes are repeated until there are no more changes and a maximum of modularity is attained. LV naturally incorporates a notion of hierarchy, as communities of communities are built during the process.

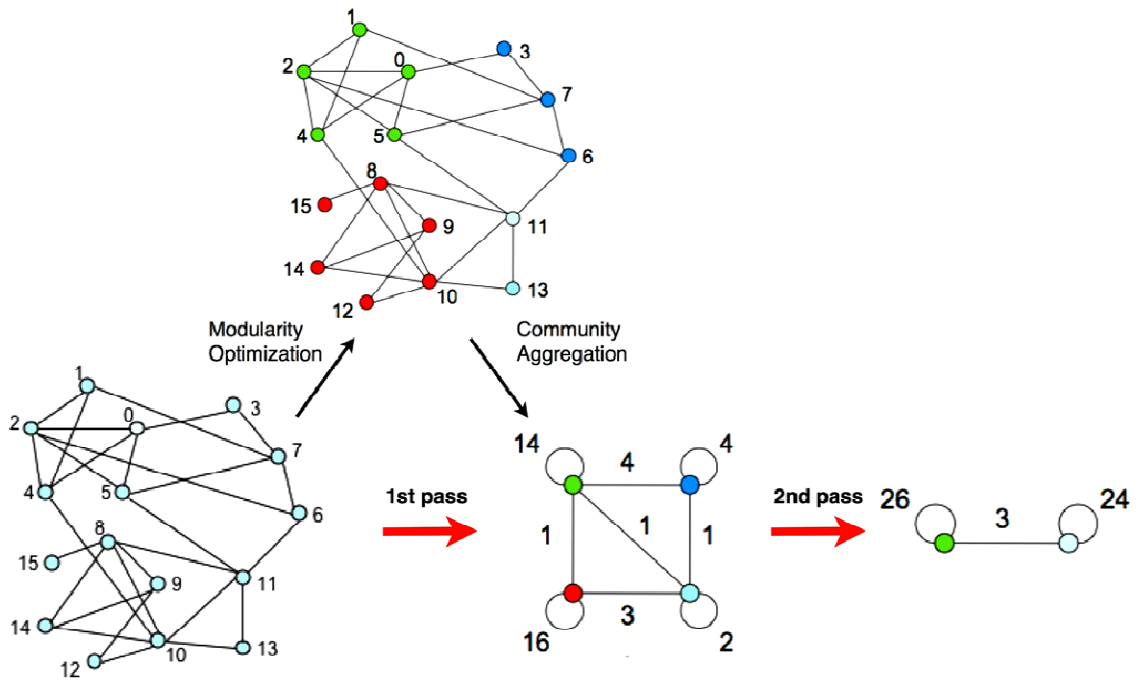


Figure 3.1 Visualization of the steps of LV algorithm.

We have shown the image representation of the phases of LV algorithm on Figure 3.1 [32]. Each pass on image is made of two phases: one where modularity is optimized by allowing only local changes of communities; one where the communities found are aggregated in order to build a new network of communities. The passes are repeated iteratively until no increase of modularity is possible.

LV algorithm seem to have many advantages, this does not need any community number, as many other optimization algorithms, it is easy to implement and LV is very fast, the complexity is linear on typical and sparse data.

3.1.2.3 Reichardt and Bornholdt's Spinglass

Reichardt and Bornholdt also developed an optimization algorithm for community detection [34]. This algorithm is called Spinglass (SG). It uses an analogy between a very popular statistical mechanic model *Potts spin glass* and the community structure of CN, to apply the simulated annealing optimization technique [35].

Potts model aims to describe how nearest neighbor elements with different spins interact with one another [36]. The spins can have the value of one of the q equally spaced angles. *Kronecker function* defines the interaction energy between any two spins of the system and total energy of the system in terms of these interaction energies is described by the function named *Hamiltonian* (H). There is two variant of model such that in first one the interactions of the spins are ferromagnetic. It means, the spins are aligned, at zero temperature; all spins are at same state. Second variant is the case of the interaction is antiferromagnetic so the spins are not necessarily aligned at the ground state of the system and different spin values can exist in homogeneous clusters. Finding community structure problem can be mapped to the finding ground state of an infinite range *Potts spin glass*. Further, H replaces modularity function in that case. This can be defined as below;

$$H = -J \sum_{ij} \delta_{\sigma_i \sigma_j} + \gamma \sum_{s=1}^q \frac{n_s(n_s-1)}{2} \quad (3.1)$$

Here the first term of H describes the nearest neighbor ferromagnetic interaction of the *Potts model*, homogeneous distribution of the spins and second term describes global antiferromagnetic interaction namely diversity. n_s is the number of spins in state s , q is maximum number of states, corresponds to the community number, J and γ are the coupling parameters. *Potts model* can be generalized for infinite range to describe not only nearest neighbors but also any couple of spins' interactions. In that case H can be written as below:

$$H = \sum_{i \neq j} \delta_{\sigma_i \sigma_j} (A_{ij} - \gamma) \quad (3.2)$$

The energy of the system is smaller if spins align within subgroups such that the interaction between the spins in the group will exceeds δ and out the group is smaller than δ . Thus finding the optimal community structure problem can be induced to minimize H function. The minimization of H is carried out by simulated annealing starting with the initial configuration where the spins are randomly assigned to the nodes and q is very high (less or equal than node number).

The SG algorithm is capable to detect overlapped communities. Further, SG can find the communities in a hierarchical order. This is an effective but relatively slow method.

3.1.3 Spectral Algorithms

The spectral representation of CN is used for community detection. For example, one can apply traditional graph partitioning by using spectral partitioning or improve these techniques in such a way to serve to community detection by not determining community sizes or numbers *a priori*. Here, we explain two different spectral algorithms proposed by Newman and Donetti & Muñoz.

3.1.3.1 Newman's Leading Eigenvector

Newman developed a community detection algorithm called Leading Eigenvector (LEV) which uses classical spectral partitioning method [37]. He was inspired by traditional spectral partitioning methods. The main idea under traditional methods was by minimizing the links which lie between partitions, called *cut* in graph partitioning literature, finding the appropriate partitions. To do this, one uses Laplacian matrix and defines *cut size* in terms of it. One of the classical spectral partitioning algorithms is created by Fiedler [38] and popularized by Pothen *et. al* [39]. However, classical spectral partitioning methods require the partition size. This condition is neither appropriate nor realistic for community detection problems. That is why; community detection methods changed or improved these spectral partitioning techniques.

Newman underlined that modularity maximization can be written in terms of eigenspectrum of a matrix which he calls modularity matrix. This is similar to that played by the Laplacian matrix in graph partitioning calculations. Classical graph partitioning mostly concentrate on to minimize the *cut size* but this is not the right thing to optimize for community detection. Because, differently than graph partitioning, good division of the communities in a CN is defined as the number of links between communities are smaller than expected. Thus, using modularity matrix is more relevant than using Laplacian matrix. LEV algorithm initially divides the network in 2 communities and then uses the definition of the modularity such that;

$$Q = (\text{number of links within communities}) \\ - (\text{expected number of such links}) \quad (3.3)$$

Q is rewritten in matrix form as $Q = (1/4m) s^T B s$ where, B is the symmetric matrix with the values of the difference of real and expected links between the nodes, m is the total link size and s is the vector such that 1 if current node belongs to first community and -1 if it belongs to second community. For the detailed equation explanation, see [37]. Once, the modularity matrix is defined as above, classical spectral partitioning is applied by using modularity matrix instead of Laplacian matrix. LEV algorithm finds the eigenvector corresponding to the most positive eigenvalue of the modularity matrix and divides the network into two groups according to the signs of the elements of this vector. This initial version of LEV allow to find just two communities without defining their sizes a priori but most real-world networks has more than two communities. So, more generalized version of LEV is explained in [37]. At its first version a vector s is used to keep the communities of the nodes, to generalize this, a matrix with n columns such that network size and c rows such that community size is used.

3.1.3.2 Donetti and Muñoz's Commfind

Donetti and Muñoz have proposed an algorithm which combines spectral methods with clustering techniques and uses modularity to decide the best community structure on the hierarchical level [40, 41]. We call this algorithm Commfind (CF).

CF takes advantage of the spectral analysis of the Laplacian matrix. Because, this allows projecting the network nodes into an *eigenvector space* of variable dimensionality. Each node is represented by a point in a D-dimensional space in which the coordinates are given by its projections on the first D non-trivial eigenvectors. After representing the nodes as the points in D-dimensional space, CF chooses a metric (Euclidean or angular) in order to calculate the distances of the nodes and then applies traditional hierarchical clustering technique. CF creates the hierarchy by using single linkage and complete linkage clustering. As a result, dendrogram is constituted and one decides the level of the dendrogram by using the Q .

The most costly part of CF algorithm is the part of eigenvalue calculation. Thus, although the optimal number of D is not known *a priori*, one can try to maximize the final modularity value by adjusting D . Donetti and Muñoz used Euclidean and angular distances with single and complete linkage techniques. As the results of their tests, complete linkage produces better results for both distances in terms of modularity.

3.1.4 Dynamic Algorithms

Dynamic algorithms of community detection consist of the algorithms which process the random walks in CN. They use random walks in order to obtain the information to partition the network into communities. Here, we present three different dynamic algorithms by Van Dongen, Pons and Latapy and Rosvall and Bergstrom.

3.1.4.1 Van Dongen's Markov Cluster

Van Dongen developed an algorithm which uses stochastic flows in CN [42]. The main idea is based on random walk such that if a random walker starts from a node and randomly moves to another one which is connected to the starting node, the random walker is more likely to stay within the same community than the different one. By doing the random walks upon the network, one may discover where the flow tends to gather and therefore where the communities are. We have shown an example of a creation of flow matrix on Figure 3.2. The nodes 1 and 3 are linked to 2 in the network which is at top-left on the image. The probability of going from node 1 to node 2 is 1 because there is no other choice as well from node 3 to node 2. But walker can go to node 1 or 3 from node 2. That is why; the probability of choosing one of them is 0.5. The flow matrix is prepared by those probabilities.

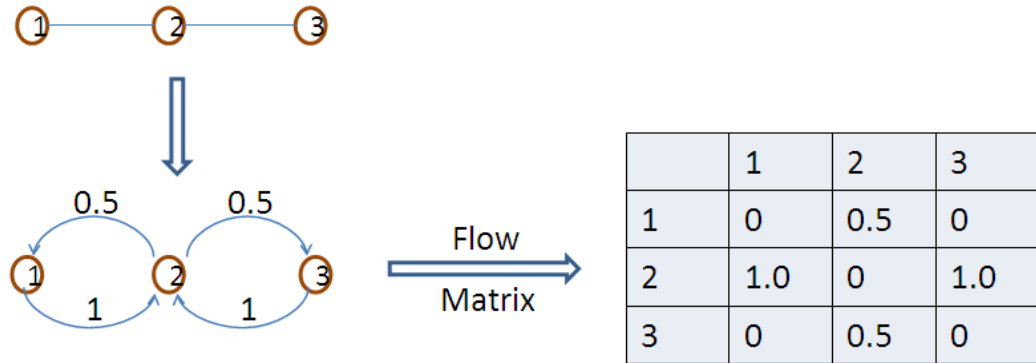


Figure 3.2 Representation of creation of flow matrix.

The random walks are calculated by Markov Chains that is why; the algorithm is named as Markov Cluster Algorithm (MCL). MCL simulated the random walks within a CN by two operations running sequentially: expansion and inflation. Expansion operation takes the powers p of the markov chain. The result of expansion is the matrix showing the probability that a random walker starts from node i and reaches node j in p steps. This is the responsible for allowing flow to connect different regions of the

network. Inflation operation is about adjusting the transition by columns. For each node, the transition values are changed so that strong nodes become strengthened and weak ones are demoted. This is done by raising a single column to a non-negative power and then by normalizing. The inflation parameter r adjusts the power of this raising and it controls the granularity of communities.

MCL algorithm starts with the input of two parameters for p and r respectively for expansion and inflation. Then it creates the markov matrix and normalizes it and starts to apply expansion and inflation operation with p and r parameters. These operations are repeated until steady state is reached and result matrix shows the communities. For n node network, matrix multiplication costs $O(n^3)$ and inflation is done in $O(n^2)$ time. This complexity can be improved by setting small values directly to zero. However, MCL does not work well for CN which has long diameter. Further, because using the original matrix only at start, neighboring information of the nodes wears out as climb on next iterations and algorithm does not penalize the diverge of flows between neighbors [43], consequently produces too many communities.

3.1.4.2 Latapy and Pons' Walktrap

Latapy and Pons announced another dynamic algorithm based on a distance measure calculated through random walk [10]. To this algorithm, random walks on a CN tend to get “trapped” into a community that is why; this is named as walktrap (WT). As MCL, WT also defines a random walk process such that at each time step, a walker is on a node and moves to another node which is neighbor to the current one uniformly and randomly. The sequence of the movement of walker is a markov chain. The transition probabilities from node i to node j are shown in the transition matrix P which can be defined as follow:

$$P_{ij} = \frac{A_{ij}}{d(i)} \quad (3.4)$$

Here, A_{ij} is the adjacency matrix of CN and $d(i)$ is the degree of node i . The t^{th} power of P, P_{ij}^t gives the random walk of length t . (1) When t tends to infinity, the probability of being any node different than starting node only depends on the final node's degree and (2) the probability of going from node i to node j and from node j to node i only depends on the degrees of those two nodes. By using (1) and (2), a distance measure r is developed for WT. One may juxtapose the property of r as below:

- If any two nodes i and j are in the same community, P_{ij}^t must be high but any high probability does not show that those two nodes are in the same community.
- The walker tends to go to the nodes with higher degree.
- If any two nodes are in the same community than they tend to visit same nodes during their walk. In another word, if the nodes i and j are in same community then one obtain probably $\forall k, P_{ik}^t \approx P_{jk}^t$.

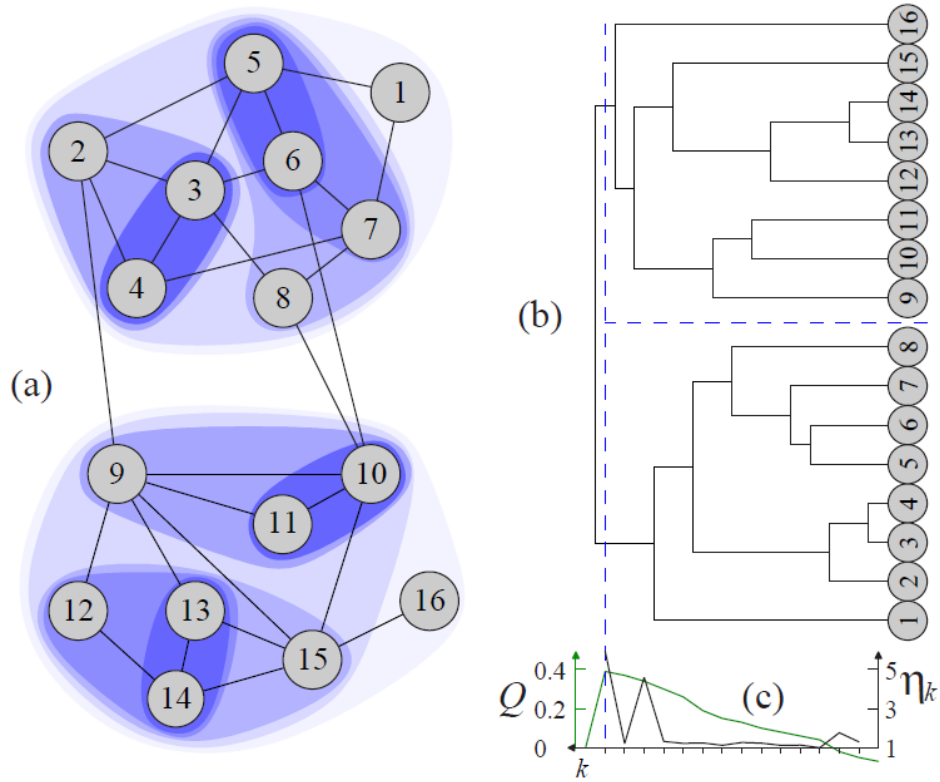


Figure 3.3 An example of WT execution and its result community structure.

After defining r , WT applies hierarchical clustering on CN and finds the communities. This starts with the state that every node is in own community and calculates the distance between them then merges the closest nodes. When WT merge the communities, this chooses only the communities which are adjacent in order to reduce the complexity and finally creates the dendrogram. We have shown an example of how WT finds the communities in a simple network in Figure 3.3 [10]. The image (a) is an example of community structure found by WT algorithm using random walks of length $t = 3$, (b) is the result dendrogram. The maximum of n_k and Q , plotted in (c), show that the best partition consists in two communities. The maximal values of n_k show also those communities of different scales may be relevant.

The total complexity of WT in time is $O(mnH)$ where m is link size, n is network size and H is the height of the dendrogram. For the sparse networks and small heighted dendrograms, the complexity becomes $O(n^2 \log n)$.

3.1.4.3 Rosvall and Bergstrom's Infomap

Rosvall and Bergstorm developed an approach that reduces finding community structure to solving a coding problem. The algorithm consist of mapping efficiently the whole network in order to address the regularities of it and random walk which uses all of the information in the network is used as a proxy for this mapping [14]. This algorithm is named infomap (INP).

Random walk in CN is a good option for describing its whole structure and finding the communities. The aim of INP is to assign the codes efficiently describing the paths on the network that arise from random walk process by not losing the information of the structure. Huffman coding [44] which gives shorter codes to common events or objects and longer codes to the rare ones allows compressing some information and saving the structure. INP uses this idea as shorter codes for the more frequent nodes on random walk and longer for the less frequent ones.

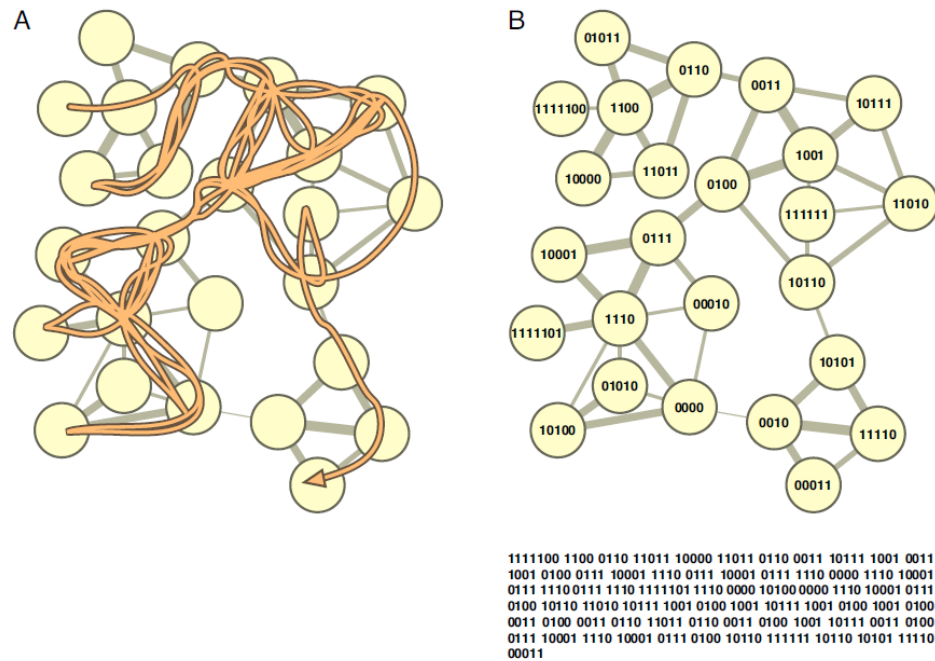


Figure 3.4 Detecting communities by compressing the description of information flows on networks.

Matching the nodes to the codewords which are assigned according to their frequency is an efficient coding but one needs to separate important structure from insignificant details in order to map. Thus, INP uses two level descriptions for network: (1) Code the important structures such that each one will have a unique name, (2) the objects inside the structure will also be coded uniquely but any two objects from different structures can have the same code. Obviously, for the random walk, these structures correspond to the communities in which the walker spends more time. Finding the community structure now corresponds to finding the efficient coding.

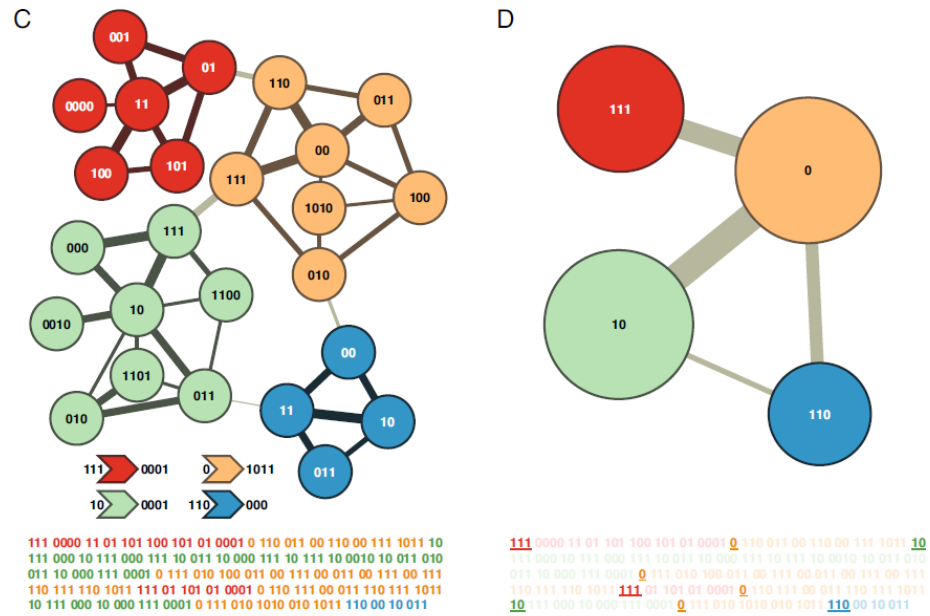


Figure 3.5 Detecting communities by compressing the description of information flows on networks.

The efficient coding is found by the entropy of the random walk inside and between communities. At first, the frequency of walker' visit to each node is calculated and possible results of community structures are found by greedy optimization. Then, the results are optimized by simulated annealing method.

INP does not use modularity. Thus, one avoids the drawbacks of modularity by using INP. However, its characteristic idea is based on information flow and that is why; INP cannot result as good as the modularity based approaches for the networks where links represent not flows as pair wise relationship networks.

A figurative representation of INP is on Figure 3.4 [14] and Figure 3.5 [14]. The image (A) on Figure 3.4 shows the description of the trajectory of a random walk on the network such that important structures have unique names. The orange line shows one sample trajectory. Image (B) shows a basic approach which is to give a unique name to every node in the network. The Huffman code illustrated here is an efficient way to do so. The 314 bits shown under the network describe the sample trajectory in (A), starting

with 1111100 for the first node on the walk in the upper left corner, 1100 for the second node, etc., and ending with 00011 for the last node on the walk in the lower right corner. The image (C) on Figure 3.5 shows a two-level description of the random walk, in which major clusters receive unique names, but the names of nodes within clusters are reused, yields on average a 32% shorter description for this network. The codes naming the modules and the codes used to indicate an exit from each module are shown to the left and the right of the arrows under the network, respectively. Using this code, we can describe the walk in A by the 243 bits shown under the network in C. The first three bits 111 indicate that the walk begins in the red module, the code 0000 specifies the first node on the walk, etc. And image (D) represents that reporting only the module names, and not the locations within the modules, provides an efficient coarse graining of the network.

3.1.5 Other Methods

There is a very wide range of interest to community detection problem in literature from various domains and this results many different type of algorithm. We have classified some of them in order to their characteristic. We have put other algorithms that we have used in the class of *Other Methods*. This class includes

3.1.5.1 Raghavan *et al.*'s Label Propagation

Raghavan *et al.* introduced an approach which benefit from the neighborhood of the nodes [45]. The basic idea is putting the node into the communities where most of its neighbors belong to. This algorithm labels every node uniquely at first and allows the propagation of label of the nodes to their neighbors. That is why; this is called label propagation (LP).

As we mentioned above, LP gives a unique label to every node initially and at each time step, each node takes the label shared by the majority of its neighbors. The algorithm can be summarized as following;

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 $C_x(t) = f\left(C_{x_{i1}}(t), \dots, C_{x_{im}}(t), C_{x_{i(m+1)}}(t-1), \dots, C_{x_{ik}}(t-1)\right)$. Here f returns the highest frequency among neighbors and ties are broken uniformly randomly.
5. If every node has a label that the maximum number of their neighbors has, then stop the algorithm. Else set $t = t + 1$ and go back to 3.

LP algorithm works asynchronously. This means, for current node which will be labeled at current iteration, LP takes into account the new labels of its neighbors which are already processed at current iteration and the old labels of the neighbors which haven't been processed yet. The propagation of labels in LP is represented by Figure 3.6 [45]. Due to a high density of links in the figure (highest possible in this case), all nodes acquire the same label.

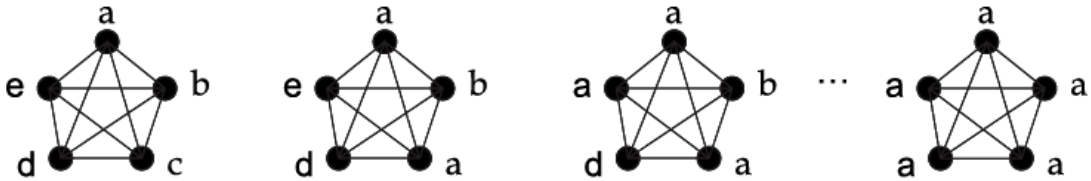


Figure 3.6 Nodes are updated one by one as we move from left to right.

LP is a fast algorithm which runs $O(m)$ in worst case. For sparse networks; this is $O(n)$. This does not require any parameter or *a priori* knowledge as community size or number and LP does not optimize any measure. But the algorithm does not produce any unique solution. Thus, one can aggregate the different results of LP for the same CN. Because this consumes less time than the other algorithms, LP can be very convenient for large networks.

3.1.5.2 Rosvall and Bergstrom's Infomod

Rosvall and Bergstrom approach the community detection problem with a different point of view. They have shown that finding appropriate community structure can be seen as a problem in data compression [46]. Rather than finding the subgroups which have denser inner links and sparser outer links, they have searched the regularities which can compress the whole network. This algorithm is called infomod (IND).

IND consists of a procedure of describing CN by a simplified structure of its modules as a communication process. At this process, the link structure of CN is a random variable X and the signaler carry reduced information of X to Y such that Y must possess more information about X as possible. Signaler sends the encoded message Y to the receiver. The receiver decodes this message to Z and makes guess about the structure of X . For the process of compression of X , IND uses mutual information between the description and network $I(X; Y)$.

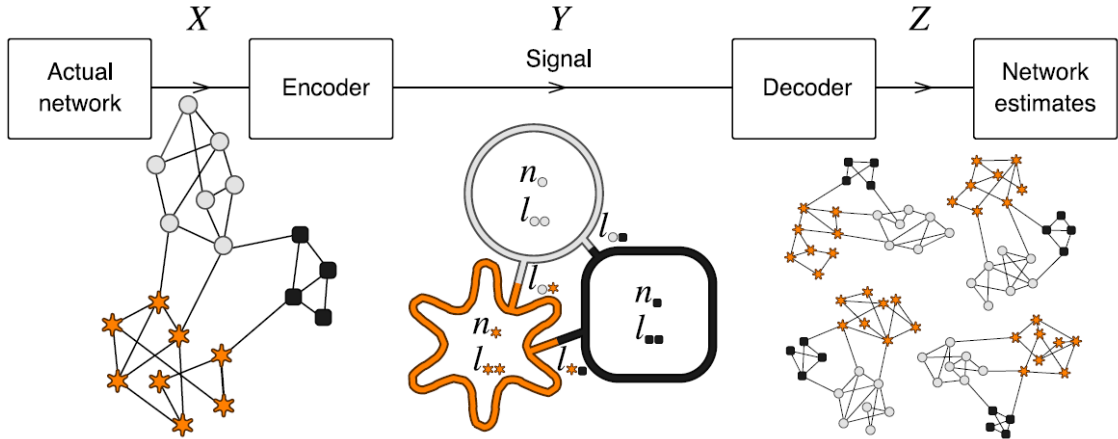


Figure 3.7 Basic framework for detecting communities as a communication process.

The encoding of Y is prepared by two items. First one is the information of the affected communities for the nodes, this is a sequence of length network size showing the

community numbers from 1 to c ; second item is the $c \times c$ community matrix describes how existent communities are linked to each other by showing the link numbers between the communities. To find the best assignment of the communities, the one which maximize the mutual information $I(X;Y) = H(X) - H(X|Y)$ over all possible assignments of c communities should be chosen. In another word, the second term of this equation should be minimized. Of course, $H(X|Y)$ is 0 when $Y=X$ which means every node is in their own community but this implies that one cannot do any compression on data. That is why; one has to choose an appropriate number for c . In IND algorithm, this is done by using Minimum Description Length (MDL) principle [47] and the optimization of $H(X|Y)$ is done by simulated annealing.

We show the schematic representation of IND algorithms on Figure 3.7 [46]. A signaler knows the full network structure and wants to send as much information as possible about the network to a receiver over a channel with limited capacity. The signaler therefore encodes the network into modules in a way that maximizes the amount of information about the original network. This figure illustrates an encoder that compresses the network into three modules, i = circle, square, star, with n_i nodes and l_{ii} links connected by l_{ij} links between the modules. The receiver can then decode the message and construct a set of possible candidates for the original network. The smaller the set of candidates, the more information the signaler has managed to transfer.

As INP, IND is an approach that does not use the modularity too. That is why; one does not come across the drawbacks of the modularity with this algorithm. IND results better for the CN which have different sized communities than the algorithms which optimizes the modularity. But because IND uses simulated annealing, it is slower. IND works well until CN with 10^4 nodes but this is not appropriate in terms of time complexity for larger networks.

3.2 Comparing Algorithms

3.2.1 Algorithms Properties

One can compare the algorithms according to many aspects such as the parameters that they require; *a priori* knowledge that they need, finding hierarchical communities or not or their time and memory cost. Detecting communities is a big and alive challenge. The algorithms which find hierarchical communities permit to determine the number of result communities. Table 3.1 Algorithms comparison shows some of the properties of the algorithms.

We focus on the partitioning performance of the algorithms. The algorithms performance differs to their community detection criteria and the properties of the networks. We compare the algorithms in order to observe how their performance differs depending on the realism level of the networks. We use artificial networks with real-world properties and *a priori* known community structure. Thus, we can use reference community structure to determine the performance of the algorithms.

As we stated before, the modularity measure is a standard for assessing the quality of a network partition. But it was designed to be an approximation of the partition quality, to guide community detection algorithms when the actual communities are unknown. The value computed for a given situation depends on both the quality of the detected communities and of the nature of the network community structure. This dependence on the network structure prevents from using modularity to compare algorithm performances on different networks. That is why; modularity is not an appropriate measure in our study. For interpretation purposes, we nevertheless processed the modularity for the various tests we performed (several tested algorithms use modularity during their processing). Instead of this, we use normalized mutual information measure which is explained in following part.

Table 3.1 Algorithms comparison

Name	Symbol	Hierarchical	Time Complexity	Required Parameters	Modularity Use
Edge Betweenness	EB	Yes	$O(n^3)$	None	Cut Selection
Radetal	RA	Yes	$O(n^2)$	Transitivity criteria	Cut Selection
Fast Greedy	FG	No	$O(n \log^2 n)$	None	Optimization
Louvain	LV	No	$O(n)$	None	Optimization
Spin glass	SG	No	$O((m + n)n)$	Spin number	Optimization
Leading Eigenvector	LEV	Yes	N/A	None	Cut Selection
Commfind	CF	No	N/A	Number of eigenvector	Cut Selection
Markov Clustering	MCL	No	$O(n^3)$	Expansion/inflation	Cut Selection
Walk trap	WT	Yes	$O(n^2 \log n)$	Random walk length	Cut Selection
Label Propagation	LP	No	$O(n)$	None	No
Infomod	IND	No	N/A	Community number	No
Infomap	INP	No	N/A	None	No

3.2.2 Normalized Mutual Information

Instead of modularity, we compared the performances using the normalized mutual information (NMI). This measure was defined in the context of conventional clustering

to compare two different partitions of one data set [2, 48] and was later used to assess community detection performance [7, 49-51].

The measure is derived from a confusion matrix whose element m_{ij} represents the number of nodes put in community i by the considered algorithm, whereas they actually belong to community j . This matrix is usually rectangular, because the algorithm does not necessary find the correct number of communities.

$$I = \frac{-2 \sum_i \sum_j m_{ij} \log (m_{ij} / m_{i+} m_{+j})}{\sum_i m_{i+} \log (m_{i+} / n) + \sum_j m_{+j} \log (m_{+j} / n)} \quad (3.5)$$

If the estimated communities correspond perfectly to reality, the measure takes the value 1, whereas it is 0 when the estimated communities are independent from the actual ones.

3.2.3 Artificially Generated Networks

Until recently, authors test their algorithms on a limited area which includes generally small sized, very common real-world networks like Zachary karate club, scientific collaboration network, college football or dolphin network [5, 31, 40, 45, 46]. Only few authors use large sized networks too like WWW or Belgian mobile phone network [32, 45]. However, although one may test an algorithm on a large sized CN, their community structure is unknown. Further, creating them is an expensive and difficult process and their properties are uncontrollable. The communities have been noted relatively, so, it is not objective.

In order to test the performance of the algorithms, one needs the networks with reference communities. In other words, in order to measure the performance, one must compare the real community structure with the result community structure of the

algorithm. In that point, we need artificial networks. Artificial networks have facilitated community detection algorithm's test. They have no theoretical size limit. Because they have been computer generated, one may control their properties and community structure with some parameters and creating them is an easier process than real ones. But, of course artificial networks cannot replace real ones. They are complementary to the weak points of real-world networks. Some authors have tested their algorithms with computer generated artificial networks which have *a priori* decided communities [31, 40].

4 RANDOM NETWORK MODELS

Since the beginning of the CN study, researchers tried to explain real-world CNs for some universal models. Because, every CN is different and it is difficult to find a unique model to them and the analysis is also subjective on a chosen network. Despite of their difference, the real-world networks have some similar properties which are explained in section 2.4. Random models which generate networks by some random process opened a door to represent real-world CNs. The process starts with n nodes and the links are added among those nodes randomly. In fact, this subject is tightly linked to graph theory and the works in this field started for a long time ago. In this section, we will present only the models linked to our work including the models generating networks with and without a community structure. At first, we will explain random network models which do not include any community structure. We will start with Erdős–Rényi model, because it is a reference to assess the realism of some properties such a small-world and transitivity. Then, we will explain configuration model, Barabási and Albert model and evolutionary preferential attachment model which we use in our experiments. At second, we will explain two community structured networks. First one is Girvan-Newman model which is the standard benchmark for comparing community detection algorithms. Second one is Lancichinetti *et al.* model which is a more recent alternative to Girvan-Newman. We use this model as the basis of our experiments.

4.1 Models without Community Structure

4.1.1 Erdős–Rényi Model

Many different random network models have been introduced but the first systematic and prominent study is created by Hungarian mathematicians Paul Erdős and Alfred

Rényi [52]. This model is called the *Erdős-Rényi random model* (ER), or sometimes even simply *random model*.

The creation procedure starts with n disconnected nodes. Then, links are added between the nodes until the link number reaches m . The probability of having a link between any two nodes is independent from each other. The result set is denoted as $G_{n,m}^{ER}$ but procedure results only one network which is one of the $C_{n(n-1)/2}^m$ equally possible networks. An alternative ER creation procedure is linking each couple of n nodes with a probability p . This method results networks with different number of links and result set is denoted as $G_{n,p}^{ER}$. Network with m links will appear in this set with a probability $p^m(1-p)^{(n(n-1)/2)-m}$ [52]. ER model does not allow multiple linking and the result network is not necessarily connected. These two methods have similarities. First one is applicable easily for generation process but the latter one is more appropriate for statistical generalization.

Many properties of $G_{n,p}^{ER}$ can be studied when the network size is very large. The limit of $n \rightarrow \infty$ is studied when average degree $\langle k \rangle$ is constant and equal to $p(n-1)$. In this case, its degree distribution is studied by [53]. The probability that having $k = k_i$ degree for a node i is the binomial distribution with parameters $n-1$ and p :

$$P(k = k_i) = C_{n-1}^k p^k (1-p)^{n-1-k} \quad (4.1)$$

Where, p^k is the probability of a node having degree k and C_{n-1}^k is the combination of different possibilities of choosing the end point of k links. Since all the nodes in ER are statistically equivalent, each of them has the same distribution, and the probability that a node chosen uniformly at random has degree k has the same form as $P(k = k_i)$. For large n , and fixed $\langle k \rangle$, the degree distribution is well approximated by a Poisson distribution:

$$P(k) = e^{-\langle k \rangle} \frac{\langle k \rangle^k}{k!} \quad (4.2)$$

This model is alternatively called the Poisson model or the Binomial model in some sources. It is also shown that $G_{n,p}^{ER}$ have $C = p = \langle k \rangle / n$ as transitivity coefficient. For large networks, transitivity is very weak. Average distance is $l = \ln(n) / \ln(\langle k \rangle)$.

ER model produces the networks with no degree correlation and small average distance with Poisson degree distribution and relative transitivity to the network size and having link probability. The connectedness also depends on the chosen probability. Thus, this model does not possess most real-world network properties.

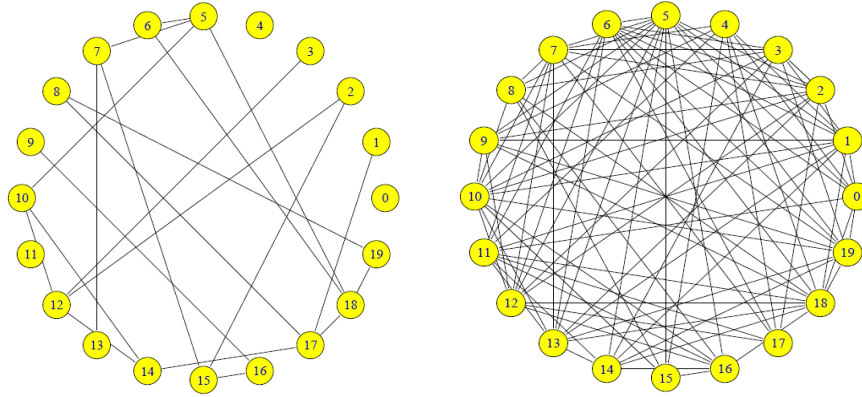


Figure 4.1 ER networks which are created with 20 nodes. The network (left) is created with $p = 1/10$ and the other (right) is created with $p = 1/2$.

4.1.2 Configuration Model

ER model covers some real-world CN properties but it is still weak for one of the most prominent property: scale-free degree distribution. Bender and Canfield [54] proposed configuration model (CM) which allows creating a random network with a given

degree sequence. Later, Molloy and Reed [55, 56] proposed a variation of CM which allows loops and multiple linking between nodes. CM takes the degree sequence and node number as an input but all other aspects are random. A degree sequence is any sequence of integer of length n which can be defined as $D = \{k_1, k_2, \dots, k_n\}$. The i^{th} member of D corresponds to the degree of the i^{th} node such that $\sum_i k_i = 2m$ where m is the total link number. The i^{th} node, i.e. i^{th} member of D , is linked to $k_i / 2$ other nodes. Of course, there are many choice of neighbor for each node. The result set of CM which includes all those possibilities with equal probability is denoted as $G_{D,n}^{CM}$ and the result network is chosen uniformly randomly from this set. Each configuration can be obtained in $\prod_i k_i!$ different ways. Obviously, $G_{n,p}^{ER}$ can be obtained as a special case of $G_{D,n}^{CM}$. CM can produce scale-free networks with a D which is created by a power-law distribution with $P(k) \sim k^{-\gamma}$. One can study the properties of result network when γ decreases from ∞ to 0. In this case, average degree and maximum degree of the network can be defined as $\langle k \rangle \sim k_{max}^{-\gamma+2}$ [11] and $k_{max} \sim n^{1/\gamma-1}$ [1]. If we assume that all the nodes of network can be reachable in l step by starting from a given node, the total number of $1^{st}, 2^{nd}, \dots, l^{th}$ neighbors of that node and itself gives total node number, $1 + \sum_{m=1}^l z_m = n$ where z_m is the m^{th} level neighbor which can be reachable from current node with a path of length m . The average distance of CM is $l = (\ln(n/z_1)/\ln(z_2/z_1)) + 1$. The general transitivity coefficient of CM is $C = 1/nz_1 [z_2/z_1]^2$ and in the case of a power-law distribution, $C \sim n^{-3\gamma-7/\gamma-1}$. For smaller γ values transitivity coefficient becomes considerable. Besides, CM is a model which produces networks with no degree correlation. It provides some other real-world networks properties although it is created randomly. For example, the result network can be transitive depending on degree exponent of power-law. Both ER and CM networks have smaller average path length than real-world CN.

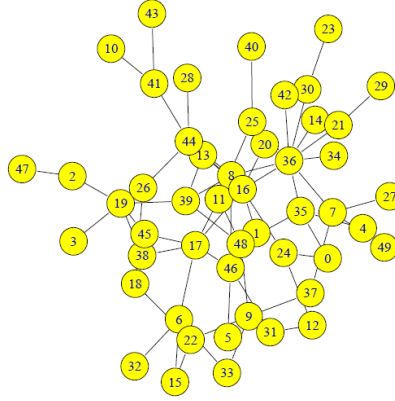


Figure 4.2 A network created by CM with $n = 50$. Degree sequence is created by exponential degree distribution.

4.1.3 Barabási-Albert Model

Another scale-free network model which is inspired by the dynamism of real-world CNs is proposed by Barabási and Albert [57]. This model is called Barabási model (BA). BA wants to imitate the process of real-world scale-free networks build. The difference with the two previous models is the way of building this model is supposed to be realistic.

BA has two creation steps: growth and preferential attachment. Growth starts from an initial network containing m_0 connected nodes; a realistic iterative process is applied to simulate growth. At each iteration, one node is added to the network, and is randomly connected to m existing nodes ($m \leq m_0$). Preferential attachment mechanism selects these m nodes with a probability which is a function of their current degree k : $P(k_i) = k_i / \sum_j k_j$. In other terms: the higher a node degree, the higher its chances of being selected. This results in a power-law degree distribution, since degree increases faster for nodes with higher degree, as new nodes are added to the network. All possible results of BA can be denoted as $G_{n,k}^{BA}$. The properties of the BA model can be analyzed for very large networks. In this case, The average degree depends directly on the m parameters: $\langle k \rangle = 2m$ [1]. Average distance is defined $l \sim \ln(n) / \ln \ln(n)$ [58]. The

average distance is always less than in same-sized ER networks, so it has the small world property [11]. The exponent cannot be controlled though, and tends towards 3 [57]. The transitivity coefficient is greater than for the ER networks, but nevertheless decreases with network size following a power-law $\sim n^{-0.75}$ [11]. Moreover BA creates networks with non-trivial degree correlated networks.

BA is a powerful model for describing the scale-free behavior of the networks because it models real-world networks growth and so, it has many realistic properties. But, it creates networks with constant exponent. In large real-world networks, the power-law exponent generally varies between 2 and 3. For different type of networks, we can obtain different exponent values. To take into account this change of exponent values, many different variation of the BA have been proposed [13, 59-61].

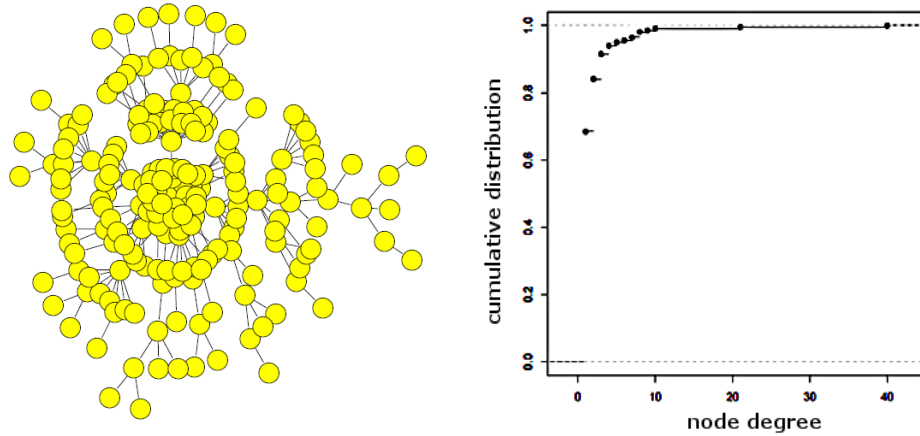


Figure 4.3 A network created by BA model with $n = 200$ (left), cumulative distribution of degree of same network (right).

4.1.4 Evolutionary Preferential Attachment Model

The evolutionary preferential attachment (EV) model [61] is a variant of the BA model. It also uses the preferential attachment and growth mechanisms, except the attachment

probabilities are not based on some topological properties, like the current degree in the case of BA, but on some nodal dynamic property. EV is inspired the connecting properties of nodes in social networks. According to this, the nodes are cooperative or defective individuals involved in a social dilemma [62]. EV creates initially a fully connected network with m_0 nodes like BA model. Then, at each iteration, it links the new coming node to m existing nodes preferentially chosen by their evolutionary fitness value. The evolutionary fitness of nodes is explained by inspiring Prisoners Dilemma game. There are three possible situations for each link in the network: (1) if two cooperators meet both receive R , when (2) two defectors play both receive P , while (3) if a cooperator and a defector compete the former receives S and the latter obtains T . The four payoffs are ordered as $T = b > R = 1 > P = S = 0$. After playing, every node compares its evolutionary fitness (payoff) with that corresponding to a randomly chosen neighbor. The payoff is summarized in Table 4.1:

Table 4.1 Payoff of EV model

	C	D
C	1 (R)	0 (S)
D	b (T)	0 (P)

Every little iteration, each node plays either cooperation or defection against all its neighbors. It gets a total fitness depending on the individual results: 0 for bilateral defection or unilateral cooperation (S or P), 1 for bilateral cooperation (R), and b for unilateral cooperation (T), with $b > 1$. The first move is randomly chosen, whereas the next one depends on the respective results of the considered node and a randomly picked neighbor. If the neighbor's fitness is better, the node might switch to its strategy, with a probability depending on the difference between their fitness. Nodes with higher fitness are more attractive to a node added to the network, because by being connected to them, it may use a strategy which proved to be successful.

According to its authors, this process is more realistic and leads to networks with high transitivity and degree correlation. Transitivity coefficient is defined as $C \sim k^{-b}$. Besides the parameters already needed by BA (n, m_0, m), EV uses b (points scored for unilateral cooperation) and ε (selection pressure). The latter allows modulating the influence of the preferential attachment mechanism: all nodes are equally probable when $\varepsilon = 0$, whereas the nodes scores are fully considered for $\varepsilon = 1$.

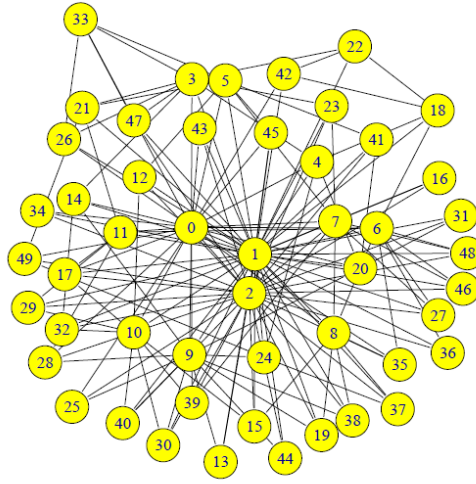


Figure 4.4 A network created by EV model. $n = 50, b = 1.5, \varepsilon = 0.99$

4.2 Models with Community Structure

Artificial network models evolved in a way to generate artificial community structures. Community structure can be drawn with different methods. One of the recent methods is using a rewiring process which consists changing the start or end points of the links by not touching to link size or degree distribution. Following part contains two important approaches of artificial network creators with artificial community structure.

4.2.1 Girvan-Newman Model

Girvan and Newman produced their artificial network set (GN) whose creation procedure is simple in order to test their EB algorithm [5]. It starts with 128 unconnected nodes which are assigned to four communities such that each of them includes 32 nodes. Then, links are placed between the nodes with a probability p_{in} if the nodes are in same community and p_{out} if not. These probabilities are chosen in a way of keeping an average degree close to 16 and $p_{out} < p_{in}$. GN produces network with known community structure but all other respects are random.

GN model is used by many authors to test their community detection algorithms [5, 40, 46, 63]. There are some drawbacks of this model. Firstly, all nodes have same degrees, community sizes are all same and network has very small by the side of real-world large CN. However, real-world networks have heterogeneous degree distribution which is mostly power-law distribution; their community size also follows power-law distribution and most of real-world networks size is in general larger than GN benchmark. A reliable benchmark should cover those real-world network properties. Most community detection algorithms perform very well on GN benchmark because of its structures simplicity. This makes to compare the algorithms harder. GN benchmark does not permit the appearance of the algorithms limits. Figure 4.5 [5] shows an example of GN model generated network. In this network four communities are represented by different colors.

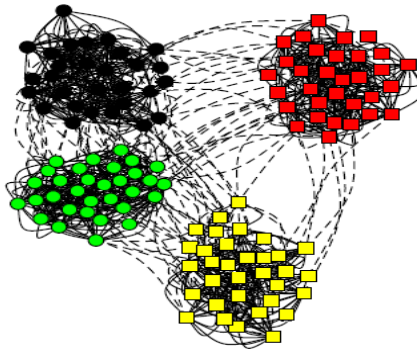


Figure 4.5 A schematic representation of a GN model generated network.

4.2.2 Lancichinetti *et al.* Model

To correct the drawbacks of GN, Lancichinetti *et al.* have proposed a model which randomly generate undirected and unweighted networks with mutually exclusive communities [7]. This model is named Lancichinetti, Fortunato and Radicchi model (LFR). Nodes degrees and community sizes are both distributed according to a power-law. The method was subsequently extended to generate weighted and/or directed networks, with possibly overlapping communities [50, 51], but here we focus on the first version.

This method allows controlling directly the following parameters: number of nodes n , desired average $\langle k \rangle$ and maximum k_{max} degrees, exponent γ for the degree distribution, exponent β for the community size distribution, and mixing coefficient μ . The latter represents the desired average proportion of links between a node and nodes located outside its community, called inter-community links. Consequently, the proportion of intra-community links is $1 - \mu$. It is generally not possible to meet this constraint exactly, and the mixing coefficient is therefore only approximated in practice. It is an important parameter, because it determines how clearly the communities are defined in terms of structure. For small μ values, the communities are distinctly separated, whereas for high values, the network has almost no community structure, making community identification a difficult task. Intuitively, one may think the community structure remains clear for $\mu < 0.5$, because this value corresponds to a network where, in average, nodes have the same number of inter- and intra-community links. In practice though, some algorithms are able to successfully detect communities even when μ approaches 0.8 [8, 51]. Let us note p_{in} (resp. p_{out}) the ratio of existing to possible intra- (resp. inter-) community links, or in other terms: the probability to have a link between two nodes belonging to the same community (resp. different communities). Then the communities are well-defined when $p_{in} > p_{out}$, which translates to equation (4.3).

$$\mu < \frac{(n - n_c^{max})}{n} \quad (4.3)$$

Where n and n_c^{max} are the number of nodes and in the biggest community, respectively [51]. The LFR method is three-stepped. First, it uses the CM to generate a network with average degree $\langle k \rangle$, maximum degree k_{max} and power-law degree distribution with exponent γ . Second, virtual communities are defined, so that their sizes follow a power-law distribution with exponent β . Third, an iterative process takes place to rewire certain links, so that μ is approximated, but without changing the degree distribution.

By construction, the LFR method guaranties to obtain several realistic properties: size of the network, power-law distributed degrees and community sizes. Moreover, some parameters give the user a direct control on these properties: network size (n), degree distribution ($\gamma, k_{max}, \langle k \rangle$), community structure (β, μ). For these reason, we call them controlled properties. However, it does not allow controlling other properties like small world, transitivity or degree correlation. We will examine how those uncontrolled properties change with those parameters in detail in following part.

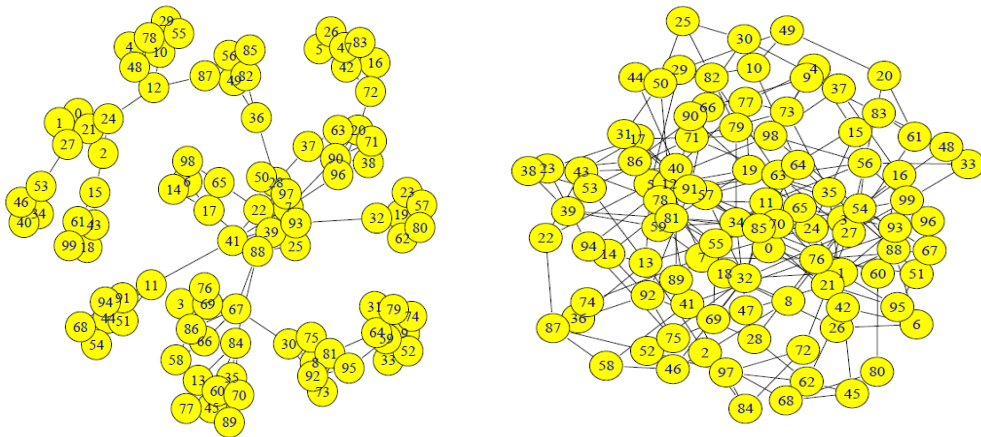


Figure 4.6 Two networks created by LFR model. Creation parameters; $n = 100, \langle k \rangle = 5, k_{max} = 15, \gamma = 3, \beta = 2$. The network (left) is created with $\mu = 0.05$ and the other (right) is created with $\mu = 0.95$.

4.3 Summary

We have indicated the parameters and the roles of those parameters of each generation model without community structure in Table 4.2 and with community structure in Table 4.3.

Table 4.2 Summary of the generation models without community structure

NAME	REQUIRED PARAMETERS	PARAMETER ROLE
Erdős–Rényi	Node number (n)	Network size
	Probability of having a link (p)	Network density
	Link number on each iteration (m)	Network density
Configuration (CM)	Node number (n)	Network size
	Degree sequence (D)	Degree distribution
Barabási-Albert (BA)	Node number (n)	Network size
	Initial link number (m_0)	-
	Link number on each iteration (m)	Average degree
Evolutionary Preferential Attachment (EV)	Node number (n)	Network size
	Initial link number (m_0)	-
	Link number on each iteration (m)	Average degree
	Level of cooperation in Prisoners game (b)	Transitivity control
	Selection pressure (ε)	Prisoners game effect

Table 4.3 Summary of the generation model with community structure

NAME	REQUIRED PARAMETERS	PARAMETER ROLE
Girvan-Newman (GN)	Probability of being in same community (p_{in})	Community sharpness
	Probability of being in different community (p_{out})	Community sharpness
Lancichinetti <i>et al.</i> (LFR)	Node number (n)	Network size
	Average degree ($\langle k \rangle$)	Average degree
	Maximum degree (k_{max})	Maximum degree
	Degree distribution exponent (γ)	Power-law control
	Community size distribution exponent (β)	Community size power-law control
	Mixing coefficient (μ)	Community sharpness

5 GENERATION OF REALISTIC NETWORKS

We have explained artificial network generation models in two different classes: with and without community structure at previous part. In the context of this work, the purpose of network generation is to compare algorithms which completely rely on the networks structure to identify communities. That is why; we have chosen the most recent and realistic network model with community structure which is LFR model. We examine the realism level of the networks which are generated by LFR-CM model and its effect on community detection algorithms. We assign the realism level of the model to the properties corresponds to the most popular ones, as presented in section 2.4 .The properties of LFR-CM change with its generation parameters. This is an important matter, because such a change may influence the algorithms performances, which could therefore be explained either by a direct or an indirect effect. By direct effect, we mean the observed performance modifications are related to the changed controlled properties. By indirect effect, we mean they are related to a change in some uncontrolled properties, caused itself by the change in controlled properties. In the following part, we study the influence of the parameters on the uncontrolled properties and show it is significant. We then propose a modification of the LFR-CM approach to tackle this problem, and study its properties.

5.1 Original LFR

We generated benchmark collection of networks using the original LFR, which we call LFR-CM since it is based on the CM. We selected our parameters values based on previous experiments in artificial networks generation [7, 8] and descriptions of real-world world networks measurements from the literature [1, 11, 16, 64]. The values are indicated on Table 5.1.

Table 5.1 LFR-CM parameter values

PARAMETER	VALUE
Node Number (n)	{100,500,1000,5000}
Average Degree ($\langle k \rangle$)	{5,15,30}
Maximum Degree (k_{max})	$3 * \langle k \rangle$
Degree Distribution exponent (γ)	{2, 3}
Community Size Distribution Exponent (β)	{1, 2}
Mixing Coefficient (μ)	[0.05; 0.95] with a 0.05 step

Table 5.2 Pearson Correlation of Parameters on Properties for $n = 1000$

PARAMETER	PROPERTY NAME	PEARSON CORRELATION VALUE
$\langle k \rangle$	Degree Correlation	-0.170
	Average Distance	-0.650
	Transitivity	0.090
γ	Degree Correlation	-0.200
	Average Distance	0.080
	Transitivity	-0.060
β	Degree Correlation	0.280
	Average Distance	0.009
	Transitivity	0.050
μ	Degree Correlation	-0.500
	Average Distance	-0.340
	Transitivity	-0.910

We produced 25 networks for each combination of parameters, and averaged the measured properties. We have observed the effect of network generation parameters on uncontrolled properties such that degree correlation, transitivity and average degree. Table 5.2 shows Pearson's correlation values of the parameters on properties. To this table, $\langle k \rangle$ has little effects on degree correlation and transitivity but it has a strong

effect on average distance, γ and β has little effects on all properties and μ has considerable effect on all properties. It has especially a very strong effect on transitivity.

We will explain the effect of $\langle k \rangle$ on average distance and μ on average distance, degree correlation and transitivity in detail in following parts. We also explain the change of properties with network size which has a significant effect on properties in following part.

5.1.1 Maximal and Average Degrees

The variations in the average and maximal degrees have little or no effect on the degree correlation and transitivity coefficient ($\sigma = -0.17$ and $\sigma = 0.09$, respectively), but there is a direct relation with the average distance ($\sigma = -0.65$). Unsurprisingly, it decreases dramatically when the average degree increases, certainly due to the rise in the number of links. This change becomes more significant for low μ values and it is more obvious for higher network size. As we see on Figure 5.1, for $n = 100, \gamma = 3, \beta = 2$ and $\mu = 0.5$, average distance is approximately 3.5 when $\langle k \rangle = 5$ and this value decreases until it becomes less than 2 for higher $\langle k \rangle$. But when we increase network size to 500, 1000 and 5000, we obtain respectively approximately 4.5, 5 and 6 for low average degree and almost 2 for high average degree. This result is valid for any other γ and β values but as we mentioned before, for lower μ values, the average distance becomes higher. In fact it is not a surprise because the average distance is related to the link proportion, in other words density, in the network. For higher $\langle k \rangle$, this proportion increases and average distance decrease.

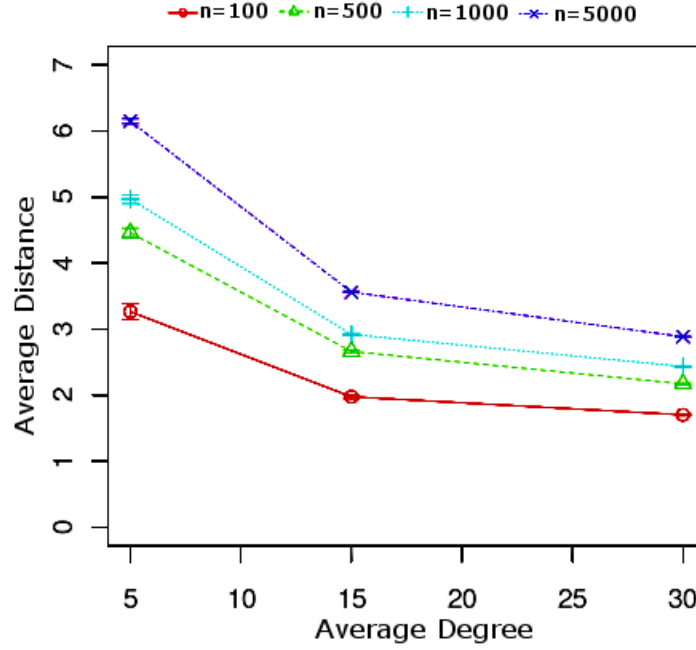


Figure 5.1 The change of average distance with average degrees for all network sizes in LFR-CM model. Here $\gamma = 3$, $\beta = 2$ and $\mu = 0.5$.

5.1.2 Network Size

Network size is another effective parameter on uncontrolled properties. As it is known, real-world CNs usually have very large size like millions of nodes and links. But, here, we only can compare small and medium sized networks because of computational limitations. We observed that we can obtain more stable results for higher sizes for all three properties. When network size is about 100 or 500, all properties are more sensitive to average degree or mixing coefficient, because the link proportions and positions totally depend on those parameters. However, when we increase the size by fixing other parameters, we obtain more stable results. As we see on Figure 5.2, while average distance value is about 2 for network size 100, it increases to 3 for network size 5000 with $\gamma = 3$, $\beta = 2$, $\langle k \rangle = 30$ and $\mu = 0.5$. Average distance increases non linearly with network size, which means the network has the small world property [1].

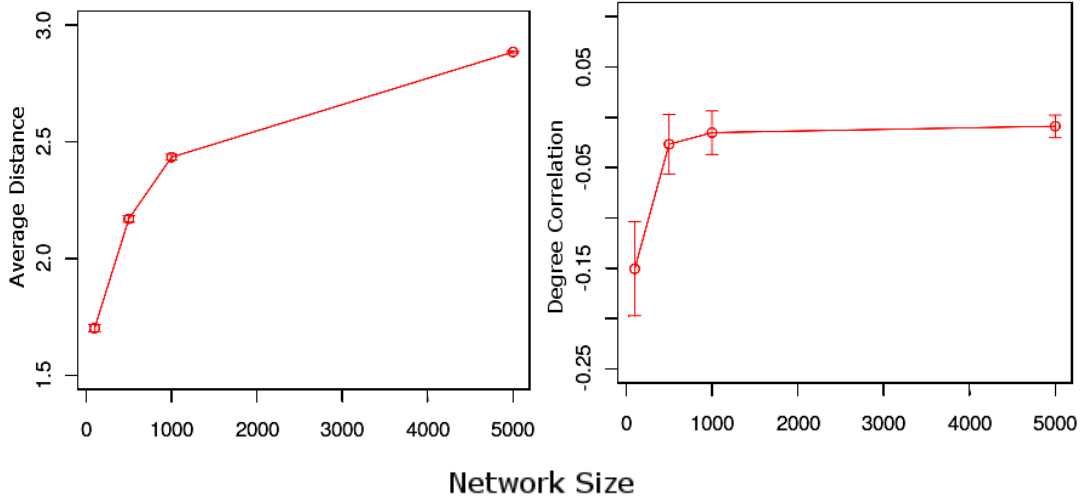


Figure 5.2 Average distance (left) and degree correlation (right) changes with network sizes in LFR-CM model. Here $\gamma = 3$, $\beta = 2$, $\langle k \rangle = 30$ and $\mu = 0.5$.

We observe on Figure 5.2 that degree correlation stays stable for network size 500, 1000 and 5000 at value 0 but it has a negative value for size 100 when using same values for the other parameters. This behavior changes depending on average degree and μ as being more stable for any network size for higher μ values. Another remarkable point is when $\mu < 0.25$, standard deviation of iterations are noticeable for degree correlation. But in any case, LFR-CM model does not create networks with a realistic degree correlation.

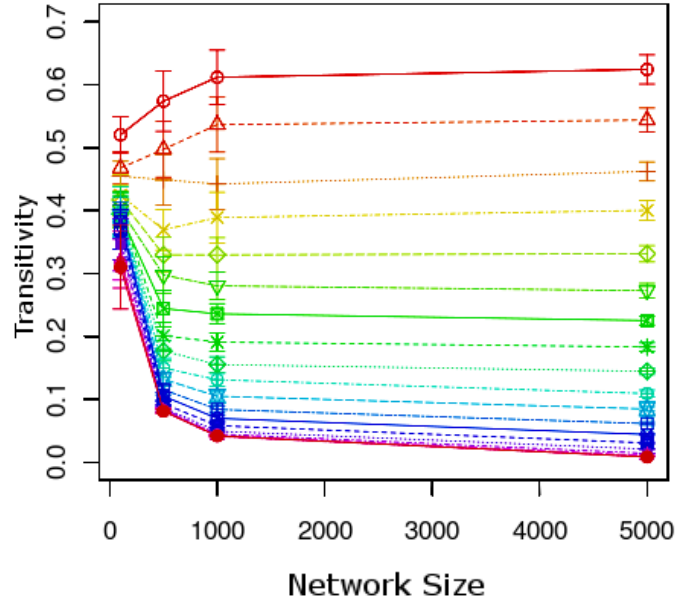


Figure 5.3 The change of transitivity with network sizes in LFR-CM model. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$ and from top to bottom each line corresponds different μ starting with 0.05 and ending 0.95 with 0.05 step.

As shown in Figure 5.3, transitivity undergoes a significant change. The trend of this change depends on μ value. When $\mu < 0.15$, transitivity increases non linearly. But after that μ , it decrease also non linearly. As we see on Figure 5.3 for $\mu = 0.05$, it increases from 0.5 to over 0.6 with network size increase. But it decreases from 0.45 to 0.1 for $\mu = 0.5$. When $\mu > 0.7$, the transitivity become about 0 for $n = 5000$. It is not a surprise because, as μ increase, the link between communities increases. Thus, it gets harder to have a link between any dyad having common neighbors. However, this is more probable when the communities are distinctive and have denser links inside.

5.1.3 Mixing Coefficient

The most influent parameter is μ . The computed correlations are not necessarily high, but the plots show a non-linear relationship between μ and all three uncontrolled properties. The average distance decreases when μ increases. We performed additional measurements on networks with sizes between 100 and 100000 nodes, and observed a

clear logarithmic relationship between the size and the average distance, which is consistent with real-world networks features. The transitivity is very high for low μ values, but gets down to the level of Poisson random networks when μ reaches 0.7. In the same way, for low μ values, the degree correlation is relatively high, but quickly decreases until μ reaches 0.4 or 0.5, and then stays close to zero. Interestingly, $\mu = 0.5$ corresponds to a limit above which the proportion of external links is higher than the proportion of internal links. Here, we must recall Lancichinetti *et al.* method consists in using the configuration model to generate a scale-free network, which is then partially rewired to create a community structure. For a given node, there are usually many more nodes outside than inside its community before the rewiring step. Therefore, the higher μ and the lesser the original network is modified. Put differently: in average, when μ grows, the generated networks get more similar to scale-free networks generated by the configuration model.

The configuration model is known to produce networks with no degree correlation [65]. Furthermore, Newman [1] showed that when it is used to generate scale-free networks, for $\gamma > 7/3$ the transitivity tends toward zero as the number of nodes is increasing. Our measures show close to zero degree correlation and transitivity when μ gets close to 1, which is consistent with the previous remarks. The average distance is also close to what is expected from network produced with a configuration model [66]. Using smaller μ values, i.e. defining more distinct communities, makes all three properties grow. The effect on degree correlation could be due to the apparition of hub-to-hub links between communities. The definition of community used here relies on stronger inner density, and is therefore related to the concept of transitivity, which may explain its increase. The disappearance of shortcut links between the communities could explain the observed decrease in average distance.

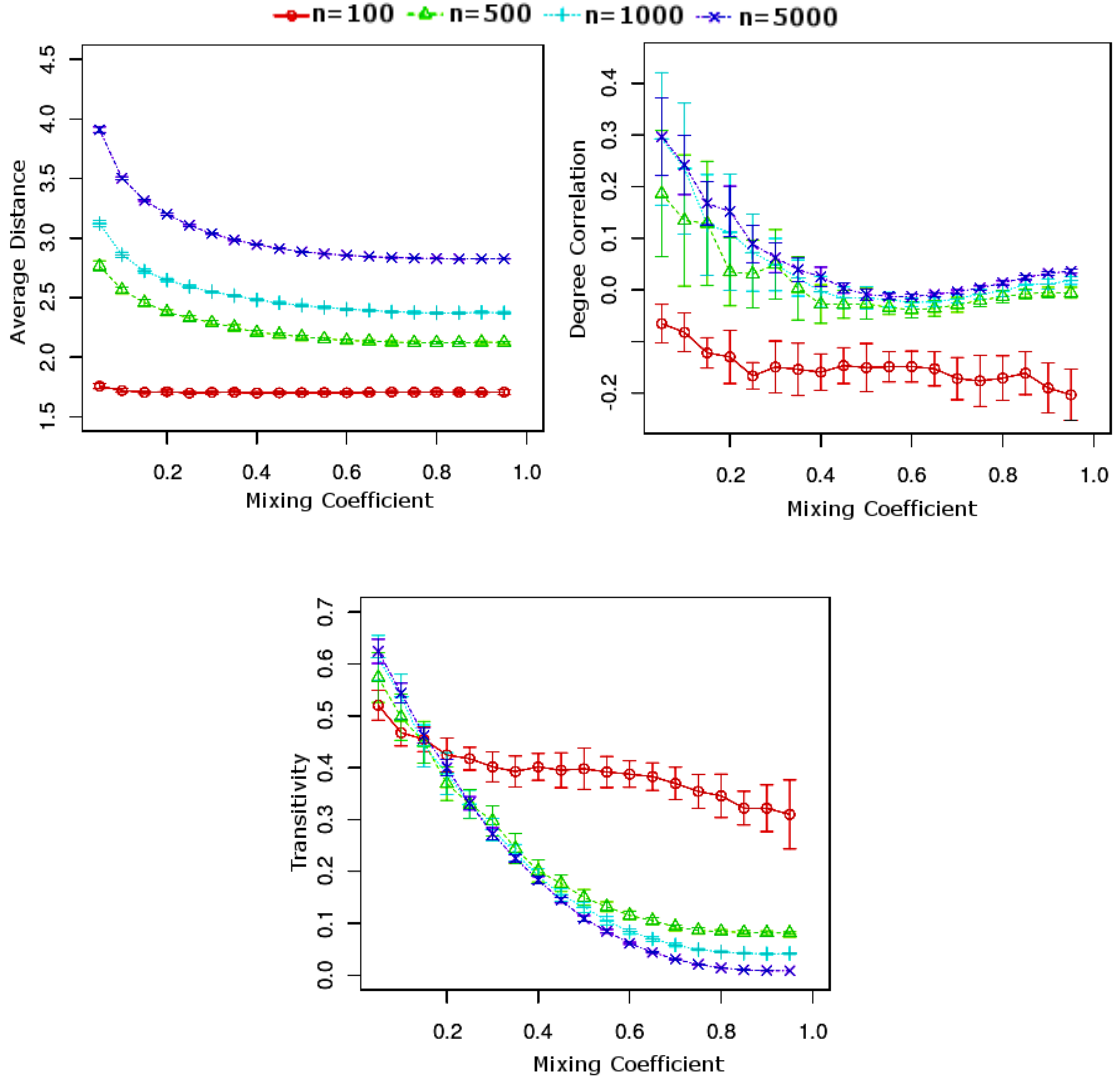


Figure 5.4 The change of average distance (top left), degree correlation (top right) and transitivity (bottom center) with mixing coefficient on all network sizes in LFR-CM model. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$.

As we see on Figure 5.4, the average distance's non linear decrease is more evident for low μ values and higher n . One may say, when $n = 100$, degree correlation trend is incompatible with other network sizes. Because, generated networks are inconstant at this network size. We see the same incompatibility with transitivity. While it decreases non linearly for $n = 500, 1000$ and 5000 , it is not an exact non linear decrease for $n = 100$ and for high values of μ , lower network sized networks becomes more transitive.

5.1.4 Other Parameters

The remaining parameters effect is weaker than the previously mentioned ones but they nonetheless affect some properties. β has little or no effect on the average distance and the transitivity coefficient, but it relatively affects the degree correlation. The β parameter controls the homogeneity of the community sizes: when it increases, the communities tend to be more uniform in terms of size [7]. Our interpretation is that with a small β , we have many small communities with no hubs, much less medium communities with a few hubs, and a few big communities with more hubs. Medium community hubs have less chance to get linked with other hubs, because there are only a few hubs in their community, and links between communities are rarer, which prevents them to get linked with hubs in other communities. When β increases, this chance also increases because the number of hubs in the same community gets larger.

The γ parameter has little or no effect on average distance and transitivity but it relatively affects the degree correlation. When γ increases, the network degree distribution becomes more homogeneous, so this is consistent with the fact that degree correlation is close to zero in Poisson random networks.

5.1.5 Conclusion

To conclude these observations, we can state the generated networks show some reasonably realistic properties when μ is relatively small. However, increasing this parameter not only causes communities to become less distinct, but also makes the whole network becoming less realistic, its average distance, transitivity and degree correlation decreasing rapidly. We also remark that network size has an effect on all three properties: they become more consistent and considerable for higher network size. But, when network size is increased, LFR-CM produces networks which are not transitive and have no degree correlation. One should also state that the link proportion affects the stability of the properties. We observe better results for higher degrees. Briefly, LFR-CM networks are realistic when μ is low, but become unrealistic for

higher μ . Moreover, the properties realistic level depends also on network size and average degree.

5.2 Modified LFR

5.2.1 Proposed Modification

One of the possible causes for the unrealistic properties observed in the networks generated with the original LFR approach, is the use of the CM [67] to generate the initial network during the first step. On the one hand, the CM is very flexible in the sense it is able to produce networks with any size and degree distribution, but on the other hand it is known these networks have zero degree correlation [65] and low transitivity when degrees are power-law distributed [1]. We propose to replace it with a different generative model, with more realistic properties. We considered the Barabási–Albert preferential attachment model [11] and the evolutionary preferential attachment [61], presented in the previous section. The rest of the method is not modified: community sizes are still drawn from a power-law distribution, and the rewiring process must be applied to make the community structure appear. We call the new generators LFR-BA and LFR-EV, respectively.

The alternative models do not allow as much control as the CM, and we had to adjust their parameters so that the resulting networks had approximately the same degree-related properties. Preferential attachment does not give any control on γ , which tends towards 3 by construction. To control the average degree, we used $m = 2, 7, 15$ for both LFR-BA and LFR-EV. The maximal degree is not controlled, but the values measured in the resulting networks are of the same order than the values specified for LFR-CM. LFR-EV additionally allows controlling transitivity, and we found out score $b=1.5$ and selection pressure $\varepsilon=0.99$ gave the best results. By using this ε value, we apply completely evolutionary criteria [61] to the network. We use $\beta = 2$ and the same values of μ with LFR-CM. We produced 25 networks for each combination of parameters, and averaged the measured properties. One can find the correlation of the

average degree and mixing coefficient parameters on properties in Table 5.3. To these values; μ has strong effect on modified models too and $\langle k \rangle$ seems more effective on degree correlation and transitivity than LFR-CM networks. But, we cannot observe these effects on the plots of degree correlation and transitivity with $\langle k \rangle$ change.

Table 5.3 Pearson Correlation values of parameters on properties of LFR-BA and LFR-EV networks for $n = 1000$

PARAMETER	PROPERTY NAME	LFR-BA CORRELATION VALUE	LFR-EV CORRELATION VALUE
$\langle k \rangle$	Degree Correlation	0.36	0.14
	Average Distance	-0.64	-0.68
	Transitivity	0.19	0.24
μ	Degree Correlation	-0.28	-0.75
	Average Distance	-0.32	-0.38
	Transitivity	-0.88	-0.91

We will explain the effect of $\langle k \rangle$ on average distance and μ on average distance, degree correlation and transitivity in detail by comparing LFR-CM, LFR-BA and LFR-EV models in following parts. We also explain the change of properties with network size which has a significant effect on properties in following part.

5.2.2 Average Degree

Unsurprisingly, average degree has a strong effect on average distance for both models. Average distance decreases with average degree increase. The behavior is the same as for the LFR-CM, but the average distance value depends on the considered model. It ranges from 4 to 6 for LFR-BA for $\langle k \rangle = 5$ with different network sizes and becomes almost 2 when $\langle k \rangle = 30$ as shown on Figure 5.5.

LFR-EV can produce networks with shorter average distance for low average degree. Its value is between 2 and 4 for $\langle k \rangle = 5$ with different network sizes and becomes

almost 2 when $\langle k \rangle = 30$. The average distance changes in a similar way for all three models. The values of LFR-EV are less than LFR-BA and LFR-CM while they are both similar.

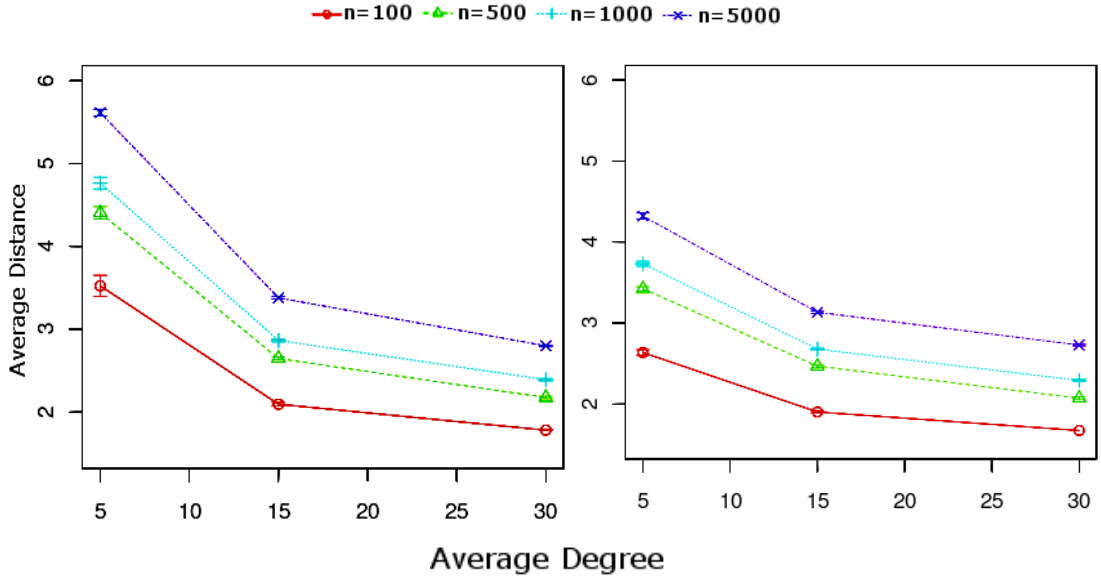


Figure 5.5 The change of average distance with average degrees for all network sizes in LFR-BA (left) and LFR-EV (right) model. Here $\beta = 2$ and $\mu = 0.5$.

5.2.3 Network Size

Network size is also one of the effective parameters on properties. We observe the same trend for each property on both LFR-BA and LFR-EV models. Average distance increases non linearly with the network size. The values and inflection points depends on the average degree and μ . For example for lower average degree and lower μ , the increase is more distinctive than higher ones with network size increase. The numerical values are also similar for both models. It stays stable after network size 1000. Average distance may still be stable for higher network sizes, but there is a range from 100 to 1000. For $n = 100$, its value is about 2 while it is 3 for $n = 5000$ as shown in Figure 5.6 and Figure 5.7. Those values still signify small average distance.

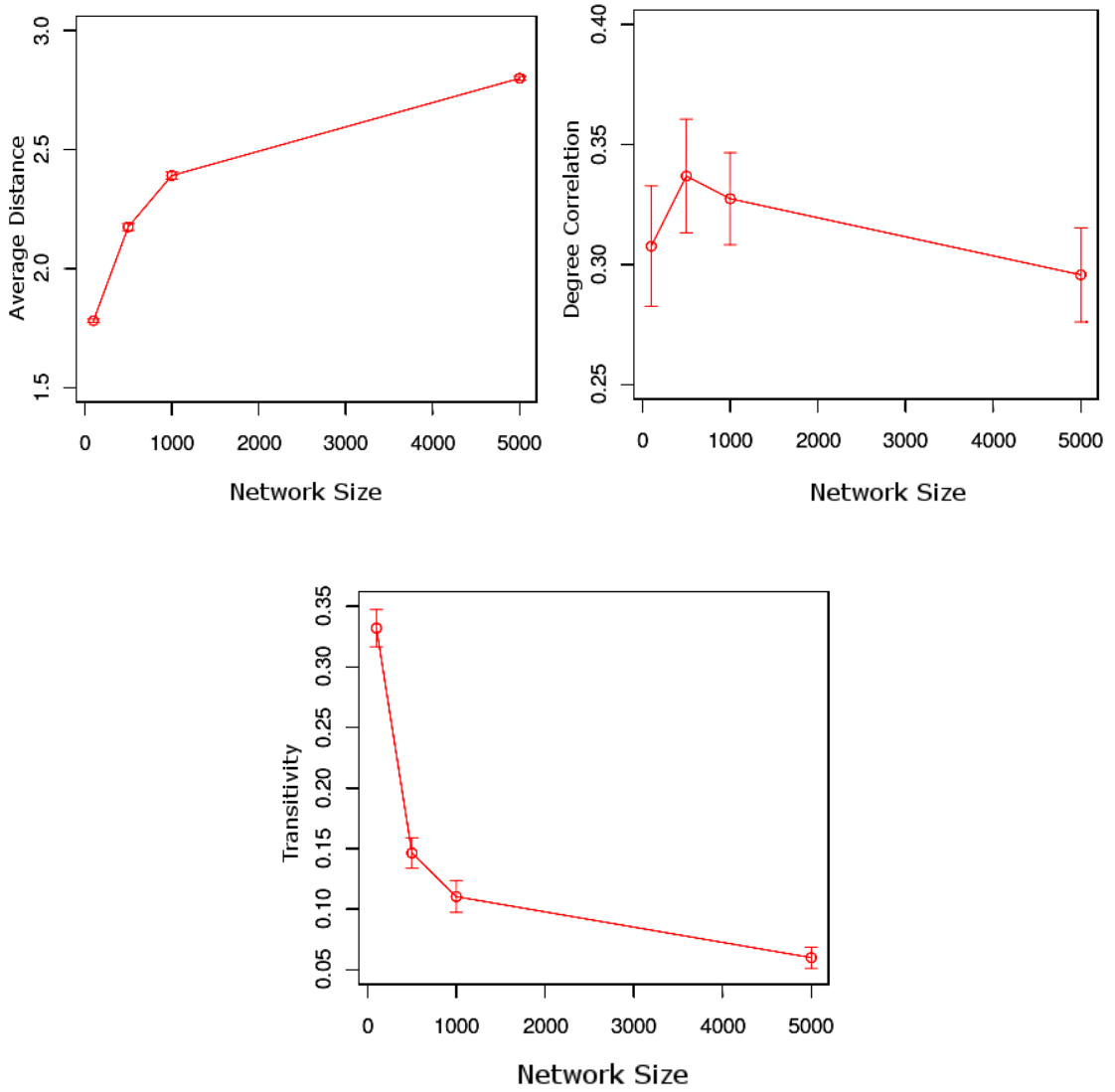


Figure 5.6 The change of average distance (top left) , degree correlation (top right) and transitivity (bottom center) with network sizes in LFR-BA model. Here $\beta = 2$, $\langle k \rangle = 30$ and $\mu = 0.5$

Degree correlation is a stable property for LFR-BA as shown in Figure 5.6. Network size seems to have no effect on this property. Its value is about 0.3 for any parameter. But, there is little progress in LFR-EV as one can see this on Figure 5.7. Degree correlation increases non linearly with network size. Moreover, we observe this more clearly when μ is lower as 0.05 to 0.3. The numerical value is about 0.5 for network size 5000 when μ is low and its value is minimum 0.3.

Transitivity also decreases non linearly for both models. Differently than LFR-CM, it decreases for any value of any parameter with network size increase. The trend does not depend on μ . We observe small decrease for higher μ and lower average degree because transitivity is close to 0.0 for those parameters. It ranges from 0.1 to 0.0 with node increase.

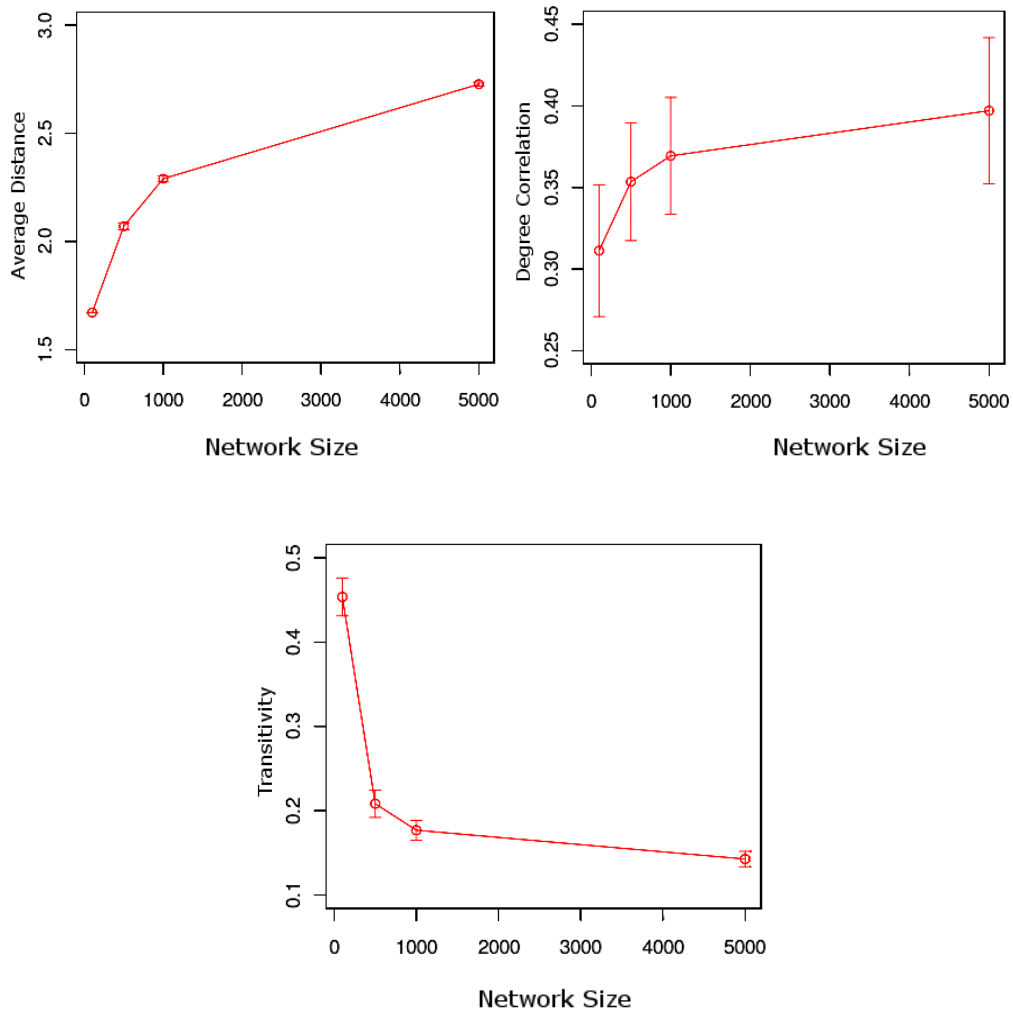


Figure 5.7 The change of average distance (top left) , degree correlation (top right) and transitivity (bottom center) with network sizes in LFR-EV model. Here $\beta = 2$, $\langle k \rangle = 30$ and $\mu = 0.5$

5.2.4 Mixing Coefficient

As LFR-CM, the most effective parameter on properties is μ . Average distance decrease non linearly when μ increase and we obtain almost the same numerical values for both models. It decreases until μ reaches 0.4 and then stays stable. We observe this decrease more clearly for higher network sizes or lower average degrees. When $n = 100$ with $\langle k \rangle = 15$, the average distance is almost 2 and it seems almost stable relatively to changes in μ . But when we increase network size, the trend becomes more noticeable. For example at $n = 5000$, average distance decreases from 5 to 3. Before explaining degree correlation change, we should state that; because the standard deviation of degree correlation on both LFR-BA and LFR-EV network makes the plots unclear, we do not use this information on the representation. But we would like to indicate that for both modified model, this standard deviation is higher than the original LFR-CM network. When we observe degree correlation, we see that LFR-BA is not very sensitive to μ changes. It seems stable for any network size and average degrees with μ increase. Moreover, degree correlation value is always about 0.3. By an attentive observation on the plots, one may notice a very few increase on average distance with μ for $n = 5000$. This change may be more distinguishable for larger networks,. One may also state that there is a more considerable variance of the iterations for lower network sizes and average degrees. Degree correlation seems to stay stable in LFR-EV too. But there is a decrease for high network sizes which is more noticeable than LFR-BA. The numerical value is about 0.5 and the variances of the iterations are more discernible with this model.

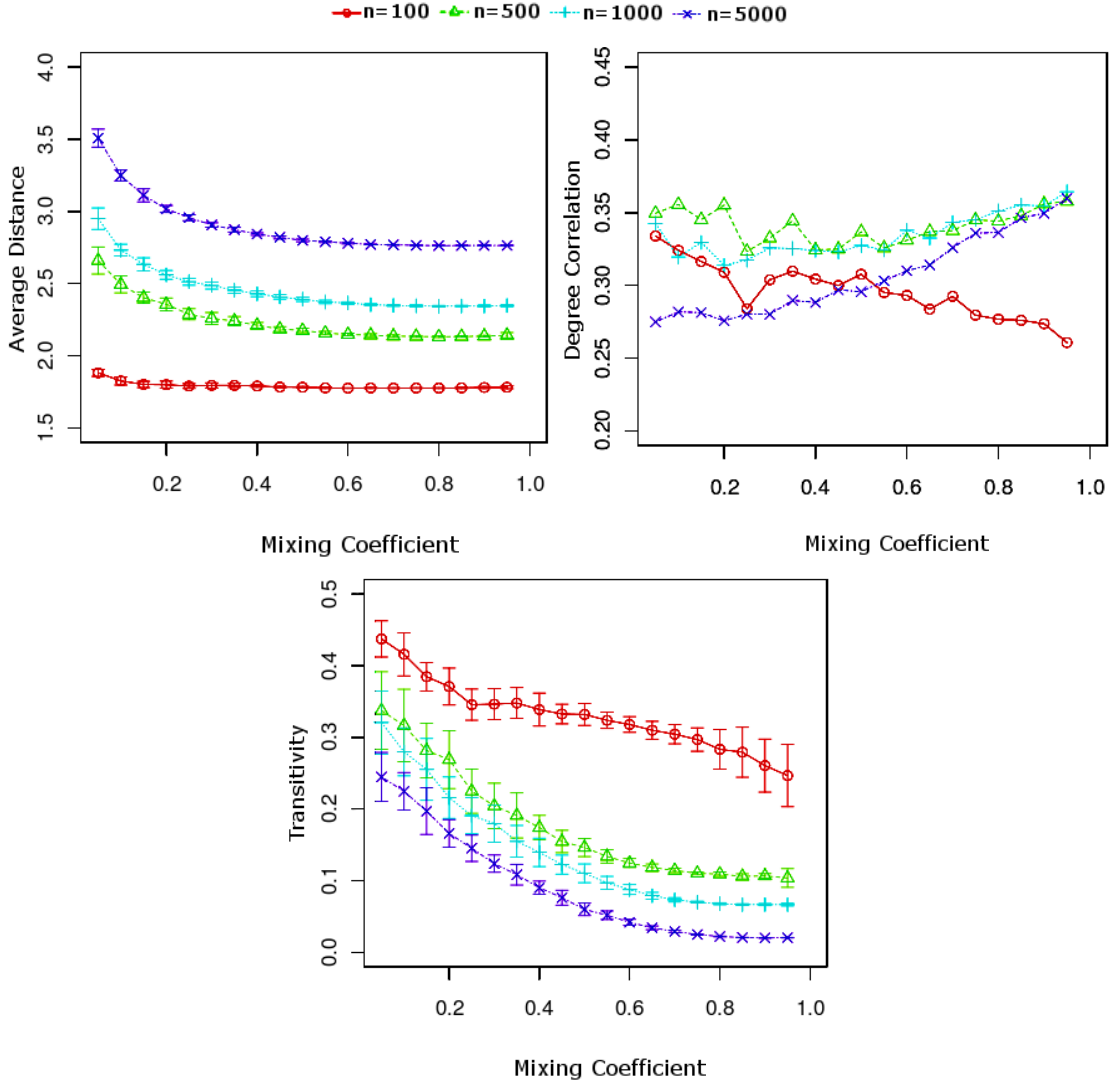


Figure 5.8 The change of average distance (top left), degree correlation (top right) and transitivity (bottom center) with mixing coefficient on all network sizes in LFR-BA model. Here $\beta = 2$ and $\langle k \rangle = 30$.

Transitivity decreases non linearly in LFR-BA and its numerical value depends on network size. For $n = 100$ and $\langle k \rangle = 30$, the decrease is not very distinctive from 0.45 to 0.3. But for higher network sizes, it becomes more noticeable. When $n = 500, 1000$ and 5000 with same average degree, it decreases from 0.35 to 0.1; from 0.3 to 0.05; from 0.2 to 0.0 respectively. We observe similar behavior for LFR-EV. For example, for $n = 100, \langle k \rangle = 30$, transitivity decreases from 0.6 to 0.2 and the plot has reverse sigmoid shape. For higher network sizes, it decreases non linearly. For

example; for $n = 500, 1000$ and 5000 , transitivity decreases from 0.5 to 0.1; from 0.5 to 0.1 and from 0.4 to 0.0 respectively.

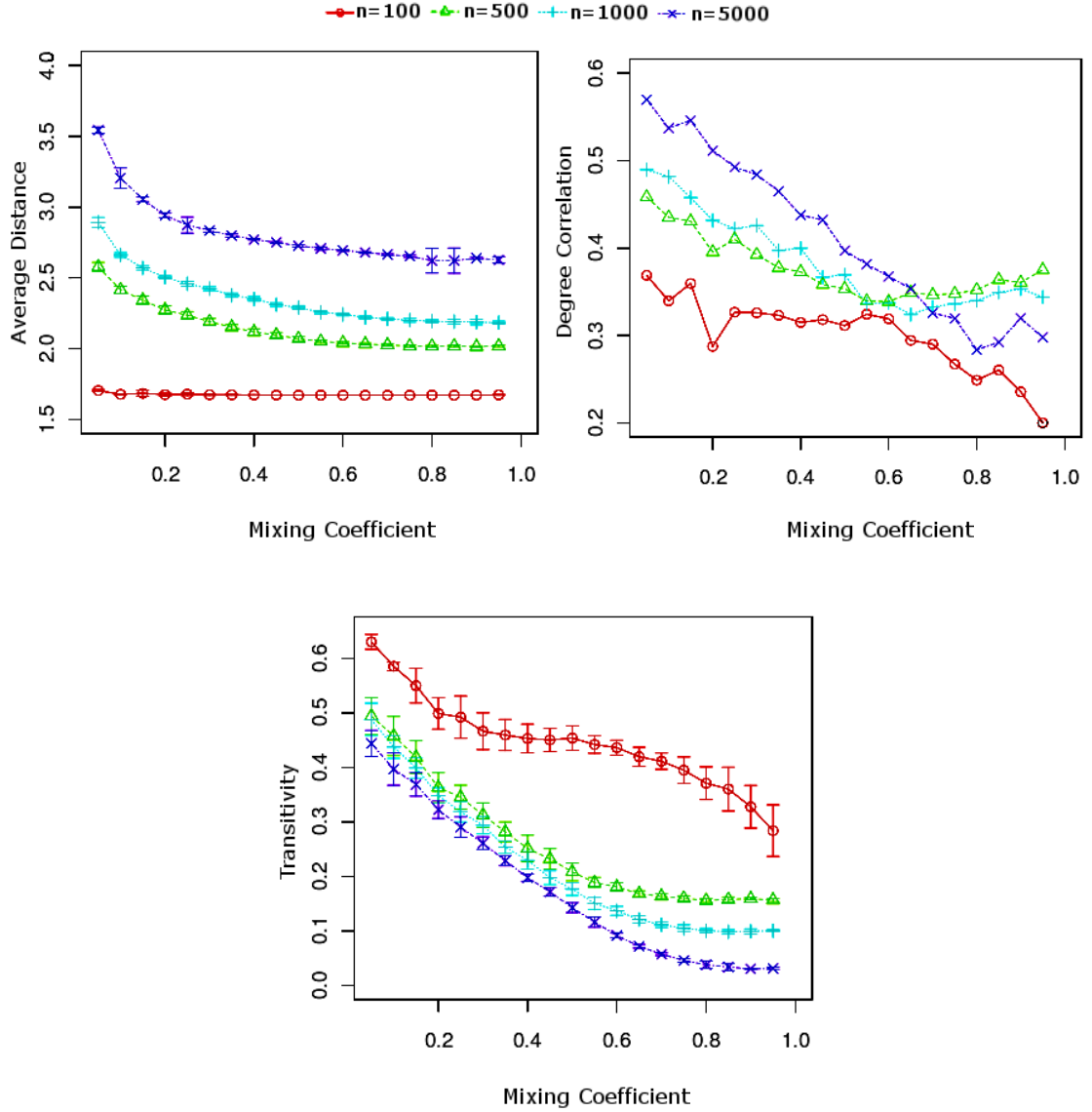


Figure 5.9 The change of average distance (top left), degree correlation (top right) and transitivity (bottom center) with mixing coefficient on all network sizes in LFR-EV model. Here $\beta = 2$ and $\langle k \rangle = 30$.

5.2.5 General Comparison

Figure 5.10 shows compared results for average distance, degree correlation and transitivity for all three models with network size 5000 and $\langle k \rangle = 30$. The largest communities in the generated networks have around 700 nodes, so communities are supposed to be structurally well-defined for $\mu < 0.86$. This μ limit is represented on the plots under the form of a vertical line. ANOVA and Tukey's test shows that average distance in LFR-CM, LFR-BA and LFR-EV lead to significantly different performances ($p < 0.05$). But although their values are statistically different this difference is small relatively to this property definition domain (they are all small-world) and their sensibility to mixing coefficient is rather similar for all three models. It ranges approximately from 2.5 to 4, and is relatively stable, especially for $\mu > 0.3$. On the one hand, the stability of this property is a good point, since it means networks with much separated communities (small μ) and networks with very mixed communities (high μ) have comparable average distances. Consequently, the effect of this property can be considered as negligible when comparing algorithms performances on networks generated with various μ values. But on the other hand, since all three models lead to very close average distances, this property cannot be used to compare them in terms of realism of the generated networks.

As one can observed on Figure 5.10, the models exhibit rather different transivities, which is confirmed by the ANOVA ($p < 0.01$) and Tukey's test ($p < 0.01$ for all pairs of models). However, this difference is not significant anymore for high μ values: for $\mu > 0.65$, LFR-CM and LFR-BA have similar transitivity ($p > 0.5$ for Tukey's test) and for $\mu > 0.85$, all three models have similar transitivity ($p > 0.1$). LFR-CM has the highest transitivity, with values around 0.6 (the theoretical minimum and maximum being 0 and 1, respectively) for $\mu \approx 0$, but it also has almost zero transitivity for $\mu \approx 1$, exhibiting a serious sensitiveness to μ . Other methods also show a decreasing transitivity when μ increases, but the range is much smaller, mainly because their values for $\mu \approx 0$ are significantly smaller: around 0.25 and 0.45 for LFR-BA and LFR-EV, respectively. Like LFR-CM, they reach almost zero value when $\mu \approx 1$. So contrarily

to what we expected, networks generated with LFR-EV do not have a higher transitivity than LFR-CM, at least for small μ . However, thanks to its lesser sensitivity to μ , LFR-EV has a better transitivity for $\mu > 0.3$. Note that in the literature, real-world networks with greater than 0.3 transitivity are considered highly transitive [64], so we can state all three models exhibit realistic transitivity for small μ . The issue is more about their sensitivity to μ , leading to non realistic values for high μ . This non-linear decrease in transitivity observed for all three models could be linked to the rewiring process performed by the LFR method. In this case, the final transitivity would never be stable, whichever model is used to generate the initial networks. But testing this hypothesis would require an exhaustive analysis of the side-effects of rewiring on networks, which is out of the scope of this work. Another explanation would be that network transitivity is directly related to the nature of community structure itself, independently of the way the network is created. Testing this hypothesis would require being able to quantify the separation level of communities in real-world networks (using Newman's modularity [3], for instance), in order to compare it to the measured transitivity.

Considering the degree correlation, the graphical difference between LFR-CM and the other two models is clear on Figure 5.10. ANOVA ($p < 0.01$) and Tukey's test ($p < 0.01$ for all pairs of models) show all models have significantly different degree correlations over all μ values. However, this does not hold for values of μ greater than 0.7, which lead to degree correlations not significantly different for BA EV, as reported by Tukey's test ($p \approx 1$). LFR-CM degree correlation has acceptable values for small μ (0.25), but it decreases rapidly and oscillates around zero for $\mu > 0.4$. EV shows the highest degree correlation, with values greater than 0.5 for $\mu \approx 0$. It also decreases when μ increases, resulting in values close to 0.25 for $\mu \approx 1$. Finally, unlike other models, LFR-BA degree correlation slightly increases with μ , ranging approximately from 0.25 ($\mu \approx 0$) to 0.35 ($\mu \approx 1$). Although its values are lower than for LFR-EV, it is also more stable both in terms of sensitivity to μ and low standard-deviation (especially for $\mu > 0.7$).

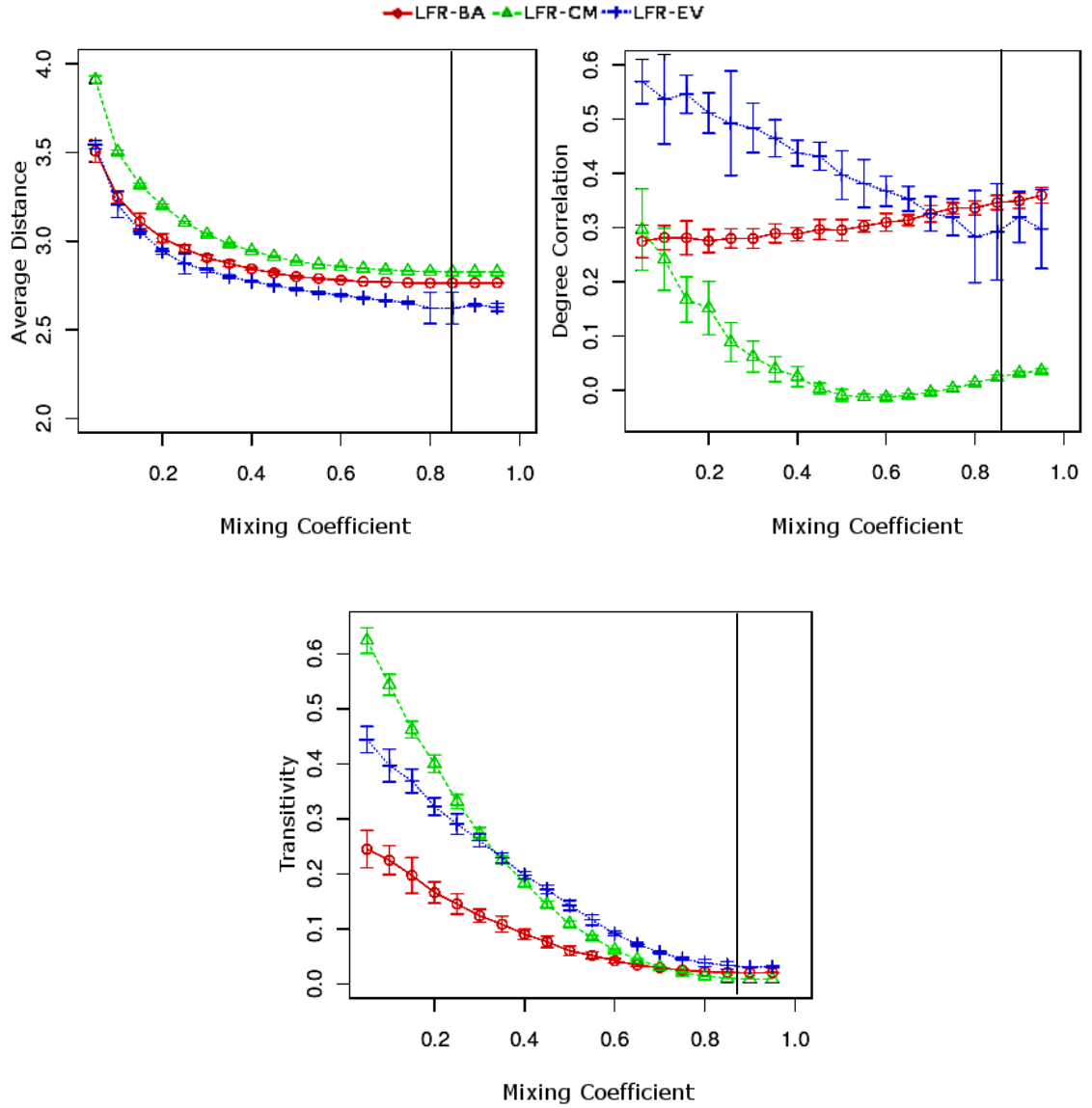


Figure 5.10 Comparison of average distance (top left), degree correlation (top right) and transitivity (bottom center) on three network models. Here $\beta = 2$, $\langle k \rangle = 30$ and $n = 5000$.

5.2.6 Conclusion

In conclusion to this section, we can state LFR-EV and LFR-BA are slightly but significantly above LFR-CM in terms of realism, as confirmed by ANOVA and Tukey's test ($p < 0.01$). Although statistically significant, the observed differences are not very meaningful for the average distance, since all three models produce small-

world networks. The transitivities are also significantly different, but nevertheless realistic for all three models, at least for small μ values. Concerning the degree correlation, both LFR-BA and LFR-EV exhibit realistic values for any μ , whereas LFR-CM is realistic only for $\mu \approx 0$. The main difference between the reviewed generative models is related to their sensitivity to μ . LFR-CM is clearly the most sensitive, showing the largest range of values for both transitivity and degree correlation, whereas LFR-BA is the most stable. However, LFR-EV generally has highest values than LFR-BA, so it is difficult to decide which one is the most adapted. When testing community detection algorithms, one generally wants to compare the algorithm performances on networks with more or less mixed communities. For this purpose, one considers different values of the mixing coefficient. For this matter, the observed sensitivity can be an issue since it shows the mixing coefficient has an effect not only on the community structure, but also on the network general level of realism. In other terms, a change in performance for different values of μ can be explained by the change in the community structure, but also by those in the network realism, or even by both of them. The next section will be dedicated to study how the differences in stability and realism translate in terms of community detection performances.

6 COMMUNITY DETECTION ALGORITHMS COMPARISON

To study the effects of the networks realism on the community detection process, we have applied all the community detection algorithms presented in section 3 on original and modified LFR networks. We could apply Girvan and Newman's EB algorithm on small sized networks but its time complexity makes it impossible to test this algorithm for higher network sizes. That is why; we do not present its results in this study. Some of the algorithms need some user defined parameters. The values of those parameters in our experiments are listed in Table 6.1. In this section, we explain the trend of the algorithms community detection performances depending on the parameters of network generation model. First, we analyze the results of each algorithm relatively to their category, on networks generated with the LFR-CM model. Second, we compare the three generative models in terms of algorithms partitioning performance.

Table 6.1 Values of the algorithms parameters

ALGORITHM NAME	REQUIRED PARAMETERS	VALUE
Radetal (RA)	Transitivity criteria	3 (triangle transitivity)
Spin glass (SG)	Spin number	Which maximizes modularity
Commfind (CF)	Number of eigenvector	Which maximizes modularity
Markov Clustering (MCL)	Inflation	2.0 (default)
Walk trap (WT) (hierarchical)	Random walk length	4 (default)
Infomod (IND)	Community number	Which maximizes modularity

6.1 LFR-CM Results

We have compared all eleven algorithms estimations with the reference community structures by using the NMI on our LFR-CM benchmark. Because we have many parameters and many algorithms, it is not easy to compare them all at once. We consider here the most prominent parameters and explain their effects on the algorithms performances. Of course, these performances are very sensitive to the mixing coefficient μ , which affects directly the community structure. In our benchmark, communities are structurally well-defined depending on network size and average degree as explained in equation (4.3). That is why; one expects that a good algorithm has to find appropriate community structure until this limit of μ . The values of these thresholds are declared on Table 6.2.

Table 6.2 Community sizes and mixing thresholds in function of the networks size and average degrees

NETWORK SIZE	AVERAGE DEGREE	LARGEST COMMUNITY SIZE	MIXING THRESHOLD
100	5	45	0.55
	15	55	0.45
	30	90	0.1
500	5	100	0.8
	15	130	0.74
	30	190	0.62
1000	5	130	0.87
	15	200	0.8
	30	250	0.75
5000	5	300	0.94
	15	400	0.92
	30	700	0.86

The performances were averaged over 25 iterations like for the properties in section 5. Note the graphical representation of the standard deviation is sometimes making the plot difficult to read, so we omitted it unless explicitly specified. But we consider this information on the evaluation of the algorithms. All algorithms exhibit high standard deviation during the fall of their performance. Additionally, standard deviation is generally high for LEV, CF, RA and LP, which means these algorithms are not very robust, since they can have very different performances for networks generated with the same parameters. The LP is particularly inconstant: when it is applied several times on the same network, it can output very different partitions. Besides, the density of the networks affects very much the consistency and performance of the algorithms. In general, they work well for $n = 5000$, $\langle k \rangle = 30$ with small standard deviations. We explain the change on the partitioning performance of the algorithms with μ by considering their category in following part.

6.1.1 Divisive Algorithm Performance

We can say that RA, which is a divisive algorithm, performs well for $n = 100$ when $\mu < 0.2$ such that its NMI value becomes 1 until that mixing coefficient limit. After that limit it decreases unsteadily until $\mu < 0.4$ and then decreases linearly. For $\mu > 0.7$, RA's NMI value becomes less than 0.1. But for higher network sizes, the plot shape is more similar to a reverse sigmoid. Its NMI is between 0.8 and 1 for $\mu < 0.6$ and then, decreases suddenly. For $\mu > 0.8$, RA keeps finding appropriate communities with NMI almost 0.3 and this value never reach 0. The partitioning performances of RA on $n = 1000, 5000$ networks are shown on Figure 6.1. The vertical lines on the plots represent μ threshold of community mixing on considered network size and average degree. For $n = 5000$, RA becomes more robust. Moreover, RA cannot find exact community structures even for low μ values. Its NMI is 1 only when $\mu = 0.05$

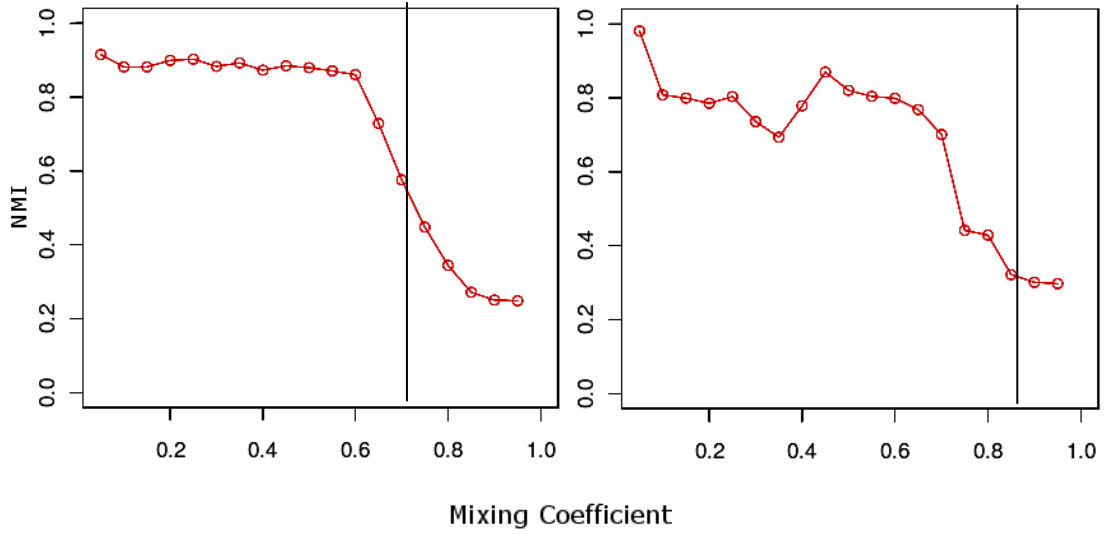


Figure 6.1 RA performance on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$.

6.1.2 Modularity Based Optimization Algorithms Performance

The modularity optimization algorithms FG differ significantly than LV and SG (all $p < 0.01$ for both ANOVA and Tukey's test). Surprisingly, FG's performance decrease trend is different from the other two algorithms. FG performance decreases almost linearly while LV and SG performances plots are shaped like reverse sigmoid. As all other algorithms, we cannot state their true behavior on lower network size. FG can find appropriate communities until $\mu = 0.8$ but after that limit, its NMI becomes 0. But FG can find exact communities only when they are very clearly defined as $\mu < 0.2$. LV and SG performance are very similar, not only by their shape but also according to Tukey's test ($p \approx 1$). They perform very well and find exact community structure for $\mu < 0.75$ and $\mu < 0.65$, respectively. Then, their performances decrease suddenly and $\mu > 0.85$, they cannot find appropriate communities. Unlike LV, SG's performance is almost the same for $n = 500, 1000$ and 5000 . But LV's inflection point is higher than other sizes for $n = 5000$ and before this point it performs worse. The partitioning performance of modularity optimization algorithms on $n = 1000, 5000$ networks are shown on Figure 6.2.

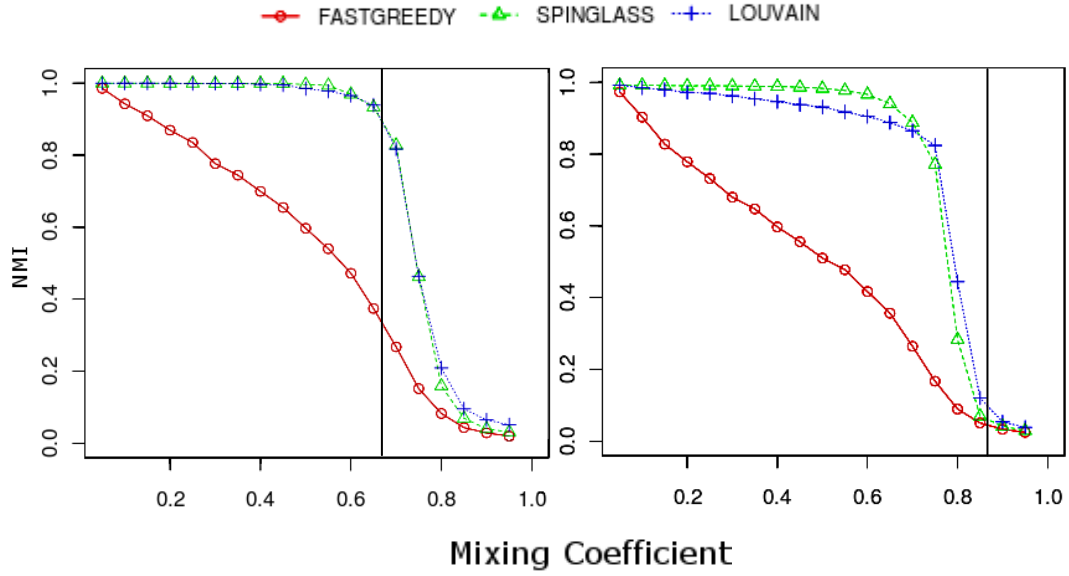


Figure 6.2 Comparison of optimization algorithms performance on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$.

6.1.3 Spectral Algorithms Performance

All spectral algorithms performances differ significantly for $\mu < 0.9$ ($p < 0.01$ for both ANOVA and Tukey's test). After this limit, ANOVA does not highlight any significant difference, but of course the values are all very low due to the difficulty to detect the communities. The LEV algorithm is one of the most inconstant algorithms on our test. Despite this, we may say its performance decreases linearly. It cannot find exact community structure even for low μ values. When $\mu < 0.1$, its NMI is 0.8, 0.8 and 0.55 for network sizes 500, 1000 and 5000, respectively. After $\mu = 0.8$, LEV cannot find appropriate communities. CF is less robust. Its performance also decreases linearly for $n = 1000, 5000$. But interestingly, for smaller network sizes, CF performs very well until $\mu = 0.45$ and then, decreases suddenly. We suppose that this trend difference depends on the average degree or density of the networks. Like LEV, CF cannot find appropriate communities for $\mu > 0.8$. The partitioning performances of spectral algorithms on $n = 1000, 5000$ networks are shown on Figure 6.3.

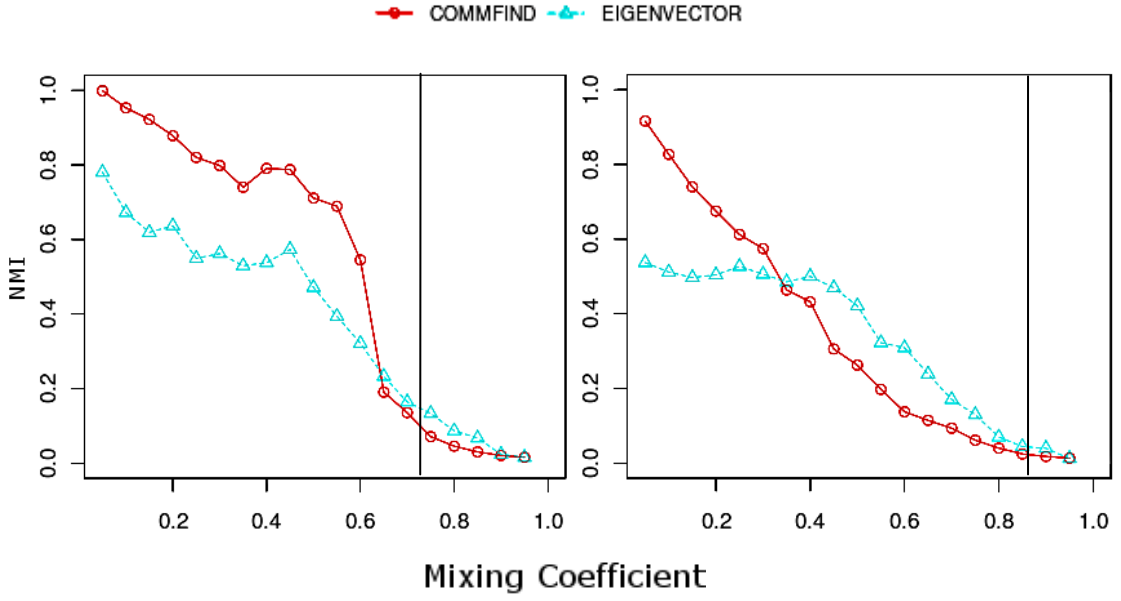


Figure 6.3 Comparison of spectral algorithms performance on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$.

6.1.4 Dynamic Algorithms Performance

The dynamic algorithms MCL, WT and INP are in general robust and perform well. ANOVA shows their performances are not significantly different below a certain μ threshold ($p > 0.01$ when $\mu < 0.6$ for $n = 1000$ and when $\mu < 0.7$ for $n = 5000$, according to Tukey's tests). For $n = 100$ and $\langle k \rangle = 30$, MCL cannot find appropriate communities when $\mu > 0.2$. Its performance decreases suddenly on that inflection point and NMI value becomes 0. But for the same size, depending on average degrees, its performance becomes better and for lower $\langle k \rangle$, even for high μ , its NMI is almost 0.6. For higher network sizes, MCL finds appropriate communities for $\mu < 0.4$ and then, decreases suddenly until $\mu = 0.7$. After that μ , it still performs better. For $n = 500, 1000, 5000$, the NMI values are 0.5, 0.6 and 0.7, respectively. In fact, it is an interesting point, because the communities are not well defined for higher μ . NMI seems to be sensitive to the number of estimated communities, which may unrealistically increase the measured performances for algorithms tending to create many communities. This assumption should be tested more further, but it is out of the

scope of this work. WT is another dynamic algorithm which performs consistent and well. Even for $n = 100$, $\langle k \rangle = 30$, WT can find exact community structures when $\mu < 0.3$. For higher network sizes, its NMI value is 1 until μ reaches 0.7. After that limit, its performance decreases and when μ reaches 0.85, $n = 500, 1000, 5000$, the NMI values become 0.05, 0.1, 0.3 respectively.

INP's performance decrease trend is different than for the other dynamic algorithms. For $\langle k \rangle = 30$, we do not observe reverse sigmoid shape on its plot. INP keeps its good performance until a μ threshold which depends on network size, and after this limit its performance decreases suddenly and reaches 0. This μ limit is 0.2, 0.6, 0.65 and 0.8 for network sizes 100, 500, 1000 and 5000, respectively. But for other average degrees, its plot are also reverse sigmoid shaped. For $\langle k \rangle = 5$, INP works well and finds exact communities when $\mu < 0.4$ for any network size. Then, its performance decreases but even for $\mu > 0.7$, its NMI is greater than 0. We suppose this different trend may depend on the density of the networks. The partitioning performances of dynamic algorithms on $n = 1000, 5000$ networks are shown on Figure 6.4.

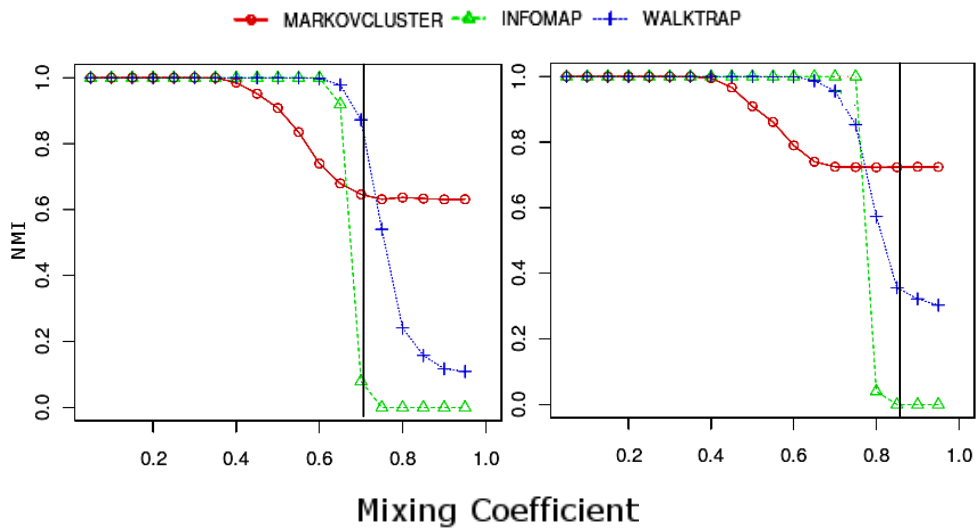


Figure 6.4 Comparison of dynamic algorithms performances on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$.

6.1.5 Other Algorithms Performance

LP and IND performance plots also follow a reverse sigmoid shape. For $n = 5000$, ANOVA shows a significant difference between those two algorithms performances. LP's very large variation is a prominent property of this algorithm. LP is also very sensitive to network size and average degree, as is INP. LP's behavior is pretty similar to INP's behavior. For sparser networks, its plot is as reverse sigmoid but for denser ones, LP's performance decreases more suddenly and after an inflection point of μ depending on network size, it cannot find appropriate communities. For $\langle k \rangle = 30$, LP's performance is very well. It finds exact community structure but its performance decreases suddenly until μ reaches 0.1, 0.4, 0.55 and 0.7 for network sizes 100, 500, 1000 and 5000 respectively and after those limits, LP's NMI becomes 0. The sharpness of decrease depends on the network size too. IND is not as successful as LP on low μ values. It is the most sensitive algorithm to average degree. For different average degrees, there is an obvious numerical change on NMI values for low μ . It is about 0.7, 0.85 and 1 for average degrees 5, 15 and 30 respectively. IND is not work consistently for $n = 100$. The inflection point of NMI on μ depends on the network size. Until μ reaches 0.5, 0.6 and 0.65, it performs better for network sizes 500, 1000 and 5000. After that inflection point, its performance decreases and becomes 0. The partitioning performance of IND and LP algorithms on $n = 1000, 5000$ networks are shown on Figure 6.5.

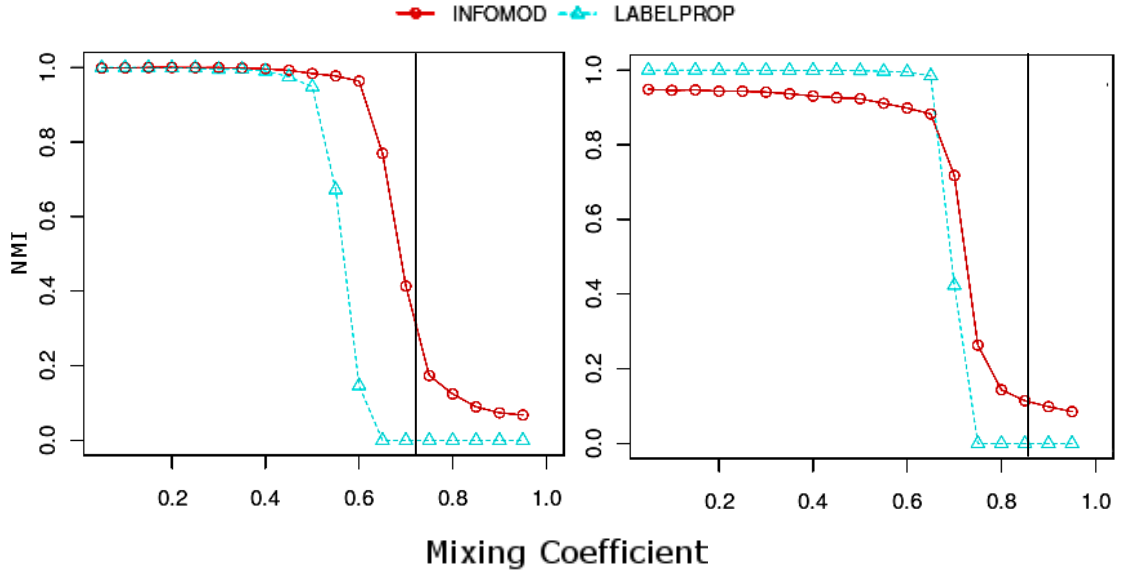


Figure 6.5 Comparison of other algorithms performances on 1000 (left) and 5000 (right) nodes LFR-CM networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$.

6.1.6 Joint Effects of Parameters

Mixing coefficient is the most effective parameter on algorithms performance. There is also a joint effect of average degree and μ . When observing this joint effect on the performance, it is interesting to note the reversal taking place around $\mu = 0.8$, as illustrated by Figure 6.6 for WT and SG. We present here the most significant algorithms plot but this is valid for all algorithms. Below this limit, the higher the degree and the better the performance. Above this limit, the lower the degree and the better the performance. This means high density helps discovering community structure when it is strong, whereas it hides it when it is weak. But as we stated in the previous section, the generated networks become less realistic when μ increases, so the observed change in performance could alternatively be caused not directly by the degree variations, but by consequent decreases in the transitivity or degree correlation.

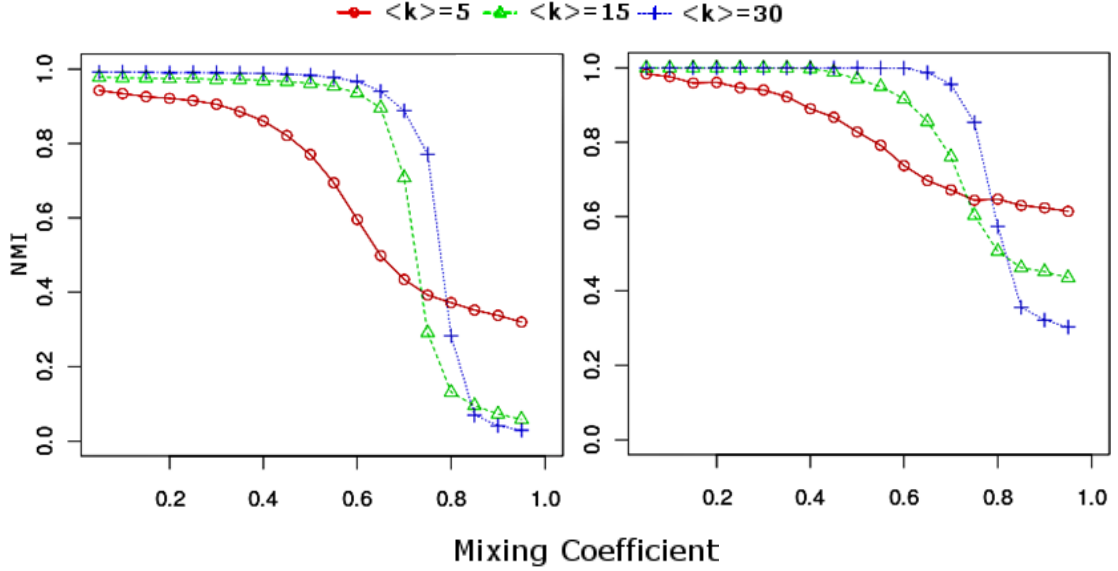


Figure 6.6 SG (left) and WT (right) performance with joint influence of the mixing coefficient and the average on LFR-CM networks. Here $\gamma = 3, \beta = 2$ and $n = 5000$.

6.1.7 Summary

We can summarize the partitioning performance of the algorithms according to the results described in this section. As the result of the properties, the γ and β parameters have almost no effect, but $\langle k \rangle, n$ and especially μ have notable effects. The performance of many algorithms decreases along the mixing coefficient, taking the form of a reverse sigmoid shape, with different slopes and inflection points. But some algorithms exhibit different behavior. FG's performance decrease linearly and LEV and CF do not show a general aspect in their fall. RA, CF and LEV perform worse for higher network size. They cannot find exact community structure and their performance drops on lower μ than the limit listed on Table 6.2. IND and LV respects these limits, but while they both can find exact community structures for lower network sizes, they cannot do the same thing for higher ones. Another remarkable point is the fact the performance of most algorithms for $n = 100$ is much different than what is observed for other sizes. The communities are well defined on $n = 100$ and $\langle k \rangle = 30$ for $\mu < 0.1$. However, some algorithms can find exact communities when $\mu < 0.3$. This may be because of the unstable structure of the networks: the fact they contain only 100 nodes makes it

difficult for the rewiring process efficiently create the community structure. Dynamic algorithms are one step ahead than other algorithms, but some modularity optimization methods also perform well. Spectral methods are the most inconstant ones, with high standard deviation and unstable trend. So a clear hierarchy appears between algorithms, in terms of general accuracy, we can order the algorithms: $LEV < CF < FG < RA < IND, LP < LV, SG < MCL, WT < INP$.

6.2 Compared Results

We observed more or less notable differences between the algorithms performances when applied to the LFR-CM generated networks. But previous studies showed the realism level of the generated networks has an effect on algorithms performance [7]. In the light of our analysis of the networks properties realized in section 5, we know the networks generated with the LFR-BA and LFR-EV variants are more realistic and more robust to changes in the mixing coefficient. Therefore, we will now apply the same eleven community detection algorithms on the corresponding collections of networks in order to study the effect these specific changes in the networks realism have on the algorithms performances.

6.2.1 Divisive Algorithm Performance Change

The divisive algorithm RA's performance does not have a regular decrease in its performance on networks generated by LFR-CM. As seen on Figure 6.7, it works more stable on LFR-BA and LFR-EV. We have underlined in section 5 that LFR-BA produces networks with the most stable uncontrolled properties, which may explain this change in RA's behavior. The μ threshold is almost 0.5 and 0.7 for LFR-BA and LFR-EV, respectively. Although the NMI value is greater for LFR-BA when $\mu < 0.5$, this algorithm can find appropriate communities relatively accurately even when they are not very distinct in LFR-EV. We observe for all network sizes and average degrees, RA's performances are above on the networks which are generated by LFR-EV than other models. This is confirmed by ANOVA, and Tukey's test shows LFR-CM and

LFR-BA do not lead to significantly different performances ($p > 0.05$) whereas LFR-EV does ($p < 0.01$ for both LFR-BA and LFR-CM). We assume this can be the result of the higher transitivity of LFR-EV compared to LFR-BA. As we observe on Figure 6.7, when $0.6 < \mu < 0.9$, RA performs better on LFR-EV than LFR-BA ($p < 0.01$ for Tukey's test). This interval corresponds to the points for which LFR-EV's transitivity is better. Moreover, for the same points RA works also well on the networks which are generated by LFR-CM, which is also more transitive than LFR-BA.

RA's performance is very sensitive to the density of networks. Its performance plot becomes a reverse sigmoid when $\langle k \rangle = 30$ and $n = 500$ or 1000 , for all three models. Especially, when $n = 500$, for all three models it can find exact communities as long as $\mu < 0.45$. However, it performs worse for higher network size or lower average degree. It is not robust on networks generated by LFR-CM.

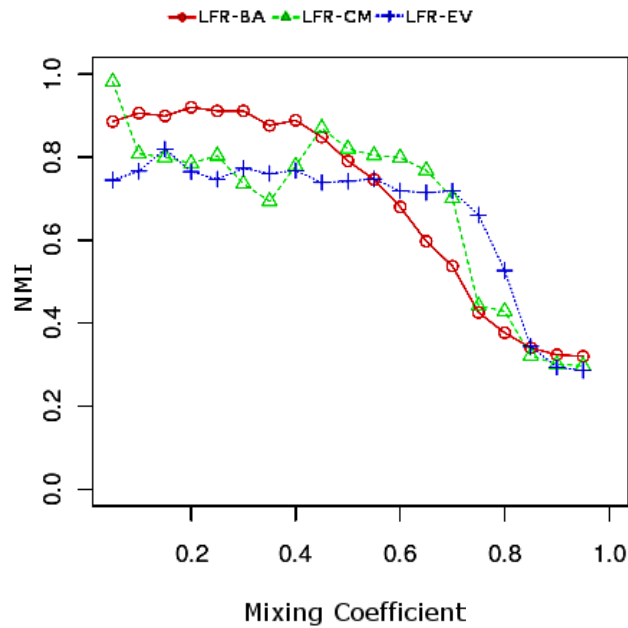


Figure 6.7 RA performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV networks. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$.

6.2.2 Modularity Based Optimization Algorithms Performance Change

Modularity optimization algorithms (except FG) work well on all three models. Their NMI results are almost 1 when μ is below the mixing threshold. As seen on Figure 6.8, the trend of each modularity optimization algorithm is the same for all three models. Unlike most algorithms, FG's performance decreases linearly when μ increases. FG does not seem to be sensitive at all to the generative model, since its performances for all three models are not significantly different (ANOVA $p \approx 1$). This suggests the information it uses to identify communities is not related at all to transitivity nor degree correlation. FG is essentially based on a modularity optimization approach, so on the one hand, this raises a question regarding the sensitivity of modularity to these properties. On the other hand, FG is not the best algorithm for modularity optimization, plus LV, which is also modularity-based, shows signs of sensitivity to the model. ANOVA shows the LV algorithm does not exhibit any model-sensitiveness until it reaches its drop-off limit, below which $p > 0.01$. Then, according to Tukey's test, performances are clearly and significantly better for LFR-CM and LFR-EV than for LFR-BA: For $\mu > 0.55$ and $\mu > 0.65$, $n = 5000$, $p < 0.01$ for Tukey's test on LFR-CM-LFR-BA, LFR-EV- LFR-BA respectively. This order fits with the models transitivity, so we could assume LV performs better when this property is high enough. However, LFR- EV transitivity is higher than LFR-CM's for $\mu > 0.3$ and this does not appear at all in the performance plot. On the contrary, the performance for LFR-CM stays above the other models until 0.8, whereas its transitivity is roughly the same. SG, which is the best modularity optimization algorithm according to our tests, performs more or less similar on all models ($p > 0.01$ for $\mu < 0.65$ and $\mu > 0.85$). However, we observe on Figure 6.8 that when considering LFR-BA, the μ threshold is lower than for most other algorithms. Especially, when $0.6 < \mu < 0.85$, SG performance is better for LFR-EV generated networks than LFR-CM generated networks and it performs the worst for LFR-BA generated networks. ($p < 0.01$ for Tukey's test).

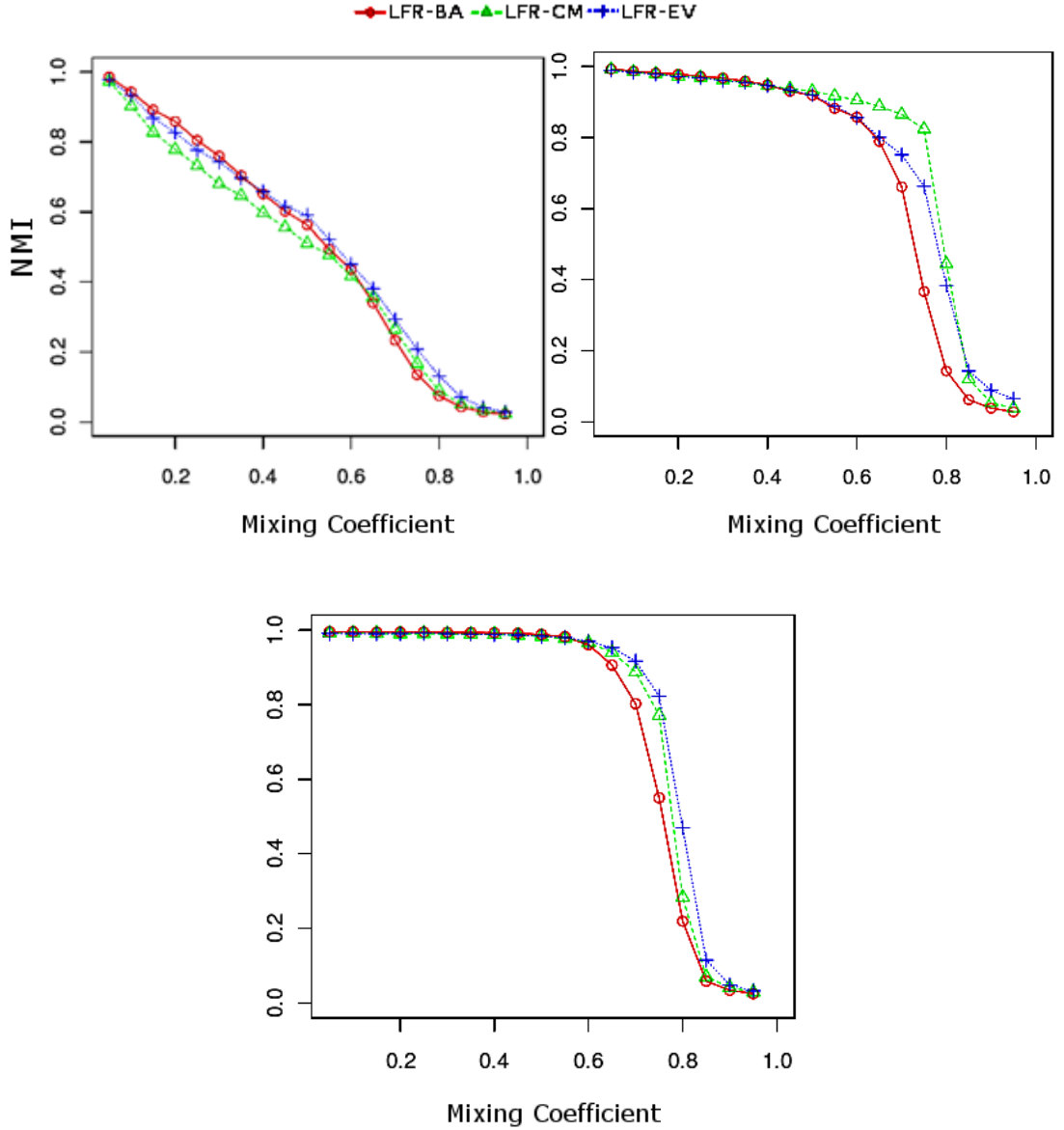


Figure 6.8 FG (top left), LV (top right) and SG (bottom center) performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV networks. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$

There is an interesting common point on all modularity optimization algorithms. There is a joint effect of generation model and μ when $\langle k \rangle = 5$ for any network size. All three algorithms FG, LV and SG perform better on the networks which are generated by LFR-EV before a μ limit depending on network size. After this limit, the rank of the models change and all those algorithms start to work worse on the networks which are generated by LFR-EV. The rank of other two models changes by size and algorithm.

This may be due to the considerable average distance observed when $\langle k \rangle = 5$. LFR-EV is the model which produces the networks having the lowest average distance on $\langle k \rangle = 5$.

6.2.3 Spectral Algorithms Performance Change

The spectral algorithms are the most inconstant ones for all three models. They perform significantly better on the networks which are generated by LFR-CM ($p < 0.01$). We observe the trend of the CF algorithm on the networks generated by LFR-BA very clearly, as seen on Figure 6.9: its performance does not decrease linearly. CF uses a distance on the eigenvector space. Interestingly, the shape and order of the CF performance according to models is very similar to the change of the transitivity corresponding to those three models. We suppose that transitivity has an effect on CF performance. But, it is not obvious because, the trend of CF varies according to network size and average degree. For example, when $n = 500$ and $\langle k \rangle = 30$, its performance decreases and the plots has the shape of reverse sigmoid. We can affirm it performs best on networks generated by LFR-CM and worst on LFR-BA ($p < 0.01$ when testing LFR-CM vs. LFR-EV and LFR-EV vs. LFR-BA with Tukey's), considering all other values of network size and average degree. Despite of its very variable trend, LEV works also best in LFR-CM too. As shown on Figure 6.9, for any model, LEV cannot find appropriate community structure very well. Its NMI values is already around 0.5 while μ is even low. This means LEV cannot find appropriate communities even for low μ values on each three collections.

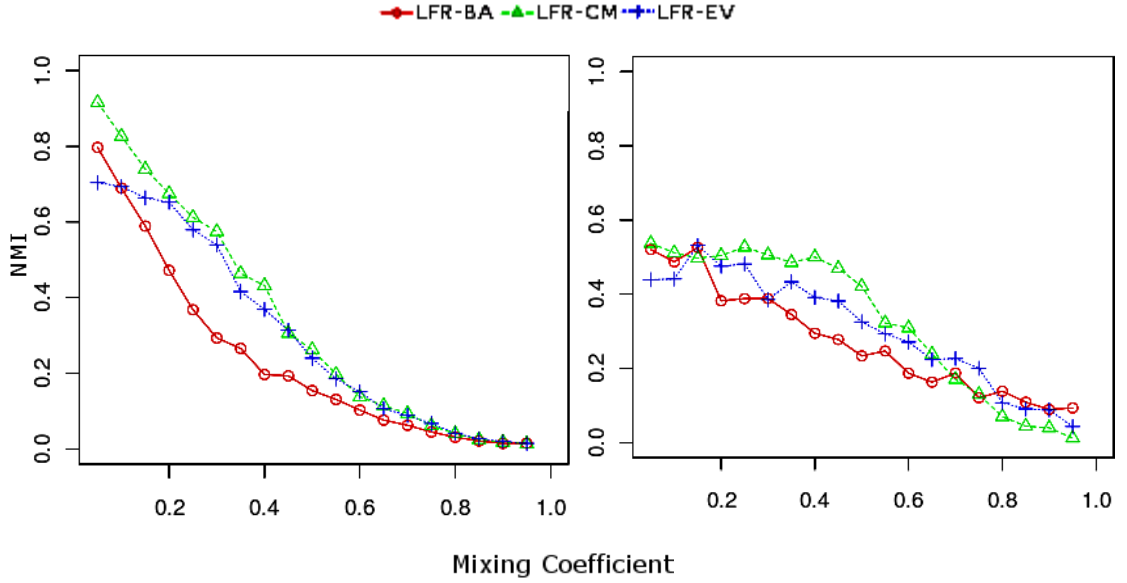


Figure 6.9 CF (left) and LEV (right) performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV network. Here $\gamma = 3$, $\beta = 2$ and $\langle k \rangle = 30$

6.2.4 Dynamic Algorithms Performance Change

Dynamic algorithms performances are relatively good on all models. Their NMI results are around 1 until μ reaches the mixing threshold. But, we also observe the order of the models on their performance very clearly. Although the μ threshold of community mixing is the same for all models, we see obviously on Figure 6.10 that the dynamic algorithms performances drop off in LFR-BA before other models. For all three models, it seems that MCL finds almost appropriate community structure even when the communities are mixed (after mixing coefficient threshold) for $\langle k \rangle = 5; 15$. But its NMI value becomes 0 and 0.4 for LFR-BA and LFR-EV respectively when $\langle k \rangle = 30$ and network size is smaller than 5000. This high NMI values results on less dense networks may due to the fact of some weakness of NMI measure. WT works with almost same performance on LFR-CM and LFR-EV ($p > 0.4$). However, the difference of LFR-BA appears clearly on Figure 6.10. This result plot is very similar for different parameter values. Although there is a change on the order of LFR-CM and LFR-EV, for all parameters values LFR-BA causes worse WT performance ($p < 0.01$). Especially when $0.6 < \mu < 0.85$, ANOVA and Tukey's test show WT performs

worse on LFR-BA generated networks ($p < 0.01$). It is also valid for other network sizes, 100,500,1000 and average degrees, 5 and 15. After the community mixing threshold, WT works with similarly on LFR-BA and LFR-CM ($p > 0.05$) while it works worse before this limit. It may be the result of the degree correlation of LFR-BA. This property's value is above on LFR-BA generated networks than other models generated networks after μ limit while it is below before that threshold. WT is a dynamic algorithm which uses a distance consisting of the random walk length. Besides, random walk process is about the probability of choosing a neighbor uniformly randomly. Thus degree correlation may have an effect on this criterion. INP's performance shows a very sudden decrease on all three models. But, we must underline that the variance during the inflection point is very high on LFR-BA model. In fact, its performance decreases more smoothly on lower average degree for all models. As seen on Figure 6.10, the inflection point order is LFR-CM > LFR-EV > LFR-BA which is the reverse of the order of comparing uncontrolled properties on those three models. INP is also a dynamic model based on random walks. That is why; the explanation we proposed to justify WT results may be valid for INP too. Moreover, the way models are ordered relatively to the algorithm performance is the same than when considering their transitivity. This is another point in favor of our hypothesis regarding the effect of transitivity on random walks.

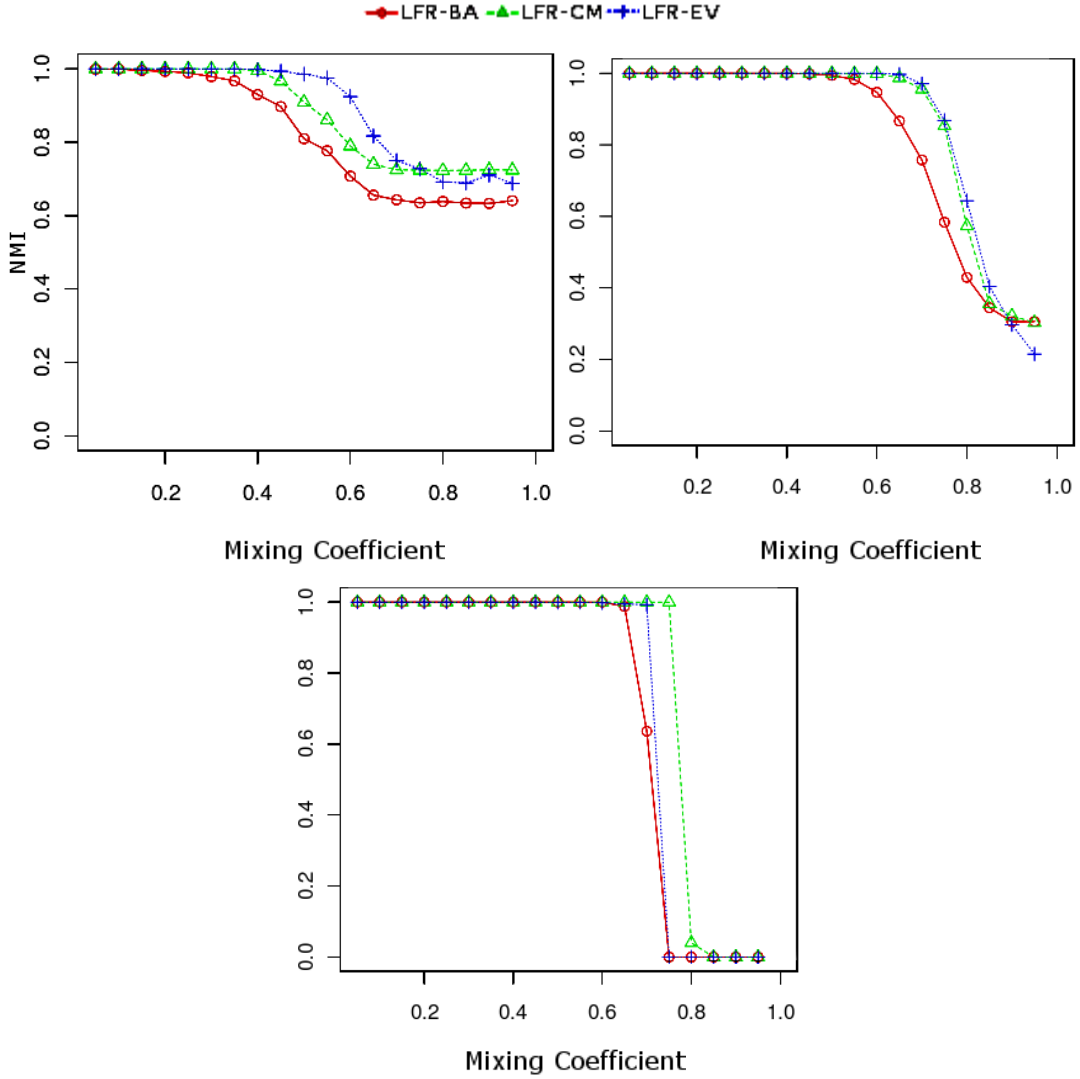


Figure 6.10 MCL (top left), WT (top right) and INP (bottom center) performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV networks. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$

6.2.5 Other Algorithms Performance Change

We observe similar shapes for LP and IND as for the dynamic algorithms, as seen on Figure 6.11. The difference between models is significant on these algorithms too, especially above the mixing threshold ($p < 0.01$). LP, which is not modularity-based, is much more sensitive to the generative model. LFR-EV and LFR-BA have close low drop-off limits, around 0.4 and 0.5, whereas it is approximately 0.7 for LFR-CM.

However, note these values may not precisely represent the actual performance, due to the high variance observed in LP results. LP performance is far better for LFR-CM than for the other models, which could suggest it finds more realistic networks harder to process. More precisely, the way models are ordered in terms of performance is the exact opposite of their order in terms of degree correlation. This could mean LP does not handle well networks with positive degree correlation, we suppose it is because such a property modifies the way labels spread in the network. But this observation is difficult to confirm here. Indeed, for $\mu > 0.7$ BA has a higher correlation degree than LFR-EV, but the effect on performance cannot be discussed because NMI has already reached zero for these values of μ . The greatest difference between models is observed on IND performance. Especially when $\langle k \rangle = 30$ and $n = 500$ or 1000 , IND's NMI value is 1 until μ reaches its limit for LFR-CM. However, this value is much smaller for both LFR-BA and LFR-EV. It is 0.6 and 0.8 respectively for network sizes 500 and 1000. Although this difference decreases, it is still distinguishable for $n = 5000$. After μ threshold, this high performance in LFR-CM decreases when $\mu > 0.8$. The performance of IND is directly proportional to the average degree in LFR-CM. But one cannot observe that exact proportion on each μ with LFR-BA or LFR-EV. We suppose that this is also because of the more stable degree correlation observed for those models.

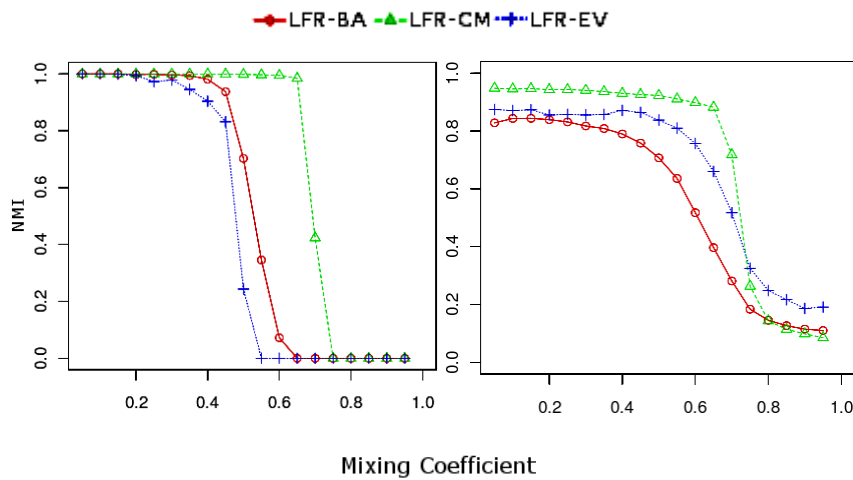


Figure 6.11 LP (left) and IND (right) performance on 5000 nodes LFR-BA, LFR-CM and LFR-EV networks. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$

6.2.6 Summary

For LFR-BA and LFR-EV as for LFR-CM, μ is the most effective parameter and $\langle k \rangle$ and n have also an effect on the partitioning performance of the algorithms. The trend of the NMI performances in function of μ is generally similar. However, some algorithms are more robust on the networks which are generated by modified models. The μ threshold after which the algorithms performances start to decrease also differs model by model. Although the μ limit which makes the communities mixed is roughly the same for all three benchmarks, the algorithms performances start to drop on lower μ for LFR-BA networks. As seen on Figure 6.12, μ limit is more or less the same for LFR-CM and LFR-EV but it is smaller for LFR-BA, for both 1000 and 5000 nodes. This can due to the lower transitivity and higher and more stable degree correlation of LFR-BA model. Moreover, while some algorithms as WT, INP, SG, FG trend is the same for all models, some others as RA, LEV or CF change model by model. Thus, the effect of the generative model on community detection performance depends strongly on the considered algorithm. All algorithms dispersion is larger in LFR-CM networks than in both LFR-BA and LFR-EV networks. Especially for low network size, LEV, CF, RA, LP and also IND have a very significant variance (having high dispersion on 25 iterations). Even for larger networks, we observe a considerable variance on these algorithms in LFR-BA. The ranking of the partitioning performance of the algorithms is changing with modified models too.

For all three LFR variants, a higher density tends to improve community finding when the communities are distinct, but makes it harder to find them when the community structure is weak. In these algorithms and in the modularity measure, a community is defined as a subset of nodes densely interconnected relatively to the rest of the network. This definition does not hold anymore when the mixing coefficient crosses a threshold which depends on the network size and largest community size. In other terms, above this limit, the network structure does not reflect the community structure, and the information conveyed by the network links is not pertinent anymore. Moreover, increases in the mixing coefficient also make the networks becoming less realistic, which could as well be a cause for the observed drop in performance.

The compared algorithms use different principles and mechanisms to identify communities, which can explain why their performances are influenced in various ways by the studied generative models. Even for the algorithms which are in the same category, we observe different variations depending on the model. But we have stated before that the separation of the algorithms is not always very strict. However, according to most of the algorithms, it generally appears networks generated with the LFR-BA model are the most difficult to process, whereas those coming from the LFR-CM are associated to the highest results. The LFR-EV model lies somewhere in between, sometimes closer to the former and sometimes closer to the latter. It remains difficult to explain exactly why in general we observed differences in performances, because the data seem to be incomplete for that matter. Drawing more solid conclusions would necessitate first applying other algorithms, if possible using a wide range of principles to detect communities. Additionally, one could consider studying other network properties, possibly responsible for the changes in performance. However, for now, we would say results measured on the BA-based LFR method are the more reliable, because of the stability of generated networks properties to changes in μ .

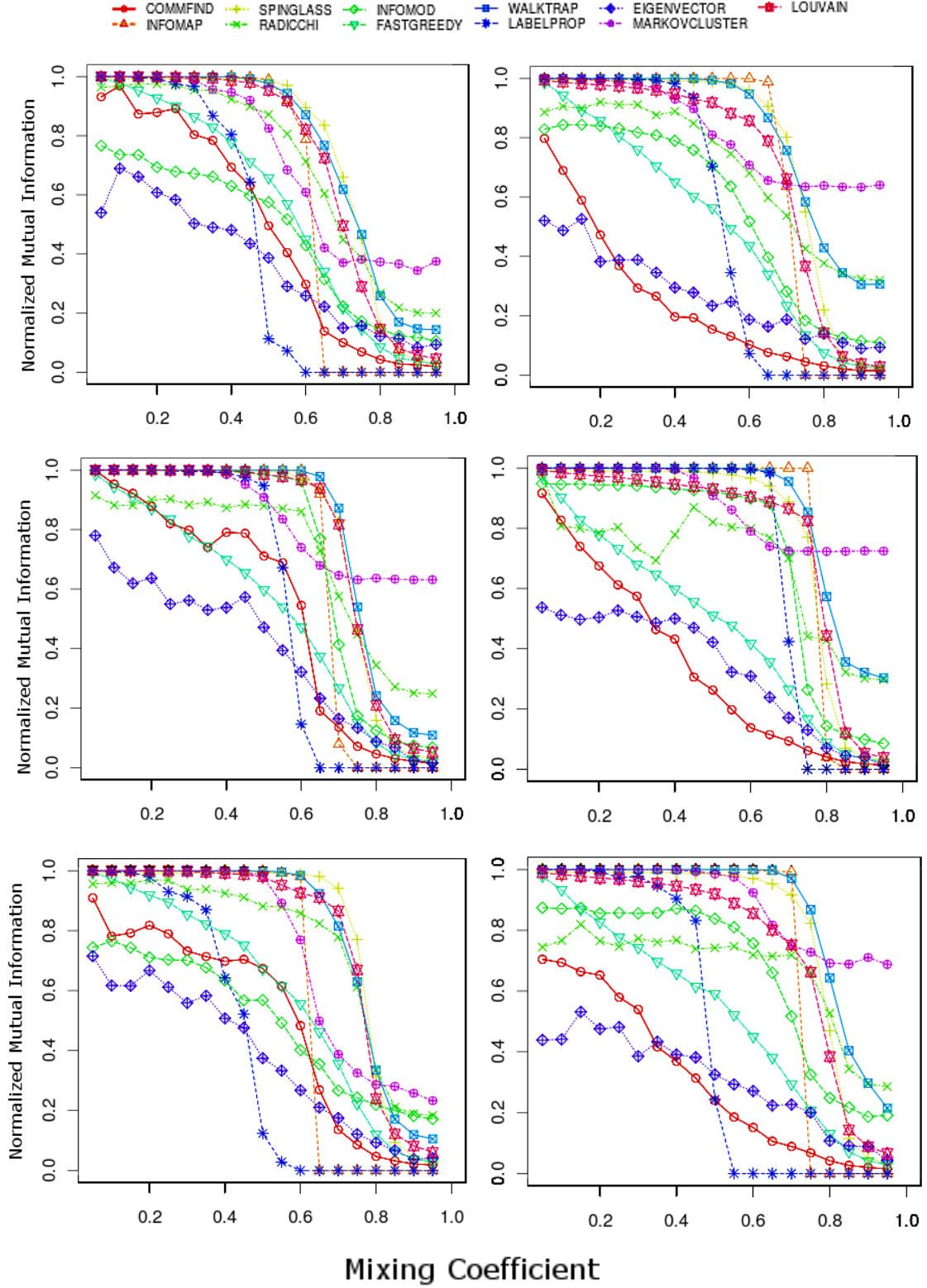


Figure 6.12 Comparison of the eleven algorithm's performance on 1000 (left column) and 5000 (right column) nodes LFR-BA (top), LFR-CM (center) and LFR-EV (bottom) networks. Here $\gamma = 3, \beta = 2$ and $\langle k \rangle = 30$.

7 CONCLUSION

In this work, we have studied the effect of the realism level of artificially generated networks on the performances of community detection algorithms. First, we used the model by Lancichinetti *et al.* [7] (LFR) to randomly produce networks with a community structure and power-law degree and community size distributions, using realistic values for the properties the model allows to control directly. This model basically uses the configuration model (CM) [67] to randomly generate a network with power-law distributed degree, and then rewires it partially to make a community structure appear, with a specified mixing coefficient μ . This type of model is used to benchmark community detection algorithms on huge collections of artificial networks, before testing them on real-world ones [7, 8, 50]. For this reason, their realistic aspect is of utmost importance and is known to affect the algorithms performances [9]. To our knowledge, LFR is currently the method generating the most realistic networks with community structure. We have examined the realistic level of the generated networks by analyzing their properties. Some are realistic (average degree, power-law degree distribution, power-law community size distribution) but some others are not (transitivity, degree correlation and average distance). Moreover, most properties are more or less sensitive to changes in some parameters of interest such as the mixing coefficient, network size and average degree. In order to improve the realism level, we have proposed a modification of the LFR method, consisting in replacing the configuration model by alternative models. We selected the Barabási–Albert (BA) [57] and evolutionary preferential attachment (EV) [61] models, which are known to produce more realistic networks regarding the faulty properties. Note these modifications were the subject of an article we published recently [68]. We generated produced two networks collections using these modified models, and studied their properties to compare them with those of the original model. We found out our

modifications allow producing networks with comparable average distance, more realistic and stable degree correlation and more stable transitivity, compared to the original LFR method (using CM) (LFR-CM). For these properties, the EV-based model (LFR-EV) exhibits better absolute values and the BA-based (LFR-BA) one is more stable.

In the second part of our work, which we also partially published under the form of two articles [8, 69], we applied eleven different community detection algorithms on the generated collections. We have classified those algorithms according to the principle they rely on to identify communities: Radetal (RA) [19] is a divisive algorithm, Fast Greedy (FG) [31, 33], Louvain (LV) [32] and Spinglass (SG) [34] are modularity optimization algorithms, Leading Eigenvector (LEV) [37] and Commfind (CF) [40, 41] are spectral algorithms, Markov Clustering (MCL) [42], Walktrap (WT) [10] and Infomap (INP) [14] are dynamic algorithms and Label Propagation (LP) [45] and Infomod (IND) [46] use various other principles. We assessed their performances thanks to the normalized mutual information measure, already used for the same purpose in previous works [7, 49-51]. For all algorithms and on all networks, we observed the usual decrease in performance caused when increasing the mixing coefficient [7, 50, 51] (i.e. when making the communities less separated).

We have seen that on the LFR-CM collection, dynamic algorithms are one step ahead than other class of algorithms. Especially, WT and INP are the most consistent and produce the best results for all network sizes, average degree and different value of mixing coefficient. Some optimization methods work also well. SG is as consistent as WT and INP and LV can find exact community structure depending on network size. The algorithms from spectral and divisive cannot work as consistent as the first two classes. Their performance depends on the network generation parameters very much. They are not only inconstant for different network sizes and average degrees but also results worse. Other algorithms, LP and IND, exhibit different performance trends depending on network size. If we do not take into account their high standard deviation (especially for LP), we may conclude that they work well for higher network sizes. The

order of the categories does not change when we use the LFR-BA and LFR-EV collections, so we cannot say realism affects the relative performances. However, we observe significant changes for each algorithm taken individually, except for FG. This one is not sensitive at all to realism, but has the poorest results and seems a bit out of date compared to more recent algorithms. Globally, for the remaining ten sensitive algorithms, the highest performances are in general obtained when applied to the LFR-CM collection, whereas the lowest correspond to LFR-BA. Only RA and LP obtained their lowest performance with their application on the networks which are generated by LFR-EV model. The results on the LFR-EV networks are close to LFR-CM. We could not determine if the observed changes in performance were due to some network property in particular, though. LFR-BA seems to be the most interesting model in terms of discrimination of the community detection algorithms, because its stability to changes in the mixing coefficient allows to consistently compare performances for different levels of separation of the communities.

Our first goal was to improve the realism of the networks generated by the LFR method, and from this point of view the modifications we proposed were efficient. But they also resulted in a loss of control, since the replacement models (LFR-BA and LFR-EV) do not allow to specify directly as many properties as LFR-CM. For instance, in LFR-CM the exponent of the degree power-law distribution is a parameter, whereas LFR-BA can generate only networks with an exponent 3 (which, hopefully, is an extremely realistic value). We suppose it is one of the reasons why Lancichinetti *et al.* chose to base their approach on this model in the first place. Moreover, the improvement was not as strong as expected, especially concerning the transitivity, which is still very sensitive to changes in the mixing coefficient. Different ways can be explored to try to solve these limitations. First, it would be interesting to study the side effects of the rewiring process used in the LFR approach, by simply comparing the generated networks properties before and after the wiring step. This work is necessary to know if some properties observed in the final networks depend on the initial (pre-rewiring) network or on the rewiring process itself. Second, many other models exist to generate networks with power-law distributed degree [70-77]. A systematic review

could allow detecting more flexible models, offering more control on the generated networks properties, and more realistic properties. It is a long and difficult task though, because source codes are rarely easily available.

Concerning the effect of realism on community detection algorithms, our work can be extended in two ways. First, other algorithms could be applied to networks generated with the modified LFR method. Indeed, the eleven algorithms we compared exhibit rather different reactions to the models we used, making it difficult to infer general remarks concerning their effects on community detection performance. For example, we observed different behavior on modularity optimization models: FG was not sensitive to the selected generative model, whereas SG was slightly, and LV a bit more. Second, we could consider other properties to characterize networks. Maybe we did not find any strong relationships between the generated networks properties and the performance changes because we did not focus on the relevant properties. We chose the most widely used ones in the context of real-world networks analysis, but many others exist [15, 78], even if they have not been so popular up to now. Besides, these models should also be tested with higher average degrees and network sizes.

REFERENCES

- [1] Newman, M. E. J., "The structure and function of complex networks", *SIAM Review*, 45, 167-256, (2003).
- [2] Fred, A. L. N., Jain, A. K., "Robust Data Clustering", *IEEE Computer Society*, 128-136, (2003).
- [3] Newman, M. E. J., Girvan, M., "Finding and evaluating community structure in networks", *Phys Rev E*, 69, 026113, (2004).
- [4] Fortunato, S., "Community detection in graphs", *arXiv*, 0906.0612, (2009).
- [5] Girvan, M., Newman, M. E. J., "Community structure in social and biological networks", *Proceedings of the National Academy of Sciences of the United States of America*, 99, 7821-7826, Jun 11 (2002).
- [6] Bagrow, J. P., "Evaluating local community methods in networks", *Journal of Statistical Mechanics-Theory and Experiment*, May (2008).
- [7] Lancichinetti, A., Fortunato, S., Radicchi, F., "Benchmark graphs for testing community detection algorithms", *Phys Rev E*, 78, 046110, Oct (2008).
- [8] Orman, G. K., Labatut, V., "A Comparison of Community Detection Algorithms on Artificial Networks", *LNAI 5808*, Gama, J., Jorge, A., and Costa, V. S., Eds. Berlin: Springer-Verlag, 242-256, (2009).
- [9] Danon, L., Diaz-Guilera, A., Arenas, A., "The effect of size heterogeneity on community identification in complex networks", *J Stat Mech*, 11010, Nov (2006).
- [10] Pons, P., Latapy, M., "Computing communities in large networks using random walks", *arXiv*, physics/0512106, (2005).
- [11] Barabasi, A., Albert, R., "Statistical mechanics of complex networks", *Reviews of Modern physics*, 74, 47-96, (2002).
- [12] Newman, M. E. J., "Scientific collaboration networks. I. Network construction and fundamental results", *Physical Review E*, 6401, -, Jul (2001).

- [13] Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomkins, A., Wiener, J., "Graph structure in the Web", *Computer Networks*, 33, 309-320, (2000).
- [14] Rosvall, M., Bergstrom, C. T., "Maps of random walks on complex networks reveal community structure", *Proceedings of the National Academy of Sciences*, 105, 1118, (2008).
- [15] da Fontoura Costa, L., Rodrigues, F. A., Travieso, G., Villas Boas, P. R., "Characterization of complex networks: A survey of measurements", *Advances in Physics*, 56, 167-242, (2007).
- [16] Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D., "Complex networks: structure and dynamics", *Physics Reports*, 424, 175-308, (2006).
- [17] Kleinberg, J., "Authoritative sources in a hyperlinked environment", *Journal of the ACM*, 46, 604-632, (1999).
- [18] de Nooy, W., Batagelj, V., Mrvar, A., *Exploratory Social Network Analysis with Pajek*: Cambridge University Press, (2005).
- [19] Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D., "Defining and identifying communities in networks", *Proceedings of the National Academy of Sciences of the United States of America*, 101, 2658-2663, Mar 2 (2004).
- [20] Zachary, W. W., "An information flow model for conflict and fission in small groups", *Journal of Anthropological Research*, 33, 452-473, (1977).
- [21] Lusseau D., Schneider K., Boisseau O.J., Haase P., S.M., S. E. D., "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations", *Behavioral Ecology and Sociobiology*, 54, 396-405, (2003).
- [22] Travers, J., Milgram, S., "An Experimental Study of the Small World Problem", *Sociometry*, 32, 425-443, (1969).
- [23] Freeman, L. C., "Centrality in Social Networks I: Conceptual Clarification", *Social Networks*, 1, 215-239, (1979).
- [24] Fortunato, S., Barthelemy, M., "Resolution limit in community detection", *Proceedings of the National Academy of Science of the USA*, 104, 36-41, (2007).
- [25] Guimerà, R., Sales-Pardo, M., Amaral, L. A. N., "Modularity from fluctuations in random graphs and complex networks", *Physical Review E*, 70, 025101, (2004).

- [26] Leicht, E. A., Newman, M. E. J., "Community structure in directed networks", *Physical Review Letters*, 100, -, Mar 21 (2008).
- [27] Newman, M. E. J., "Analysis of weighted networks", *Physical Review E*, 70, 056131, 20 July (2004).
- [28] Arenas, A., et al., "Size reduction of complex networks preserving modularity", *New Journal of Physics*, 9, 176, (2007).
- [29] Albert, R., Jeong, H., Barabasi, A. L., "{The diameter of the world wide web}", *Nature*, 401, 130-131, (1999).
- [30] Zhang, A., *Protein Interaction Networks* Cambridge, UK: Cambridge University Press, (2009).
- [31] Newman, M. E. J., "Fast algorithm for detecting community structure in networks", *Physical Review E*, 69, -, Jun (2004).
- [32] Blondel, V. D., Guillaume, J. L., Lambiotte, R., Lefebvre, E., "Fast unfolding of communities in large networks", *Journal of Statistical Mechanics-Theory and Experiment*, -, Oct (2008).
- [33] Clauset, A., Newman, M. E. J., Moore, C., "Finding community structure in very large networks", *Physical Review E*, 70, -, Dec (2004).
- [34] Reichardt, J., Bornholdt, S., "Statistical mechanics of community detection", *Phys. Rev. E*, 74, 016110, (2006).
- [35] Kirkpatrick, S., Gelatt, C. D., Jr. Vecchi, M. P., "Optimization by Simulated Annealing", *Science*, 220, 671-680, (1983).
- [36] Blatt, M., Wiseman, S., Domany, E., "Superparamagnetic Clustering of Data", *Phys Rev Lett*, 76, 3251, (1996).
- [37] Newman, M. E. J., "Finding community structure in networks using the eigenvectors of matrices", *Physical Review E*, 74, -, Sep (2006).
- [38] Fiedler, M., "Algebraic connectivity of graphs", *Czechoslovak Math. J.*, 23, 298 - 305, (1973).
- [39] Pothen, A., Simon, H., Liou, K.-P., "Partitioning sparse matrices with eigenvectors of graphs", *SIAM J. Matrix Anal. Appl.*, 11, 430-452, (1990).
- [40] Donetti, L., Munoz, M. A., "Detecting network communities: a new systematic and efficient algorithm", *Journal of Statistical Mechanics: Theory and Experiment*, P10012, (2004).

- [41] Donetti, L., Munoz, M. A., "Improved spectral algorithm for the detection of network communities", *arXiv*, physics/0504059v1, (2005).
- [42] van Dongen, S., "Graph clustering by flow simulation", Utrecht, *University of Utrecht, Netherlands*, (2000).
- [43] Satuluri, V., Parthasarathy, S., "Scalable graph clustering using stochastic flows: applications to community discovery", *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Paris, France, 737-746, (2009).
- [44] Huffman, D. A., "A Method for the Construction of Minimum-Redundancy Codes", *Proceedings of the I.R.E.*, 1098–1102, (September 1952).
- [45] Raghavan, U. N., Albert, R., Kumara, S., "Near linear time algorithm to detect community structures in large-scale networks", *Phys. Rev. E*, 76, 036106, (2007).
- [46] Rosvall, M., Bergstrom, C. T., "An information-theoretic framework for resolving community structure in complex networks", *Proceedings of the National Academy of Sciences of the United States of America*, 104, 7327-7331, May 1 (2007).
- [47] Grünwald, P. D., Myung, I. J., Pitt, M. A., *Advances in Minimum Description Length: Theory and Applications*, (April 2005).
- [48] Kuncheva, L. I., Hadjitodorov, S. T., "Using diversity in cluster ensembles", 1214-1219, (2004).
- [49] Danon, L., Díaz-Guilera, A., Duch, J., Arenas, A., "Comparing community structure identification", *J Stat Mech*, P09008, (2005).
- [50] Lancichinetti, A., Fortunato, S., "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities", *Phys. Rev. E*, 80, 016118, (2009).
- [51] Lancichinetti, A., Fortunato, S., "Community detection algorithms: a comparative analysis", *arXiv*, 0908.1062, (2009).
- [52] Erdos, P., Renyi, A., "On random graphs", *Publicationes Mathematicae*, 6, 290-297, (1959).
- [53] Bollobás, B., "Degree sequences of random graphs", *Discrete Mathematics*, 33, 1-19, (1981).

- [54] Bender, E. A., Canfield, E. R., "The asymptotic number of labeled graphs with given degree sequences", *Journal of Combinatorial Theory, Series A*, 24, 296-307, (1978).
- [55] Molloy, M., Reed, B., "A critical point for random graphs on a fixed degree sequence", *Random Structures and Algorithms*, 6, 161-180, (1995).
- [56] Molloy, M., Reed, B., "The size of the giant component of a random graph with a given degree sequence", *Combinatorics Probability & Computing*, 7, 295-305, Sep (1998).
- [57] Barabasi, A., Albert, R., "Emergence of scaling in random networks", *Science*, 286, 509, (1999).
- [58] Bollobás, B., Riordan, O., "The Diameter of a Scale-Free Random Graph", *Combinatorica*, 24, 5-34, (2004).
- [59] Dorogovtsev, S. N., Mendes, J. F. F., Samukhin, A. N., "Size dependent degree distribution of a scale-free growing network", *Phys. Rev. E*, 63, 062101, (2001).
- [60] Dorogovtsev, S. N., Mendes, J. F. F., "Evolution of networks", *Advances in Physics*, 51, 1079-1187, (2002).
- [61] Poncela, J., Gomez-Gardeñes, J., Floria, L. M., Sanchez, A., Moreno, Y., "Complex Cooperative Networks from Evolutionary Preferential Attachment", *PLoS ONE*, 3, e2449, (2008).
- [62] Szabó, G., Fáth, G., "Evolutionary games on graphs", *Physics Reports*, 446, 97-216, (2007).
- [63] Clauset, A., "Finding local community structure in networks", *Phys Rev E*, 72, 026132, (2005).
- [64] da Fontoura Costa, L., Oliveira Jr., O. N., Travieso, G., Rodrigues, r. A., Villas Boas, P. R., Antiqueira, L., Viana, M. P., da Rocha, L. E. C., "Analyzing and Modeling Real-World Phenomena with Complex Networks: A Survey of Applications", *arXiv*, 0711.3199, (2008).
- [65] Serrano, M., Boguñá, M., "Weighted Configuration Model", *AIP Conference Proceedings*, 776, 101, (2005).
- [66] Chung, F., Lu, L., "The average distances in random graphs with given expected degrees", *PNAS* 99, 15879-15882, (2002).
- [67] Molloy, M., Reed, B., "A critical point for random graphs with a given degree sequence", *Random Structures and Algorithms*, 6, 161-179, (1995).

- [68] Orman, G. K., Labatut, V., "The effect of Network Realism on Community Detection Algorithms", *ASONAM*, Odense, Denmark, (2010).
- [69] Orman, G. K., Labatut, V., "Relative Evaluation of Partition Algorithms for Complex Networks", *1st International Conference on Networked Digital Technologies*, Ostrava, CZ, (2009).
- [70] Chen, Q., Chen, S., "A highly clustered scale-free network evolved by random walking", *Physica A*, 383, 773-781, (2007).
- [71] Chuang, L., Jian-Yuan, J., Xiao-Jie, C., Rui, C., Long, W., "Prisoner's Dilemma Game on Clustered Scale-Free Networks under Different Initial Distributions", *Chin. Phys. Lett.*, 26, 080202, (2009).
- [72] Dangalchev, C., "Generation models for scale-free networks", *Physica a-Statistical Mechanics and Its Applications*, 338, 659-671, Jul 15 (2004).
- [73] Holme, P., Kim, B. J., "Growing scale-free networks with tunable clustering", *Phys Rev E*, 65, 026107, (2002).
- [74] Klemm, K., Eguiluz, V. M., "Growing scale-free networks with small-world behavior", *Phys Rev E*, 65, (2002).
- [75] Saramäki, J., Kaski, K., "Scale-free networks generated by random walkers", *Physica A*, 341, 80-86, (2004).
- [76] Shi, D., Zhu, X., Liu, L., "A Simple Model of Scale-free Networks Driven by Both Randomness and Adaptability", *arXiv*, physics/0409061v2, (2004).
- [77] Tam, W. M., Lau, F. C. M., Tse, C. K., "Construction of Scale-Free Networks with Adjustable Clustering", Budapest, Hungary, (2008).
- [78] Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U., "Network motifs: simple building blocks of complex networks", *Science*, 298, 824-827, (2002).

BIOGRAPHICAL SKETCH

Keziban Orman was born in Ankara in 1982. She graduated from computer engineering department of the Galatasaray University in 2005. In the same year, she started to work in Türkiye İş Bankası as database administrator and she worked there until she started to work in TÜBİTAK (National Research Center of Turkey) in 2006 November. She worked on database and datamining projects there. When she was working there, she started her master of science in Computer Engineering Program of the Institute of Science and Technology of Galatasaray University in 2007. She has worked as research assistant in Computer Engineering Department at Galatasaray University since December 2009.