2009

Models for On-line Social Networks

Noor Hadi
Wilfrid Laurier University

Your file Votre référence
ISBN: 978-0-494-49981-8
Our file Notre référence
ISBN: 978-0-494-49981-8

Models for On-line Social Networks

by

Noor Hadi

(BSc, American University of Sharjah, 2007)

THESIS

Submitted to the Department/Faculty of Mathematics

in partial fulfilment of the requirements for

Master of Science in Mathematics

Wilfrid Laurier University

Abstract

On-line social networks such as Facebook or Myspace are of increasing interest to computer scientists, mathematicians, and social scientists alike. In such real-world networks, nodes represent people and edges represent friendships between them. Mathematical models have been proposed for a variety of complex real-world networks such as the web graph, but relatively few models exist for on-line social networks.

We present two new models for on-line social networks: a deterministic model we call Iterated Local Transitivity (ILT), and a random ILT model. We study various properties in the deterministic ILT model such as average degree, average distance, and diameter. We show that the domination number and cop number stay the same no matter how many nodes or edges are added over time. We investigate the automorphism groups and eigenvalues of graphs generated by the ILT model. We show that the random

ILT model follows a power-law degree distribution and we provide a theorem about the power law exponent of this model. We present simulations for the degree distribution of the random ILT model.

Acknowledgements

I would like to express my gratitude to my supervisor, Dr. Anthony Bonato, for introducing me to the world of Graph Theory and for his continuous support, guidance, patience and encouragement. I appreciate all the time he spent making this thesis a valuable experience for me.

I extend my thanks to the Department of Mathematics at Wilfrid Laurier University, and especially to Dr. Sydney Bulman-Fleming, Dr. Ross Cressman, June Aleong, and Tao Gong. I thank all the other professors who I met and were influenced by during my graduate studies: Dr. Roderick Melnik, Dr. George Lai, Dr. Roman Makarov, and Dr. Changping Wang. A special thanks to Dr. Dejan Delić, Dr. Connell McCluskey, and Dr. Manuele Santoprete for being part of my thesis committee.

Last but not least, I thank my parents for their constant support in all aspects of my life and especially during my graduate studies. Without them, I would have never been able to reach to this phase of my life. A special thanks to my brother Hussein whose presence has always brightened my life. I finally thank all my friends for being there for me when I needed them.

Contents

List of Figures

CHAPTER 1

Introduction

1.1. Motivation

The popularity of on-line social networks like Facebook, MySpace, and Orkut has increased dramatically over recent years. These networks are modelled by undirected graphs where nodes represent people and edges represent friendship between them (we always assume such networks are *undirected*: if x is friends with y, then y is friends with x). In these massive real-world networks with millions of nodes and edges, new nodes and edges appear over time. There has been increasing interest in the mathematical and general scientific community in such networks, in both gathering data and statistics about the networks, and in finding accurate and rigorous models simulating their evolution. As a small snapshot of one of these networks, Figure 1.1 shows the subgraph induced by my friends on Facebook, generated using the *Nexus* application.

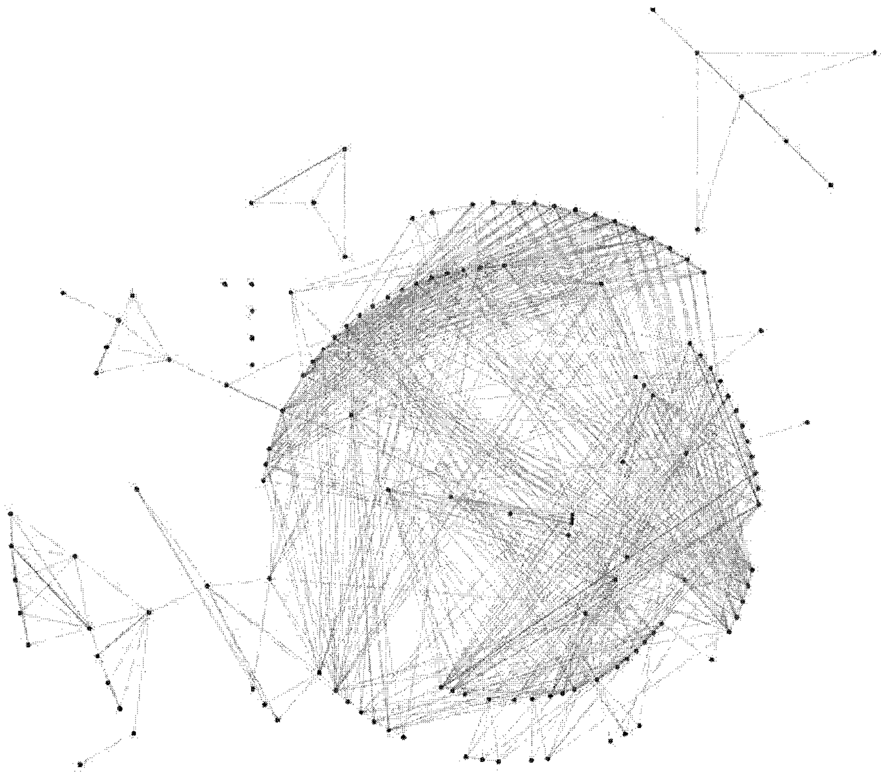


Figure 1.1: Subgraph induced by the neighbours of the Noor Hadi node on Facebook.

A central idea in complex networks is the notion of the small world property which was introduced by Watts and Strogatz [**18**], and has roots in the work of Milgram [**15**] which suggests short paths of friends connecting strangers. The small world property demands low average distance (or diameter) and high clustering, and has been observed in a wide variety of complex networks. For more on the small world property and other properties of complex networks, see [**6**].

Many recent studies have analyzed on-line social networks focusing on the small world property and other complex network properties seen in on-line social networks. Kumar et al. [**12**] studied the evolution of the on-line networks Flickr and Yahoo!360. They found that the average distance between users decreases over time, implying that these networks have the small world property. They also found that they exhibit power-law degree distributions. Golder et al. [**11**] analyzed the Facebook network by studying the messaging pattern between friends. They also found a power law degree distribution and the small world property. Similar results were found in [**1**] which studied Cyworld, MySpace, and Orkut, and in [**4**] which examined data collected from four on-line social networks: Flickr, YouTube, LiveJournal, and Orkut.

In this thesis, we aim to develop mathematical models that dynamically simulate the on-line social networks and possess the aforementioned properties. We propose two models: a deterministic model and a random one.

The deterministic model, which we call the *Iterated Local Transitivity (ILT)* model, relies on the idea of what sociologists call *transitivity*: if u is a friend of v, and v is a friend of w, then u is a friend of w (see [**9, 16, 20**]). Figure 1.2 shows an example of transitivity. In its simplest form, transitivity

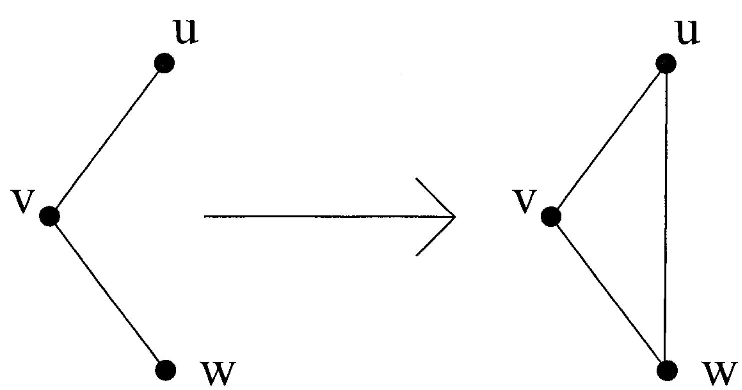


Figure 1.2: An example of transitivity.

gives rise to the notion of *cloning*, where u is joined to all of the neighbours of v. In the ILT model, given some initial graph as a starting point, nodes are repeatedly added over time which clone each node, so that the new nodes formed have no edges between them. The ILT model uses only local knowledge in its evolution, in that a new node only joins to neighbours of an existing node. Local knowledge is an important feature of social and complex networks, where nodes have only limited influence on the network topology.

The random ILT model performs at each step, with certain probability, a cloning operation or a preferential attachment operation. All nodes of the initial graph are assigned probabilities depending on their degrees. The higher the degree of a node, the higher the probability that it would be chosen. Cloning occurs in a similar way as in the deterministic ILT model; however, one existing node is chosen uniformly at random and only this node is cloned. In the preferential attachment step, a node is chosen randomly giving preference to those with higher degrees and a new node is created and joined only to the randomly chosen node.

1.2. Graph Theory

In this section, we introduce various graph theoretical terminologies and concepts used throughout the thesis. A *graph* or *undirected graph* G consists of a non-empty node set $V(G)$, and an edge set $E(G)$ of 2-element sets from $V(G)$. More formally, we may consider $E(G)$ as a binary relation on $V(G)$ which is irreflexive and symmetric. The graphs we consider are finite, undirected, and simple (no loops nor multiple edges). A graph is also sometimes called a *network*, especially with regards to real-world examples. We often write $G = (V(G), E(G))$, or if G is clear from the context, $G = (V, E)$. Elements of $V(G)$ are *vertices*, and elements of $E(G)$ are *edges*. Vertices are also often referred to as *nodes*. We write uv for an edge u, v, and say that u and v are *joined* or *adjacent*; we say that u and v are *incident* to the edge uv, and that u and v are the *endpoints* of uv. Graphs are usually visualized by simply drawing dots to represent nodes and lines to represent edges. The cardinality $|V(G)|$ is the *order* of G, while $|E(G)|$ is its *size*. For a node $v \in V(G)$, $\deg_G(v)$ is the degree of v in G; namely the number of edges in G incident with v. For example, in Figure 1.3 the 4-cycle C_4 has order 4, size 4, and the degree of each node is 2. We often drop the subscript G if it is clear from context.

We mention the so-called *First Theorem of Graph Theory* which says the following.

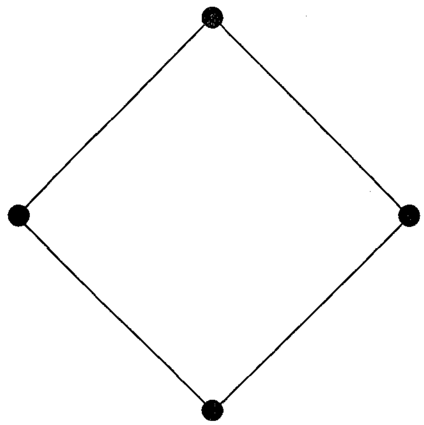


Figure 1.3: The graph C_4.

THEOREM 1.1. *If G is a graph, then*

$$2|E(G)| = \sum_{u \in V(G)} \deg_G(u).$$

A *path* is defined as an open walk with no repeated node. A *complete graph of order n* or *n-clique* has all edges present, and is written K_n. A graph is *connected* if for each pair of nodes there is a path between them. Given a node u, define its *neighbour set* $N(u)$ to be the set of nodes joined to u (also called *neighbours* of u). The distance between u and v, written $d(u, v)$, is either the length of a shortest path connecting u and v (and 0 if $u = v$) or ∞ otherwise. The *diameter* of a connected graph G, written diam(G), is the maximum of all distances between distinct pairs of nodes.

In a graph G, a set S of nodes is a *dominating* set if every node not in S has a neighbour in S. The *domination number* of G, $\gamma(G)$, is the minimum cardinality of a dominating set in G. We use S to represent a dominating set in G, where each node not in S is joined to some node of S.

A graph parameter related to the domination number is the so-called cop (or search) number of a graph. The game of Cops and Robber is a node pursuit game played on a graph G. There are two players, a set of k cops (or searchers) C , where $k > 0$ is a fixed integer, and the robber R. The cops begin the game by occupying a set of k nodes, and the cops and robber move in alternate rounds. More than one cop is allowed to occupy a node, and the players may *pass*; that is, remain on their current node. The players know each others current locations and can remember all the previous moves; that is, the game is played with perfect information. The cops win and the game ends if at least one of the cops can eventually occupy the same node as the robber; otherwise, R wins. A winning strategy for

$|V(G)|$ cops is to occupy each node of G. Based on this, the *cop number*, written $c(G)$, is defined as the minimum number of cops needed to win on G. Note that

$$c(G) \leq \gamma(G),$$

since placing a cop on each node of a dominating set ensures that the cops win in at most one move.

The *Wiener index* of a connected graph G, written $W(G)$, is defined as

$$W(G) = \sum_{x,y \in V(G)} d(x,y),$$

where $d(x,y)$ is the distance between any two distinct nodes. The *Wiener index* arises in applications of graph theory to Chemistry (see [**19**]), and may be used to define the *average distance* of G as

$$L(G) = \frac{W(G)}{\binom{n}{2}},$$

where n is the order of G.

A *subgraph* of G is a graph H such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. If $S \subseteq V$, then the subgraph induced by S, written as $G \restriction S$, is defined as the graph with nodes

S and with two nodes joined in $G \restriction S$ if and only if they are joined in G.

A *homomorphism* f between graphs G and H is a function $f : V(G) \to V(H)$ which *preserves edges*; that is, if $xy \in E(G)$, then $f(x)f(y) \in E(H)$. We abuse notation and simply write $f : G \to H$. An *embedding* from G to H is an injective homomorphism $f : G \to H$ with the property that $xy \in E(G)$ if and only if $f(x)f(y) \in E(H)$. An *isomorphism* is a bijective embedding; if there is an isomorphism between two graphs, then we say they are *isomorphic.* If graphs G and H are isomorphic, then we write $G \cong H$. An *automorphism* of a graph G is an isomorphism from G to itself; the set of all automorphisms forms a group under the operation of composition, written Aut(G).

As the results we present are sometimes asymptotic (especially in Chapter 4), we give some notation. Let f and g be functions whose domain is some fixed subset of $\mathbb{R}$. We write $f \in O(g)$ if

$$\lim_{x\to\infty} \frac{f(x)}{g(x)}$$

exists and is finite. We will abuse notation and write $f = O(g)$. We write $f = \Omega(g)$ if $g = O(f)$, and $f = \Theta(g)$ if

$f = O(g)$ and $f = \Omega(g)$. If

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} = 0,$$

then $f = o(g)$ (or $g = \omega(f)$). So if $f = o(1)$, then f tends to 0. We write $f \sim g$ if

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} = 1.$$

An important property of many complex networks is the presence of power-law degree distributions. Given a graph G and a non-negative integer k, we define $N_{k,G}$ by

$$N_{k,G} = |\{x \in V(G) : \deg_G(x) = k\}|.$$

The parameter $N_{k,G}$ is the *number of nodes of degree k in G*. The *degree distribution* of G is the sequence

$$(N_{k,G} : 0 \leq k \leq t),$$

where t is the order of the graph G. The degree distribution of G follows a *power law* if for each degree k,

$$\frac{N_{k,G}}{t} \sim k^{-\beta},$$

for a fixed real constant $\beta > 1$. We say that β is the *exponent of the power law.* A graph whose degree distribution follows a power law is often referred to as a *power law graph.* Figure 1.4 shows an example of the degree distribution of the set of neighbours of the Noor Hadi node on Facebook.

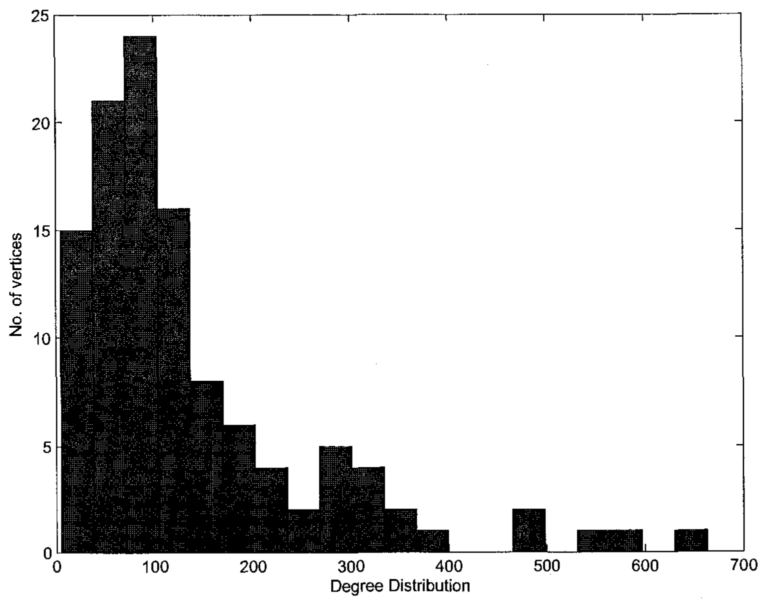


Figure 1.4: Degree distribution of the neighbour set of the Noor Hadi node on Facebook.

1.3. Linear Algebra

Graphs are often represented by adjacency matrices. Let G have vertices $1, 2, \ldots, n$. The *adjacency matrix,* written

$A(G)$, of the graph G is the $n \times n$ matrix defined by

$$A(G)_{ij} = \begin{cases} 1 & \text{if } ij \in E(G), \\ 0 & \text{otherwise.} \end{cases}$$

Adjacency matrices are non-negative, symmetric and have zeros on the main diagonal. Several graph parameters can be read off from the adjacency matrix. For example, the degree of a node in a graph can be found by summing either the column or row of an adjacency matrix, while the size of a graph can be found from an adjacency matrix by summing all the ones in the matrix and dividing by 2. As an example, the adjacency matrix for C_4 is

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}.$$

For a square matrix A, a scalar λ for which

$$\det(A - \lambda I) = 0$$

is called an *eigenvalue* of A. The eigenvalues for $A(C_4)$ by direct checking are $\{-2, 0, 2\}$.

1.4. Probability

We provide some background on elementary probability theory. For additional background, see [**10**]. A *(discrete) probability space* $\mathcal{S}$ consists of a triple $(S, \mathcal{F}, \mathbb{P})$. The set S, called the *sample space*, is nonempty and finite. For us the set $\mathcal{F}$ is the collection of all subsets of S; the elements of $\mathcal{F}$ are *events*. The function $\mathbb{P} : \mathcal{F} \to \mathbb{R}$, named the *probability measure*, satisfies the following properties.

(1) For all events A, $\mathbb{P}(A) \in [0, 1]$, and $\mathbb{P}(S) = 1$.

(2) If $(A_i : i \in I)$ is a countable set of events that are pairwise disjoint, then

$$\mathbb{P}\left(\bigcup_{i \in I} A_i\right) = \sum_{i \in I} \mathbb{P}(A_i).$$

In a probability space with $|S| = n$ a positive integer, an element chosen with probability $\frac{1}{n}$ from S is said to be chosen *uniformly at random*, also written *u.a.r.* A *random variable* X on a probability space $\mathcal{S}$ is a function $X : S \to \mathbb{R}$. The *expectation* of a random variable X, written $\mathbb{E}(X)$, is defined by

$$\mathbb{E}(X) = \sum_{s \in S} X(s)\mathbb{P}(\{s\}).$$

Note that $\mathbb{E}(X)$ is always finite. If $X \geq 0$, then $\mathbb{E}(X) \geq 0$. An important property of expectation is the *Linearity of Expectation.*

THEOREM 1.2. *Suppose that X is a random variable defined on a probability space. Let c_i, where $1 \leq i \leq n$, be real numbers. Then,*

$$\mathbb{E}\left(\sum_{i=1}^{n} c_i X_i\right) = \sum_{i=1}^{n} c_i \mathbb{E}\left(X_i\right).$$

We also use the notion of conditional expectation. Let X, Y be random variables on a common probability space. The *conditional mass function* of X given $Y = y$, written $f_{X|Y}(\cdot|y)$, is defined as

$$f_{X|Y}(x|y) = \mathbb{P}(X = x|Y = y),$$

for all y such that $\mathbb{P}(Y = y) > 0$. Given $Y = y$, we may think of $f_{X|Y}(x|y)$ as a function of x. The expected value of this distribution, which is

$$\sum_x x f_{X|Y}(x|y)$$

is the *conditional expectation of* X *when* $Y = y$, and is written

$$\mathbb{E}[X|Y = y].$$

Define $g(y) = \mathbb{E}[X|Y = y]$. The function g is the *conditional expectation of* X *on* Y, written $\mathbb{E}[X|Y]$. Note that $\mathbb{E}[X|Y]$ is a random variable, and so has an expected value. Intuitively, $\mathbb{E}[X|Y]$ is the expected value of X assuming Y is known. It can be shown that (see [**10**])

$$\mathbb{E}(\mathbb{E}[X|Y]) = \mathbb{E}(X).$$

1.5. Outline of Thesis

The remainder of this thesis is organized as follows. In Chapter 2, we introduce the ILT model. We will provide results on the following parameters and properties for graphs generated by the ILT model : order, size, average degree and densification. We will also show the results for the degree distribution from some simulations of the model. In

Chapter 3, we consider further properties of the ILT model: average distance, cop number and spectral properties. In Chapter 4, we introduce the random ILT model and analyze its degree distribution. In Chapter 5, we state some open problems related to this thesis.

We note that the results of this thesis are original work. Parts of Chapters 2 and 3 were included in the accepted paper [**7**].

CHAPTER 2

The Deterministic ILT model

2.1. Introduction

The (deterministic) Iterated Local Transitivity (ILT) model generates simple, undirected graphs $(G_t : t \geq 0)$ over a countably infinite sequence of discrete time-steps. The only parameter of the model is the initial graph G_0, which is any fixed finite connected graph. At $t+1$, all nodes in $V(G_t)$ are "cloned", in the sense that for every $x \in V(G_t)$ there is an $x' \in V(G_{t+1})$ that is connected to x and all of its neighbours. Note that all the new nodes created at $t+1$ form an independent set (that is, contains no edges) of cardinality $|V(G_t)|$. The idea of cloning is analogous to how on-line social networks grow over time. At a specific time t, let G_t represent the graph of an on-line social network. At $t+1$, a new user y joins the network and finds his friend, say x, and becomes a friend with him. Now using the idea of transitivity, y also becomes friends with the friends of x. Hence, the phenomenon of cloning naturally arises in real-world on-line social networks.

Let $\deg_t(x)$ be the degree of a vertex x at time t. The important recurrences governing the degrees of nodes are given as

$$\begin{aligned} \deg_{t+1}(x) &= 2\deg_t(x) + 1, && (2.1) \\ \deg_{t+1}(x') &= \deg_t(x) + 1. && (2.2) \end{aligned}$$

Equation (2.1) comes from the fact that each neighbour of x contributes a new edge to x at time $t+1$; hence, adding another $\deg(x)$ to the degree of x, and x' connects to x giving

$$\deg_{t+1}(x) = \deg_t(x) + \deg_t(x) + 1 = 2\deg_t(x) + 1.$$

Equation (2.2) comes from the fact that the new node x' connects to all the neighbours of x and to x itself. Hence,

$$\deg_{t+1}(x') = \deg_t(x) + 1.$$

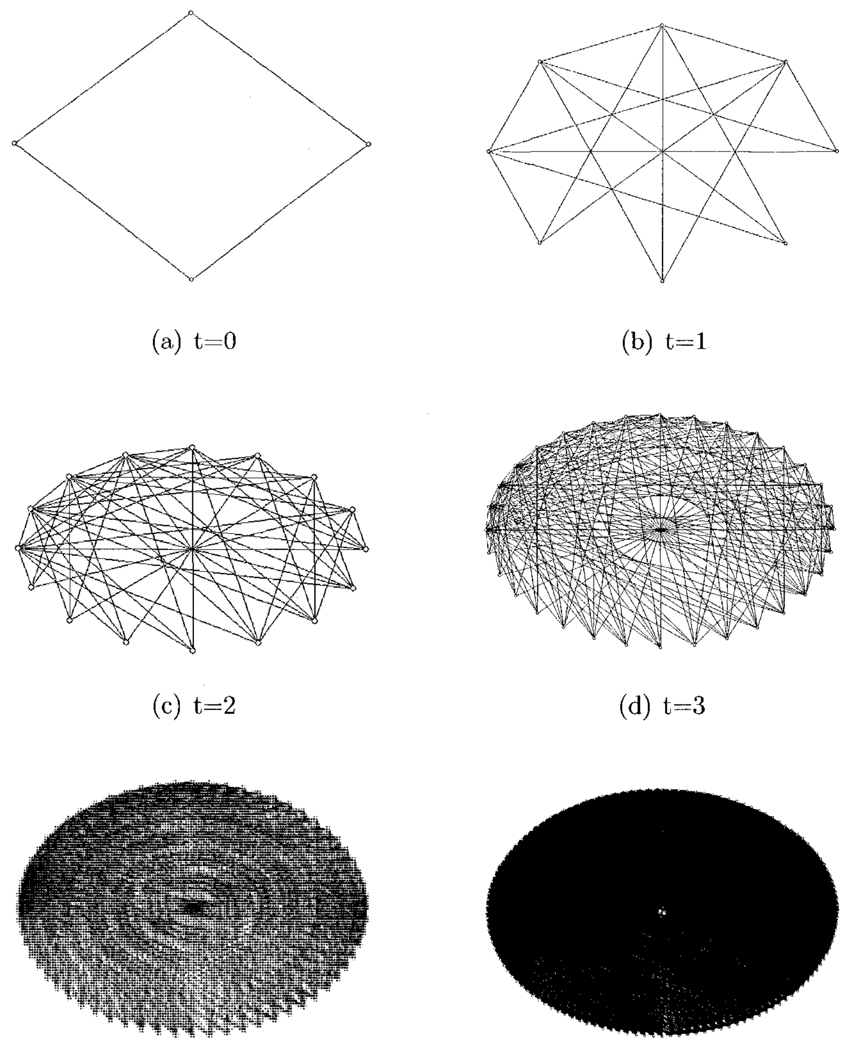
As an example of the evolution of the graphs in the ILT model starting with the 4-cycle C_4 graph, see Figure 2.1.

We use n_t to denote the order of G_t, and e_t to denote its size. We now derive the order of the graph at time t.

THEOREM 2.1. *For* $t \geq 0$, $n_t = 2^t n_0$

Proof. We proceed by induction on $t \geq 0$. If $t = 0$, then $n_0 = 2^0 n_0$. As the induction hypothesis, for a fixed $t \geq 0$ set $n_t = 2^t n_0$. Now for n_{t+1}, note that G_{t+1} doubles its order at time $t+1$. In other words, $n_{t+1} = 2n_t$. Hence,

$$n_{t+1} = 2n_t = 2(2^t n_0) = 2^{t+1} n_0. \qquad \square$$



(a) t=0 (b) t=1

(c) t=2 (d) t=3

(e) t=4 (f) t=5

Figure 2.1: The time-steps with $G_0 = C_4$, for $t = 0, 1, 2, 3, 4, 5$.

2.2. Size and Average Degree

Recent work by Leskovec et al. [**14**] underscores the importance of two additional properties of complex networks above and beyond more traditionally studied phenomena such as the small world property. A graph G with e_t edges and n_t nodes satisfies a densification power law if there is a constant $a \in (1, 2]$ such that $e_t \sim {n_t}^a$ (a is called the *exponent* of the power law). In particular, the average degree grows to infinity with the order of the network. In [**14**], densification power laws were reported in several real-world networks such as the physics citation graph, and the internet graph at the level of autonomous systems. We show that the ILT model follows a densification power law making the ILT model more realistic, especially in light of real-world data mined from complex networks.

Define the *volume* of G_t by

$$\mathrm{vol}(G_t) = \sum_{x \in V(G_t)} \deg_t(x) = 2e_t. \tag{2.3}$$

We find a formula for the volume of G_t by exploiting the following recurrence.

LEMMA 2.2. *For $t \geq 0$,*

$$\mathrm{vol}(G_{t+1}) = 3\mathrm{vol}(G_t) + n_{t+1}.$$

Proof. From Equations (2.1) and (2.2),

$$\begin{aligned}
\mathrm{vol}(G_{t+1}) &= \sum_{x\in V(G_t)} \deg_{t+1}(x) + \sum_{x'\in V(G_{t+1})\setminus V(G_t)} \deg_{t+1}(x') \\
&= \sum_{x\in V(G_t)} (2\deg_t(x)+1) + \sum_{x\in V(G_t)} (\deg_t(x)+1) \\
&= (2(2e_t)+n_t) + (2e_t+n_t) \\
&= 6e_t + 2n_t \\
&= 3\mathrm{vol}(G_t) + n_{t+1}. \qquad \square
\end{aligned}$$

We now give a precise formula for the volume of G_t.

THEOREM 2.3. *For* $t > 0$

$$\mathrm{vol}(G_t) = 3^t\mathrm{vol}(G_0) + 2n_0(3^t - 2^t).$$

Proof. We prove the theorem by induction on $t \geq 0$. As the base step, we have that

$$\mathrm{vol}(G_1) = 3\mathrm{vol}(G_0) + 2n_0.$$

As an induction hypothesis, for $t \geq 0$ fixed, set

$$\mathrm{vol}(G_t) = 3^t\mathrm{vol}(G_0) + 2n_0(3^t - 2^t).$$

Now at time $t+1$,

$$\begin{aligned}
\mathrm{vol}(G_{t+1}) &= 3\mathrm{vol}(G_t) + n_{t+1} \\
&= 3\mathrm{vol}(G_t) + 2^{t+1}n_0 \\
&= 3(3^t\mathrm{vol}(G_0) + 2n_0(3^t - 2^t)) + 2^{t+1}n_0 \\
&= 3^{t+1}\mathrm{vol}(G_0) + 3^1 2^1 n_0(3^t - 2^t) + 2^{t+1}n_0 \\
&= 3^{t+1}\mathrm{vol}(G_0) + 2n_0(3^{t+1} - 2^1(2^t)) \\
&= 3^{t+1}\mathrm{vol}(G_0) + 2n_0(3^{t+1} - 2^{t+1}).
\end{aligned}$$

Hence, by induction on t, we have that

$$\mathrm{vol}(G_t) = 3^t\mathrm{vol}(G_0) + 2n_0(3^t - 2^t). \qquad \square$$

We provide the formula for the average degree of a graph G_t.

THEOREM 2.4. *For $t > 0$, the average degree of G_t, written* $\deg_{\mathrm{ave}}(G_t)$, *equals*

$$\left(\frac{3}{2}\right)^t \left(\frac{\mathrm{vol}(G_0)}{n_0} + 2\right) - 2.$$

Proof. By Theorem 2.3 we have that

$$\begin{aligned}\mathrm{deg}_{\mathrm{ave}}(G_t) &= \frac{\mathrm{vol}(G_t)}{n_t}\\ &= \frac{3^t\mathrm{vol}(G_0) + 2n_0(3^t - 2^t)}{2^t n_0}\\ &= \left(\frac{3}{2}\right)^t \left(\frac{\mathrm{vol}(G_0)}{n_0} + 2\right) - 2. \qquad \square\end{aligned}$$

We can now determine the size e_t of G_t using the fact that

$$e_t = \frac{\mathrm{vol}(G_t)}{2}.$$

LEMMA 2.5. *For* $t \geq 0$,

$$e_t = 3^t(e_0 + n_0) - n_t.$$

Proof. By Theorem 2.3 we have that

$$\begin{aligned}e_t &= \frac{\mathrm{vol}(G_t)}{2}\\ &= \frac{3^t 2e_0 + 2n_0(3^t - 2^t)}{2}\\ &= 3^t(e_0 + n_0) - n_t. \qquad \square\end{aligned}$$

Note that Lemma 2.5 and Theorem 2.4 supplies a densification power law with exponent $a = \frac{log3}{log2} \approx 1.58$.

2.3. Average Distance, Diameter, and Degree Distribution

Define the *Wiener index* of G_t as

$$W(G_t) = \sum_{x,y \in V(G_t)} d_t(x,y).$$

The Wiener index arises in applications of graph theory to Chemistry [**19**], and may be used to define the *average distance* of G_t as

$$L(G_t) = \frac{W(G_t)}{\binom{n_t}{2}}.$$

We will compute the average distance by deriving first the Wiener index. Define the *ultimate average distance of* G_0, as

$$UL(G_0) = \lim_{t \to \infty} L(G_t)$$

assuming the limit exists. We provide an exact value for $L(G_t)$ and compute the ultimate average distance for any initial graph G_0. An important lemma about distances between the nodes in a graph G_t will help us compute the recurrence for the Wiener index.

LEMMA 2.6. *Let x and y be nodes in G_t with $t > 0$. Then*

$$d_{t+1}(x', y) = d_{t+1}(x, y') = d_{t+1}(x, y) = d_t(x, y),$$

and

$$d_{t+1}(x', y') = \begin{cases} d_t(x, y) & \text{if } xy \notin E(G_t), \\ d_t(x, y) + 1 = 2 & \text{if } xy \in E(G_t). \end{cases}$$

Proof. We prove that $d_{t+1}(x, y) = d_t(x, y)$. The proofs that $d_{t+1}(x, y') = d_t(x, y)$, $d_{t+1}(x', y) = d_t(x, y)$, and $d_{t+1}(x', y') = d_t(x, y)$ if x and y are not joined are analogous and so omitted. Since in the ILT model we do not delete any edges, the distance cannot increase after a "cloning" step occurs. Hence, $d_{t+1}(x, y) \leq d_t(x, y)$. Now suppose for a contradiction that there is a path P' connecting x and y in G_{t+1} with length $k < d_t(x, y)$. Hence, P' contains nodes not in G_t. Choose such a P' with the least number of nodes, say $s > 0$, not in G_t. Let z' be a node of P' not in G_t, and let the neighbours of z' in P' be u and v. Then $z \in V(G_t)$ is joined to u and v. Form the path Q' by replacing z' by z. But then Q' has length k and has $s - 1$ many nodes not in G_t, which supplies a contradiction.

In the case where $xy \in E(G_t)$, we have

$$\begin{aligned} d(x', y') &= d(x', y) + d(y, y') \\ &= d(x, y) + 1 \\ &= 1 + 1 = 2. \qquad \square \end{aligned}$$

We now give the recurrence for the Wiener index.

THEOREM 2.7. *For* $t > 0$,

$$W(G_t) \quad = \quad 4^t \left(W(G_0) + (e_0 + n_0)\left(1 - \left(\frac{3}{4}\right)^t\right)\right).$$

Proof. To compute $W(G_{t+1})$, there are five cases to be considered: distances within G_t, and distances of the forms: $d_{t+1}(x, y')$, $d_{t+1}(x', y)$, $d_{t+1}(x, x')$, and $d_{t+1}(x', y')$. The first three cases contribute $3W(G_t)$ by Lemma 2.6. The 4th case contributes n_t. The final case contributes $W(G_t) + e_t$ (the term e_t comes from the fact that each edge xy contributes

$d_t(x,y)+1)$. Hence,

$$
\begin{aligned}
W(G_{t+1}) &= \sum_{x,y\in V(G_t)} d_{t+1}(x,y) \\
&= \sum_{x,y\in V(G_t)} d_t(x,y) + \sum_{\substack{x\in V(G_t),\\ y'\in V(G_{t+1})}} d_{t+1}(x,y') \\
&+ \sum_{\substack{x'\in V(G_{t+1}),\\ y\in V(G_t)}} d_{t+1}(x',y) + \sum_{\substack{x\in V(G_t),\\ x'\in (G_{t+1})}} d_{t+1}(x,x') \\
&+ \sum_{x',y'\in V(G_{t+1})} d_{t+1}(x',y') \\
&= W(G_t) + \sum_{x,y\in V(G_t)} d_t(x,y) + \sum_{x,y\in V(G_t)} d_t(x,y) + n_t \\
&+ \sum_{x,y\in V(G_t)} (d_t(x,y)) + e_t \\
&= 4W(G_t) + e_t + n_t.
\end{aligned}
$$

By Lemma 2.5 we have that

$$
\begin{aligned}
W(G_{t+1}) &= 4W(G_t) + 3^t(e_0+n_0) - n_t + n_t \\
&= 4W(G_t) + 3^t(e_0+n_0). \qquad (2.4)
\end{aligned}
$$

Now we prove the final recurrence for $W(G_t)$ by induction. As the base step, using (2.4) we have that

$$W(G_1) = 4W(G_0) + e_0 + n_0.$$

As the induction hypothesis, for a fixed $t \geq 1$ we set

$$W(G_t) = 4^t W(G_0) + 4^t(e_0 + n_0)\left(1 - \left(\frac{3}{4}\right)^t\right)$$

At time $t+1$, we have that

$$\begin{aligned}
W(G_{t+1}) &= 4W(G_t) + 3^t(e_0 + n_0) \\
&= 4\left(4^t W(G_0) + 4^t(e_0 + n_0)\left(1 - \left(\frac{3}{4}\right)^t\right)\right) \\
&\quad + 3^t(e_0 + n_0) \\
&= 4^{t+1}W(G_0) + (4^{t+1} - 3^t 4^{-t} 4^{t+1})(e_0 + n_0) + 3^t e_0 \\
&\quad + 3^t n_0 \\
&= 4^{t+1}W(G_0) + 4^{t+1}(e_0 + n_0) - 3^{t+1}(e_0 + n_0) \\
&= 4^{t+1}W(G_0) + (e_0 + n_0)(4^{t+1} - 3^{t+1}) \\
&= 4^{t+1}W(G_0) + 4^{t+1}(e_0 + n_0)\left(1 - \left(\frac{3}{4}\right)^{t+1}\right).
\end{aligned}$$

Hence, by induction for all $t \geq 1$ we have that

$$W(G_t) = 4^t W(G_0) + 4^t(e_0 + n_0)\left(1 - \left(\frac{3}{4}\right)^t\right). \qquad \square$$

We now state the theorems for average distance and ultimate average distance for graphs generated by the ILT model.

THEOREM 2.8. *For $t > 0$,*

$$L(G_t) = 2\left(\frac{4^t\left(W(G_0) + (e_0 + n_0)\left(1 - \left(\frac{3}{4}\right)^t\right)\right)}{4^t n_0^2 - 2^t n_0}\right).$$

Proof. We have by Theorem 2.7 that

$$\begin{aligned} L(G_t) &= W(G_t)\binom{n_t}{2}^{-1} \\ &= \frac{2W(G_t)}{(n_t)^2 - n_t} \\ &= \frac{2(4^t)\left(W(G_0) + (e_0 + n_0)\left(1 - \left(\frac{3}{4}\right)^t\right)\right)}{4^t(n_0)^2 - 2^t n_0}. \qquad \square \end{aligned}$$

THEOREM 2.9. *For all graphs G_0,*

$$UL(G_0) = \frac{2(W(G_0) + e_0 + n_0)}{n_0^2}.$$

Proof. By Theorem 2.8 it follows that

$$\begin{aligned} UL(G_0) &= \lim_{t\to\infty} 2\frac{4^t\left(W(G_0) + (e_0+n_0)(1-(\frac{3}{4})^t)\right)}{4^t(n_0)^2 - 2^t n_0} \\ &= \lim_{t\to\infty} 2\frac{\left(W(G_0) + (e_0+n_0)(1-(\frac{3}{4})^t)\right)}{(n_0)^2 - 4^{-t}2^t n_0} \\ &= \lim_{t\to\infty} 2\frac{\left(W(G_0) + (e_0+n_0)(1-(\frac{3}{4})^t)\right)}{(n_0)^2 - 2^{-t} n_0} \\ &= \frac{2(W(G_0)+e_0+n_0)}{(n_0)^2}. \qquad \square \end{aligned}$$

Theorem 2.9 tells us that for certain graphs, the ultimate average distance is in fact lower than its average distance. Hence, for many initial graphs G_0, the average distance decreases, a property observed in on-line social and other networks (see [**12, 14**]).

Lemma 2.10. *$UL(G_0) \leq L(G_0)$ if and only if*

$$W(G_0) \geq (n_0 - 1)(e_0 + n_0).$$

Proof. Now $UL(G_0) \leq L(G_0)$ holds if and only if

$$0 \geq \frac{2(W(G_0)+e_0+n_0)}{(n_0)^2} - \frac{2W(G_0)}{(n_0)^2 - n_0}.$$

This in turn is equivalent to

$$2W(G_0)n_0 \geq 2(e_0n_0^2 - e_0n_0 + n_0^3 - n_0^2)$$

$$W(G_0) \geq e_0n_0^2 - e_0n_0 + n_0^3 - n_0^2$$

which simplifies to give the desired equivalence. $\square$

We found the least n required to satisfy the condition $UL(G_0) \leq L(G_0)$ for a cycle. If $n \geq 16$, where n is even, then $UL(C_n) < L(C_n)$. This was found using the fact that

$$W(C_n) = \frac{n^3}{8}.$$

Diameters are constant in the ILT model. We record this as a strong indication of the small world property in the model.

THEOREM 2.11. *For all graphs G different than a clique,*

$$\mathrm{diam}(G_t) = \mathrm{diam}(G_0),$$

and

$$\mathrm{diam}(G_t) = \mathrm{diam}(G_0) + 1 = 2$$

when G_0 is a clique.

Proof. As the diameter of a graph is the maximum over all distances, the proof follows directly from Lemma 2.6. □

A formal discussion of the degree distribution of the ILT model is beyond the scope of this thesis. As an example of the degree distribution (in log-log scale) of a graph generated by the ILT model, see Figure 2.2 which was generated using MATLAB. If $G_0 \cong K_3$ and $t = 11$, then the resulting graph G_{11} seems to follow a binomial-type distribution.

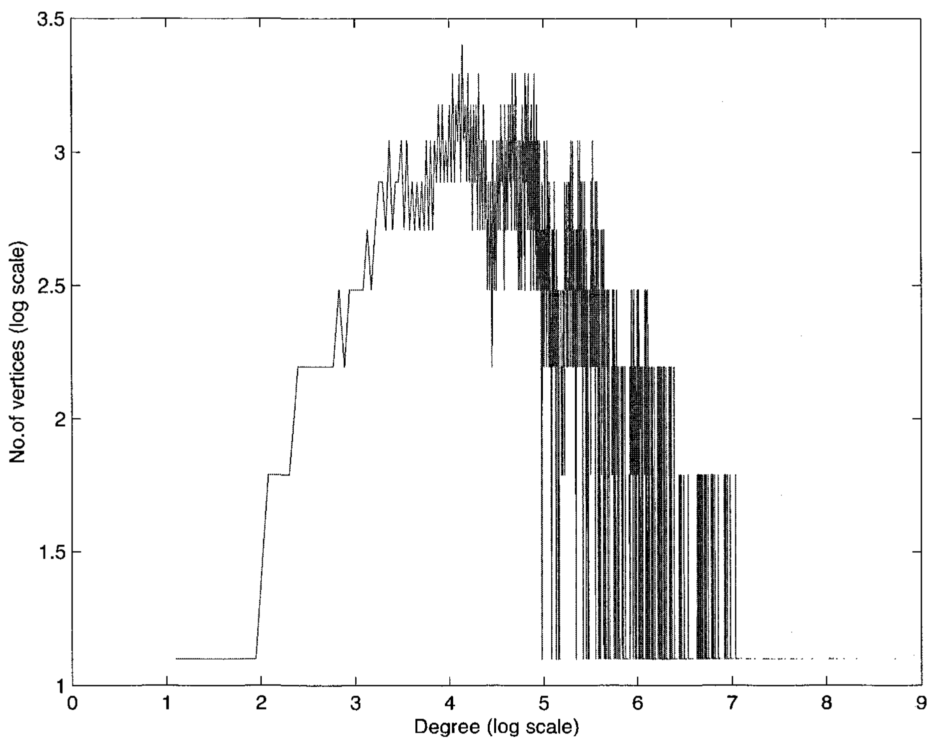


Figure 2.2: Degree distribution for G_{11} with $G_0 \cong K_3$.

CHAPTER 3

Other Properties of the ILT model

In this chapter, we supply theorems on the cop and domination numbers of the graph G_t. We provide theorems about the automorphisms of G_t. We finally prove a recurrence for eigenvalues of the adjacency matrix of G_t.

3.1. Cop and Domination number

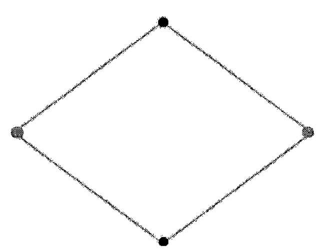
In the following theorems, we prove that the domination and cop numbers depend only on the initial graph G_0. Theorem 3.1 shows that even as the graph becomes large as t progresses, the same number of nodes needed at time 0 to dominate the graph will be needed at time t. In terms of on-line social networks, this suggests that a gossiper in the network can easily spread gossip no matter how large the graph becomes. Hence, one interpretation of Theorem 3.1 is that gossip can easily spread in an on-line social network. We now prove the theorem on the domination number of G_t.

THEOREM 3.1. *For all* $t \geq 0$,

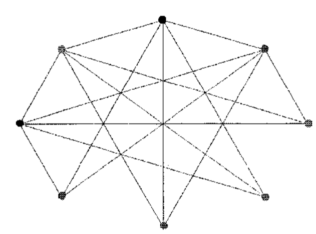
$$\gamma(G_t) = \gamma(G_0).$$

Proof. We prove that for $t \geq 0$, $\gamma(G_{t+1}) = \gamma(G_t)$. It then follows that $\gamma(G_t) = \gamma(G_0)$. When a dominating node $x \in V(G_t)$ is cloned, it's clone x' will be dominated by the same dominating node x since x' is joined to x and $N(x)$. The clone y' of a non-dominating node $y \in V(G_t)$ will be joined to a dominating node since y has a dominating node as its neighbour. Hence, a dominating set in G_t is a dominating set in G_{t+1}. □

As an example, Figure 3.1 shows a dominating set in $G_0 \cong C_4$ that is a dominating set in G_1. The black nodes constitute the dominating sets of G_0 and G_1.



(a) $\gamma(G_0) = 2$



(b) $\gamma(G_1) = 2$

Figure 3.1: The dominating sets in G_0 and G_1.

We prove that the cop number remains the same for G_t. This implies that no matter how large the graph G_t becomes, the robber can be captured by the same number of cops used at time 0. In terms of on-line social networks, the robber is synonymous to a gossiper who spreads gossips in the network. In order to "track" this gossiper at time t, we only require the initial number of cops to follow him. Therefore, one interpretation of Theorem 3.2 is that gossip can easily be tracked in an on-line social network. This contrasts with Theorem 3.1: although gossip can be tracked with few resources in an on-line social network on one hand, on the other gossip can be easily spread through the network.

THEOREM 3.2. *For all* $t \geq 0$,

$$c(G_t) = c(G_0).$$

Proof. We prove by induction that for $t \geq 0$, $c(G_t) = c(G_0)$. The base case is immediate. For the induction step, we show that $c(G_{t+1}) = c(G_t)$. Let $c = c(G_t)$. Assume that c cops play in G_{t+1} so that whenever R is on $x' \in$

$V(G_{t+1})\backslash V(G_t)$, the cops C play as if he were on $x \in V(G_t)$. Either C captures R on x', or using their winning strategy in G_t, the cops move to x with R on x'. The cops then win in the next round. Hence,

$$c(G_{t+1}) \leq c(G_t).$$

Suppose for a contradiction that $b = c(G_{t+1}) < c$. The cops then use their winning strategy in G_{t+1} to win with b cops in G_t; this contradiction will show that $b \geq c$. To see this, C plays in G_t so that whenever R is on $x \in V(G_t)$, C plays as if R were on $x' \in V(G_{t+1})$. As x and x' are joined and share the exact same neighbours in G_{t+1}, C may win in G_t with $b < c$ cops. $\square$

3.2. Automorphisms

In this section, we provide theorems about the automorphism groups of graphs generated by the ILT model. We say that an automorphism $f_t \in \mathrm{Aut}(G_t)$ *extends* to $f_{t+1} \in \mathrm{Aut}(G_{t+1})$ if

$$f_{t+1} \restriction V(G_t) = f_t.$$

We show that symmetries from $t = 0$ are preserved at time t since there is an embedding of automorphism groups as we see in Theorem 3.4. This provides further evidence that the ILT model retains a memory of the initial graph from time 0.

THEOREM 3.3. *Each $f_0 \in \mathrm{Aut}(G_0)$, extends to $f_t \in \mathrm{Aut}(G_t)$.*

Proof. Given $f_0 \in \mathrm{Aut}(G_0)$, we prove by induction on $t \geq 0$ that f_0 extends to $f_t \in \mathrm{Aut}(G_t)$. The base case is immediate. Assuming that f_t is defined, let

$$f_{t+1}(x) = \begin{cases} f_t(x) & \text{if } x \in V(G_t), \\ (f_t(y))' & \text{where } x = y' \ . \end{cases}$$

To prove that $f_{t+1}(x)$ is injective, we consider three cases. Let x, y be distinct nodes of $V(G_t)$. As f_t is one-to-one, $f_{t+1}(x) \neq f_{t+1}(y)$. For the case when $x \in V(G_t)$, we have $f_t(x) \neq f_t(y)$. Then $f_{t+1}(x) \neq f_{t+1}(y)$. In the case when $x, y \in V(G_{t+1}) \setminus V(G_t)$, we have that $f_{t+1}(x) = (f_t(z))'$, where $x = z'$, and $f_{t+1}(y) = (f_t(u))'$, where $y = u'$. Since $x \neq y$ and $z' \neq u'$ we have that $z \neq u$. It follows that

$f_t(z) \neq f_t(u)$. Hence, it follows that $(f_t(z))' \neq (f_t(u))'$, which in turn implies that $f_{t+1}(x) \neq f_{t+1}(y)$.

For the last case when $x \in V(G_t)$ and $y \in V(G_{t+1}) \setminus V(G_t)$, we have that $f_{t+1}(x) = f_t(x)$ and $f_{t+1}(y) = (f_t(z))'$, where $y = z'$. We know that $f_t(x) \in V(G_t)$ and $(f_t(z))' \in V(G_{t+1}) \setminus V(G_t)$. Hence, $f_t(x) \neq (f_t(z))'$, and so $f_{t+1}(x) \neq f_{t+1}(y)$. Thus, f_{t+1} is injective.

To show that the map $f_{t+1}(x)$ is onto, consider the cases for $x \in V(G_t)$, and $x \notin V(G_t)$. For the first case $x \in V(G_t)$, there exists a $y \in V(G_t)$ such that $f_t(y) = x$ as f_t is onto. Therefore, $f_{t+1}(y) = x$. For the second case where $x \notin V(G_t)$, let $x = y'$ for $y \in V(G_t)$. Let $f_t(z) = y$. Then $f_{t+1}(z') = y'$ for some $z \in V(G_t)$.

We show that $xy \in E(G_{t+1})$ if and only if $f_{t+1}(x)f_{t+1}(y) \in E(G_{t+1})$. This will prove that $f_{t+1} \in \mathrm{Aut}(G_t)$, as f_{t+1} extends f_t.

The case for $x, y \in V(G_t)$ is immediate as $f_t \in \mathrm{Aut}(G_t)$. Next, we consider the case for $x \in V(G_t)$ and $y' \in V(G_{t+1})$. Now $xy' \in E(G_{t+1})$ if and only if

$$f_{t+1}(x)f_{t+1}(y') = f_t(x)(f_t(y))' \in E(G_{t+1}).$$

Note that $x'y' \notin E(G_{t+1})$ for all $x', y' \in V(G_{t+1}) \setminus V(G_t)$. But $f_{t+1}(x')f_{t+1}(y') \notin E(G_{t+1})$ by definition of G_{t+1}. $\square$

A *homomorphism* of a group $(G, \cdot)$ into a group $(H, *)$ is a function T of G into H, such that if $x \in G$ and $y \in G$, then $T(x \cdot y) = T(x) * T(y)$. An *embedding* is a one-to-one homomorphism. We abuse notation and say that G *embeds* in H. We now present a theorem for the embedding of automorphism groups of graphs generated by the ILT model.

THEOREM 3.4. *For all* $t \geq 0$, $\mathrm{Aut}(G_0)$ *embeds in* $\mathrm{Aut}(G_t)$.

Proof. We show that for all $t \geq 0$, $\mathrm{Aut}(G_t)$ embeds in $\mathrm{Aut}(G_{t+1})$. The proof of the theorem then follows from this fact by induction on t. Define

$$\phi : \mathrm{Aut}(G_t) \longrightarrow \mathrm{Aut}(G_{t+1})$$

by

$$\phi(f)(x) = \begin{cases} f(x) & \text{if } x \in V(G_t), \\ (f(y))' & \text{if } x = y' \in V(G_{t+1}) \setminus V(G_t). \end{cases}$$

Note that $\phi(f)(x)$ is injective, since $f \neq g$ implies that $\phi(f) \neq \phi(g)$ by the definition of ϕ.

We now prove by cases that for all $x \in V(G_{t+1})$ and $f, g \in \mathrm{Aut}(G_t)$,

$$\phi(fg)(x) = \phi(f)\phi(g)(x).$$

Case 1: The node $x \in V(G_t)$.

In this case,

$$\phi(fg)(x) = fg(x) = \phi(f)\phi(g)(x).$$

Case 2: The node $x \notin V(G_t)$.

In this case, say $x = y'$, with $y \in V(G_t)$. Then we have that

$$\begin{aligned}
\phi(fg)(x) &= (fg(y))' \\
&= \phi(f)((g)(y))' \\
&= \phi(f)\phi g(y') \\
&= \phi(f)\phi(g)(x). \qquad \square
\end{aligned}$$

3.3. Eigenvalues of ILT Model

In this section, we consider the adjacency matrix for G_t. We present a recurrence for the eigenvalues of the graph G_t.

If $A(G_t) = A$ is the adjacency matrix of G_t, then the adjacency matrix of G_{t+1} is

$$M = \begin{pmatrix} A & A+I \\ A+I & 0 \end{pmatrix},$$

where is I is the identity matrix of order n_t. In this matrix, A corresponds to nodes at time t, $A+I$ corresponds to nodes at time $t+1$, and the zero matrix represents that there are no edges between $V(G_{t+1}) \setminus V(G_t)$. The identity matrix I appears since for every $x' \in V(G_{t+1})$ there exists a node $x \in V(G_t)$ and an edge between x and x'. For example, if $G_0 \cong C_4$, then $A(G_0)$ is

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}.$$

The adjacency matrix of G_1 in this case will be:

$$\left(\begin{array}{cccc|cccc} 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ \hline 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right).$$

We now present a theorem for a recurrence of the eigenvalues of G_t.

THEOREM 3.5. *If λ is an eigenvalue of $A(G_t)$, then*

$$\rho_{\pm} = \frac{\lambda \pm \sqrt{\lambda^2 + 4(\lambda+1)^2}}{2}$$

are eigenvalues of $A(G_{t+1})$.

Proof. We first assume that $\lambda \neq -1$. Hence, $\rho = \rho_{\pm} \neq 0$. Let $\mathbf{u}$ be an eigenvector of $A = A(G_t)$ such that

$$A\mathbf{u} = \lambda\mathbf{u}.$$

Let $\beta = \frac{(\lambda+1)}{\rho}$, and let

$$\mathbf{v} = \begin{pmatrix} \mathbf{u} \\ \beta\mathbf{u} \end{pmatrix}.$$

Then we have that,

$$\begin{aligned} M\mathbf{v} &= \begin{pmatrix} A & A+I \\ A+I & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \beta\mathbf{u} \end{pmatrix} \\ &= \begin{pmatrix} A\mathbf{u} + (A+I)\beta\mathbf{u} \\ (A+I)\mathbf{u} \end{pmatrix} \\ &= \begin{pmatrix} \lambda\mathbf{u} + (\lambda+1)\beta\mathbf{u} \\ (\lambda+1)\mathbf{u} \end{pmatrix}. \end{aligned}$$

Now $\beta\rho = \lambda + 1$, and so $(\lambda+1)\mathbf{u} = \beta\rho\mathbf{u}$. The condition

$$\rho = \lambda + \beta(\lambda+1) = \lambda + \frac{(\lambda+1)^2}{\rho}$$

is equivalent to ρ solving

$$x - \lambda - \frac{(\lambda+1)^2}{x} = 0,$$

which it does by its definition. Hence,

$$M\mathbf{v} = \rho\mathbf{v}$$

as desired.

Now let $\lambda = -1$. In this case, $\rho_- = -1$. Let

$$\mathbf{v} = \begin{pmatrix} \mathbf{u} \\ \mathbf{0} \end{pmatrix},$$

where $\mathbf{0}$ is the appropriately sized zero vector. Thus,

$$\begin{aligned} M\mathbf{v} &= \begin{pmatrix} A & A+I \\ A+I & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{0} \end{pmatrix} \\ &= \begin{pmatrix} A\mathbf{u} \\ (A+I)\mathbf{u} \end{pmatrix} \\ &= \begin{pmatrix} \lambda\mathbf{u} \\ (\lambda+1)\mathbf{u} \end{pmatrix} \\ &= \begin{pmatrix} -\mathbf{u} \\ \mathbf{0} \end{pmatrix}. \end{aligned}$$

Hence,

$$M\mathbf{v} = \rho_-\mathbf{v}$$

as desired. In this case when $\rho_+ = 0$ and $\lambda = -1$; let

$$\mathbf{v} = \begin{pmatrix} \mathbf{0} \\ \mathbf{u} \end{pmatrix}.$$

Hence,

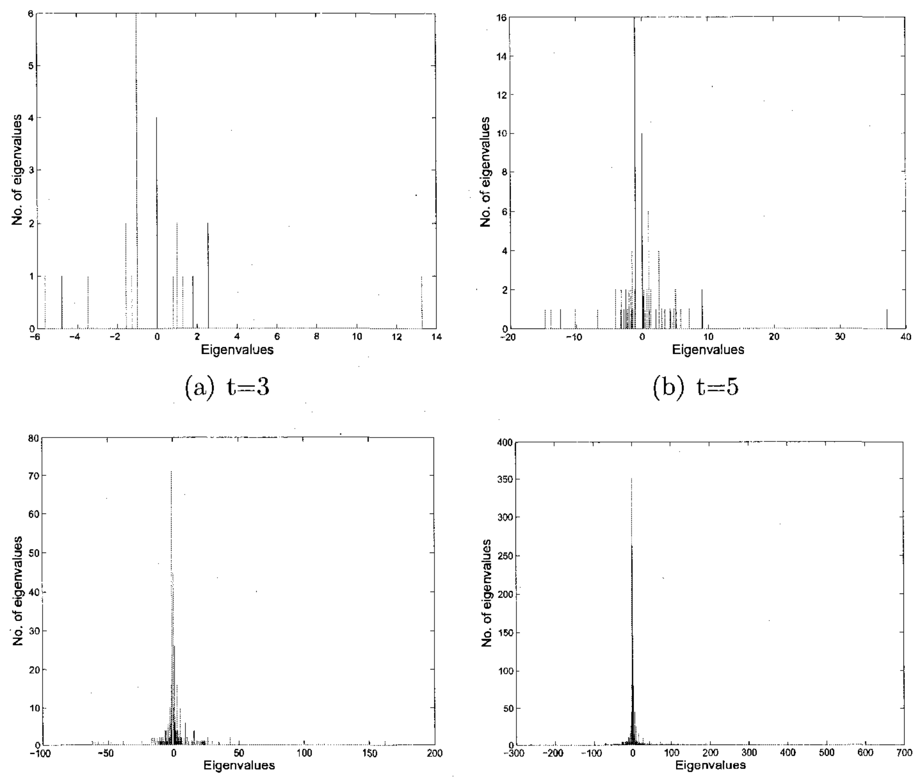
$$\begin{aligned} M\mathbf{v} &= \begin{pmatrix} A & A+I \\ A+I & 0 \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ \mathbf{u} \end{pmatrix} \\ &= \begin{pmatrix} (A+I)\mathbf{u} \\ \mathbf{0} \end{pmatrix} \\ &= \begin{pmatrix} (\lambda+1)\mathbf{u} \\ \mathbf{0} \end{pmatrix}. \end{aligned}$$

We therefore have that $M\mathbf{v} = \rho_+\mathbf{v} \qquad \square$.

The recurrence of eigenvalues can be explicitly seen by taking, for example, $G_0 \cong K_3$. The eigenvalues at various time-steps are given in Table 1. We computed these eigenvalues directly using MATLAB. Figure 3.2 shows the eigenvalue distribution for K_3 at various time-steps, and it seems to follow a binomial-type distribution.

Table 1: Eigenvalues of G_t for $t = 0, 1, 2, 3, 4$ with $G_0 \cong K_3$

t	**Eigenvalues**
0	$-1, -1, 2$
1	$-2.16, -1, -1, 0, 0, 4.16$
2	$-3.48, -2.66, -1, -1, -1, -1, 0, 0, 0.50, 1, 1, 7.64.$
3	$-5.63, -4.77, -3.47, -1.56, -1.56, -1.27,$ $-1, -1, -1, -1, -1, -1, 0, 0, 0, 0, 0.80,$ $1, 1.29, 1.78, 2.56, 2.56, 13.2$
4	$-9.10, -8.23, -6.85, -4.75, -2.50, -2.50, -2.02, -1.74$ $-1.74, -1.73, -1.56, -1.56, -1.44, -1.33, -1, -1, -1, -1, -1, -1$ $-1, -1, -1, -1, 0, 0, 0, 0, 0, 0, 0.05, 0.18, 0.18, 1, 1, 1, 1,$ $1.28, 2.08, 2.24, 2.56, 2.56, 2.60, 3.02, 3.80, 5.06, 5.06, 22.38.$



(a) t=3 (b) t=5

(c) t=8 (d) t=11

Figure 3.2: The eigenvalue distribution for G_t for various time-steps, with $G_0 \cong K_3$.

CHAPTER 4

The Random ILT Model

Random graph models have been widely used to simulate and predict the behaviour of complex real-world networks (see [**6, 12**]). A model that incorporates randomness is more realistic, and is often *tuneable*: choosing the parameters affects the observed properties. Various studies (see [**4, 13**]) have shown that networks, like blogspace (that is, the network whose nodes consist of blogs, and edges are links between blogs) and the web graph follow a power law degree distribution. In this chapter, we introduce a random ILT model whose graphs follow a power law degree distribution. We present simulations for the degree distribution of graphs generated by the random ILT model.

4.1. Power law Degree Distributions

One of the most important properties observed in complex networks is a power law degree distribution. Given an undirected graph G and a non-negative integer k, we define

$N_{k,G}$ by

$$N_{k,G} = |\{x \in V(G) : deg_G(x) = k\}|.$$

The parameter $N_{k,G}$ is the *number of nodes of degree k in G*. For simplicity, suppose that $|V(G)| = t$. Then $|N_{k,G}|$ is an integer in the interval $[0, t]$.

The *degree distribution* of G is the sequence

$$(N_{k,G} : 0 \leq k \leq t).$$

We say that the degree distribution of G follows a *power law* if for each degree k,

$$\frac{N_{k,G}}{t} \sim k^{-\beta}, \tag{4.1}$$

for a fixed real constant $\beta > 1$. Note that (4.1) is asymptotic and can be interpreted for a fixed graph as meaning that $\frac{N_{k,G}}{t}$ is approximately $k^{-\beta}$. We are more interested in the approximate rather than exact value of $\frac{N_{k,G}}{t}$ since G is a large graph.

Power law distributions are sometimes referred to as *heavy-tailed distributions*, since the real-valued function

$$f(k) = k^{-\beta}$$

exhibits a polynomial (rather than exponential) decay to 0 as k tends to ∞. We say that β is the *exponent of the power law*. If G possesses a power law degree distribution, then we simply say G is a *power law graph*. If we take logarithms on both sides of (4.1), then the relationship is expressed as

$$\log(N_{k,G}) \sim \log(t) - \beta\log(k).$$

Hence, in the log-log plot, we obtain a straight line with slope $-\beta$. In both real-world networks and graphs generated by theoretical models, the power law may only fit for a certain range of degrees, with discrepancies for small or large degree nodes.

4.2. Preferential Attachment and Duplication Models

Preferential attachment models are used to simulate the web graph and other complex networks. Barabási and Albert [**2**] designed the first model for the web graph. The main idea in their model is that new nodes are more likely to join to existing nodes with high degree. This model is now referred to as an example of a *preferential attachment (or PA) model.* Barabási and Albert concluded that the model generates graphs whose in-degree distribution follows a power law with exponent $\beta = 3$. The first rigorous analysis of a PA model was given in Bollobás, Riordan, Spencer, and Tusnady [**5**].

The duplication model is similar to the deterministic ILT model; however, cloning (or duplication) occurs on only one uniformly randomly chosen node. The node chosen to be cloned (or duplicated) will have a new node linked to it and all of its neighbours. The duplication model was designed to describe the behaviour of biological networks such as protein-protein interaction networks in a living cell. It was observed that graphs generated by duplication models follow power law degree distributions with power law exponents in the interval $(1, 2)$ (see [8]).

4.3. The Random ILT Model

In this section, we introduce a randomized version of the ILT model. The motivation for the model is that at each time-step, the new member of the on-line social network becomes friends with a popular person (modelled by preferential attachment), or clones the neighbour set of some existing node. As we will see, graphs generated by the random ILT model follow a power law degree distribution.

Let $\alpha \in (0, 1]$ be a fixed real number. At time $t = 0$, let G_0 be an initial graph with minimum degree 1. At time $t + 1$, with probability α, a *PA step* is taken; that is, an existing node is chosen giving preference to nodes with higher degrees and a new node is linked to it. Hence, the probability the new node is joined to $x \in V(G_t)$ is

$$\frac{\mathrm{deg(x)}}{\sum_{x \in V(G_t)} \mathrm{deg(x)}} = \frac{\mathrm{deg(x)}}{\mathrm{vol}(G_t)}.$$

Figure 4.1 shows an example of a graph before and after a PA step is taken where the white node is the new node that is added at $t+1$ and joined to the node with the highest degree. Note that one new edge is added in a PA step. With probability $1-\alpha$, a *duplication step* is taken; that is, a node is chosen uniformly at random and an edge is added to it and to all of its neighbours. Thus, at every time-step only one node is added to the graph. If we allowed $\alpha = 0$, then we would have the duplication model. In the case that $\alpha = 1$, we have the preferential attachment model. Hence, if $\alpha \in (0,1)$, we may view the random ILT model as a mixture of both models, so that duplication occurs more often if α is closer to 0.

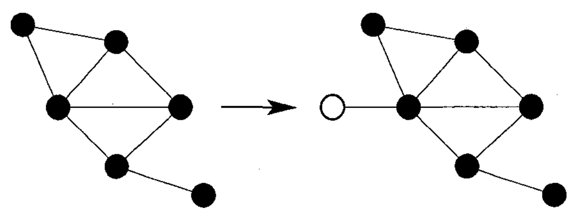


Figure 4.1: A graph before and after a PA step.

For simplicity, we write N_{k,G_t} as $N_{k,t}$. Observe that the parameters $N_{k,t}$ are random variables. We derive the so-called *master equation* for $\mathbb{E}(N_{k,t})$ and show that the power law exponent depends on the probability α. In order to find $\mathbb{E}(N_{k,t})$ we first derive a recurrence for the expected value of the number of edges e_t as shown in Lemma 4.2 and then find a solution for $e(t)$, the expectation of e_t, in Lemma 4.3. We note that $|V(G_t)| = t + n_0 \sim t$.

THEOREM 4.1. *Assuming that*

$$\mathbb{E}(N_{k,t}) = b_k t \text{ and } b_k \sim ck^{-\gamma},$$

where $c > 0$, *then*

$$\gamma = \begin{cases} \frac{2-\alpha}{1-\alpha}, & if \alpha \leq \frac{1}{2}; \\ 1 + \frac{1}{\alpha^2 - \frac{3}{2}\alpha + 1}, & if \alpha > \frac{1}{2}. \end{cases}$$

We first prove the following lemmas.

LEMMA 4.2. *For all* $t \geq 0$,

$$e(t+1) = e(t)\left(1 + \frac{2(1-\alpha)}{t + n_0}\right) + 1.$$

Proof. By the linearity of expectation, we have that

$$\begin{aligned}
e(t+1) &= \alpha(1) + (1-\alpha) \sum_{x \in V(G_t)} (1 + deg_t(x)) \frac{1}{n_t} + e(t) \\
&= \alpha + \frac{\sum_{x \in V(G_t)} (1 + deg_t(x))}{n_t} \\
&- \alpha \frac{\sum_{x \in V(G_t)} (1 + deg_t(x))}{n_t} + e(t) \\
&= \alpha + \frac{n_t}{n_t} + \frac{2e(t)}{n_t} - \frac{\alpha n_t}{n_t} - \frac{2\alpha e(t)}{n_t} + e(t) \\
&= 1 + \frac{2e(t)}{n_t}(1-\alpha) + e(t) \\
&= e(t)\left(\frac{2(1-\alpha)}{n_t} + 1\right) + 1 \\
&= e(t)\left(\frac{2(1-\alpha)}{t+n_0} + 1\right) + 1. \qquad \square
\end{aligned}$$

LEMMA 4.3. *A.a.s*

$$e(t) = \begin{cases} \Theta(t^{2(1-\alpha)}), & \alpha < 1/2; \\ \Theta(t \ln t), & \alpha = 1/2; \\ \Theta(t) & \alpha > 1/2. \end{cases}$$

Proof. A rigorous proof of the lemma is beyond the scope of this thesis. However, we use a method given in [**3**] (see the proof of Lemma 7) for approximating the recurrence by a differential equation.

We may show by direct substitution that the values given in the statement of the lemma satisfy the recurrence relation in Lemma 4.2. We only consider the case when $e(t) = \Theta(t)$ for $\alpha > \frac{1}{2}$ (the other cases are similar, and so are omitted). In this case,

$$\begin{aligned} e(t)\left(1+\frac{2(1-\alpha)}{t+n_0}\right)+1 &= \Theta(t)\left(1+\frac{2(1-\alpha)}{t+n_0}\right)+1 \\ &= \Theta(t)(1+o(1))+1 \\ &= \Theta(t+1) = e(t+1). \end{aligned}$$

The recursion in Lemma (4.2) suggests the following differential equation:

$$\frac{d(e(t))}{dt} = e(t)\frac{2(1-\alpha)}{n_0+t}+1,$$

with an initial condition $e(0) = e_0$. From Calculus (see, for example, [**17**]) we know that the solution to a differential equation of this form is,

$$\begin{aligned} e(t) &= e^{\int \frac{2(1-\alpha)}{t+n_0}dt}\left(C+\int e^{-\int \frac{2(1-\alpha)}{t+n_0}dt}dt\right) \\ &= (t+n_0)^{2(1-\alpha)}\left(C+\int (t+n_0)^{-2(1-\alpha)}dt\right), \quad (4.2) \end{aligned}$$

where C is a constant. Since $e_0 \geq 1$, it follows that $C \neq 0$.

We now consider three possible cases for α.

Case 1: The parameter α satisfies $\alpha < \frac{1}{2}$.

From (4.2) we notice that $2(1-\alpha) > 1$. Hence, we have that

$$\begin{aligned} e(t) &= (t+n_0)^{2(1-\alpha)}\left(C + \int (t+n_0)^{-2(1-\alpha)}dt\right) \\ &= C(t+n_0)^{2(1-\alpha)} + \frac{1}{2\alpha-1}(t+n_0) \\ &= \Theta\left((t+n_0)^{2(1-\alpha)}\right). \end{aligned}$$

Case 2: The parameter α satisfies $\alpha = \frac{1}{2}$.

From (4.2) we have that,

$$\begin{aligned} e(t) &= (t+n_0)\left(C + \int (t+n_0)^{-1}dt\right) \\ &= \Theta\left(t \ln t\right). \end{aligned}$$

Case 3: The parameter α satisfies $\alpha > \frac{1}{2}$.

Then by (4.2), since $2(1-\alpha) < 1$, we have that

$$\begin{aligned} e(t) &= (t+n_0)^{2(1-\alpha)}\left(C + \int (t+n_0)^{-2(1-\alpha)}dt\right) \\ &= C(t+n_0)^{2(1-\alpha)} + \frac{1}{2\alpha-1}(t+n_0) \\ &= \Theta(t). \quad \square \end{aligned}$$

We can now prove Theorem 4.1.

Proof of Theorem 4.1 We first solve for the master equation when $\alpha \leq \frac{1}{2}$. For each $u \in V(G_{t+1})$, let X_u be the indicator random variable defined as,

$$X(u) = \begin{cases} 1 & \text{if } \deg_{t+1}(u) = k, \\ 0 & \text{if } \deg_{t+1}(u) \neq k. \end{cases}$$

Then

$$N_{k,t+1} = \sum_{u \in V(G_{t+1})} X_u,$$

and so by the linearity of expectation we have that

$$\mathbb{E}(N_{k,t+1}) = \sum_{u \in V(G_{t+1})} \mathbb{P}(X_u = 1).$$

We find $\mathbb{P}(X_u = 1)$ by considering two cases for $\deg_{G_t}(u)$.

Case 1: The degree of u satisfies $\deg_{G_t}(u) = k - 1$.

Such a node u may have degree k at time $t+1$ if it was chosen as a random node for a PA step, or the node u or any of its neighbours were chosen as a node for a duplication step. Thus, we have that

$$\mathbb{P}(X_u = 1) = \alpha\frac{k-1}{2e(t)} + (1-\alpha)\frac{k}{t+n_0}.$$

Case 2: The degree of u satisfies $\deg_{G_t}(u) = k$.

Such a node u may have degree k at time $t+1$ if it neither it was chosen as a random node for a PA step, nor the node u or its neighbours were chosen as node for a duplication step. Thus, we have that

$$\mathbb{P}(X_u = 1) = 1 - \alpha\frac{k}{2e(t)} - (1-\alpha)\frac{k+1}{t+n_0}.$$

We now have

$$\begin{aligned} \mathbb{E}(N_{k,t+1}|G_t) &= N_{k-1,t}\left(\alpha\frac{k-1}{2e(t)} + (1-\alpha)\frac{k}{t+n_0}\right) \\ &+ N_{k,t}\left(1 - \alpha\frac{k}{2e(t)} - (1-\alpha)\frac{k+1}{t+n_0}\right). \quad (4.3) \end{aligned}$$

Taking expectation on both sides of (4.3), we have that

$$
\begin{aligned}
\mathbb{E}(N_{k,t+1}) = \quad & \mathbb{E}\left(N_{k-1,t}\left(\alpha\frac{k-1}{2e(t)} + (1-\alpha)\frac{k}{t+n_0}\right)\right) \\
+ \quad & \mathbb{E}\left(N_{k,t}\left(1-\alpha\frac{k}{2e(t)} - (1-\alpha)\frac{k+1}{t+n_0}\right)\right) \\
= \quad & \mathbb{E}(N_{k-1,t})\left(\alpha\frac{k-1}{2e(t)} + (1-\alpha)\frac{k}{t+n_0}\right) \\
+ \quad & \mathbb{E}(N_{k,t})\left(1-\frac{\alpha k}{2e(t)} - (1-\alpha)\frac{k+1}{t+n_0}\right).
\end{aligned} \tag{4.4}
$$

Using the assumption that $\mathbb{E}(N_{k,t}) = b_k t$, (4.4) becomes

$$
\begin{aligned}
b_k(t+1) \;=\; & \left(\frac{\alpha(k-1)}{2e(t)} + \frac{(1-\alpha)k}{t+n_0}\right) t b_{k-1} \\
+ \; & \left(1-\frac{\alpha k}{2e(t)} - (1-\alpha)\frac{k+1}{t+n_0}\right) t b_k,
\end{aligned}
$$

so that

$$
b_k + \frac{(1-\alpha)(k+1)t}{t+n_0} b_k = \frac{t(1-\alpha)k}{t+n_0} b_{k-1},
$$

which follows from the fact that $\frac{t}{e(t)} = o(1)$ by Lemma 4.3.

Hence,

$$
b_k\left(1+\frac{(1-\alpha)(k+1)t}{t+n_0}\right) = \frac{t(1-\alpha)k}{t+n_0} b_{k-1},
$$

and so

$$b_k = \frac{t(1-\alpha)k}{t+n_0+(1-\alpha)(k+1)t} b_{k-1}.$$

Therefore,

$$\begin{aligned} b_k &= \frac{k(1-\alpha)}{1+\frac{n_0}{t}+(k+1)(1-\alpha)} b_{k-1} \\ &= \frac{k(1-\alpha)}{1+(k+1)(1-\alpha)} b_{k-1}. \end{aligned}$$

Using the assumption that $b_k \sim ck^{-\gamma}$, we have that

$$\frac{b_{k-1}}{b_k} \sim \left(\frac{k}{k-1}\right)^{\gamma} = \left(1+\frac{1}{k-1}\right)^{\gamma} = 1+\gamma\frac{1}{k}+O\left(\frac{1}{k^2}\right).$$

To find γ we use the fact that

$$\left(\frac{k}{k-1}\right)^{\gamma} = \frac{1+(k+1)(1-\alpha)}{k(1-\alpha)}. \tag{4.5}$$

Using long division on (4.5), we have that

$$\left(\frac{k}{k-1}\right)^{\gamma} = 1+\frac{2-\alpha}{(1-\alpha)}\frac{1}{k}+O\left(\frac{1}{k^2}\right).$$

Hence, $\gamma = \frac{2-\alpha}{1-\alpha}$. Thus, we obtain that

$$b_k \sim k^{-\frac{2-\alpha}{1-\alpha}}.$$

We now present the master equation for $\alpha > \frac{1}{2}$.

Similar to the case when $\alpha \leq \frac{1}{2}$, we have

$$\mathbb{E}(N_{k,t+1}|G_t) = N_{k-1,t}\left(\alpha\frac{k-1}{2e(t)} + (1-\alpha)\frac{k}{t+n_0}\right)$$
$$+ N_{k,t}\left(1 - \alpha\frac{k}{2e(t)} - (1-\alpha)\frac{k+1}{t+n_0}\right). \quad (4.6)$$

Taking expectation on both sides of (4.6) and assuming that $\mathbb{E}(N_{k,t}) = b_k t$, we have that

$$b_k(t+1) = b_{k-1}t\left(\frac{\alpha(k-1)}{2e(t)} + \frac{(1-\alpha)k}{t+n_0}\right)$$
$$+ b_k t\left(1 - \frac{\alpha k}{2e(t)} - (1-\alpha)\frac{k+1}{t+n_0}\right).$$

Hence,

$$b_k + \frac{k\alpha(2\alpha-1)}{2}b_k + \frac{(1-\alpha)(k+1)t}{t+n_0}b_k$$
$$= \frac{\alpha(k-1)(2\alpha-1)}{2}b_{k-1} + \frac{t(1-\alpha)k}{t+n_0}b_{k-1},$$

which follows from the facts that $e(t) \sim \frac{t}{2\alpha-1}$ and $\frac{t}{e(t)} \sim 2\alpha - 1$ by Lemma 4.3. Hence,

$$b_k\left(1 + \frac{\alpha k(2\alpha-1)}{2} + \frac{t(1-\alpha)(k+1)}{t+n_0}\right)$$
$$= b_{k-1}\left(\frac{\alpha(k-1)(2\alpha-1)}{2} + \frac{t(1-\alpha)k}{t+n_0}\right).$$

Thus,

$$b_k = \frac{k(1-\alpha) + (k-1)\alpha(2\alpha-1)/2}{1+(k+1)(1-\alpha)+k\alpha(2\alpha-1)/2} b_{k-1}.$$

Using the assumption that $b_k \sim ck^{-\gamma}$, we have that

$$\frac{b_{k-1}}{b_k} \sim \left(\frac{k}{k-1}\right)^{\gamma} = \left(1+\frac{1}{k-1}\right)^{\gamma} = 1+\gamma\frac{1}{k}+O\left(\frac{1}{k^2}\right).$$

To find γ, we use the fact that

$$\left(\frac{k}{k-1}\right)^{\gamma} = \frac{1+(k+1)(1-\alpha)+k\alpha(2\alpha-1)/2}{k(1-\alpha)+(k-1)\alpha(2\alpha-1)/2}. \qquad (4.7)$$

Using long division on (4.7), we have that

$$\left(\frac{k}{k-1}\right)^{\gamma} = 1+\left(1+\frac{1}{\alpha^2-\frac{3}{2}\alpha+1}\right)\frac{1}{k}+O\left(\frac{1}{k^2}\right).$$

Hence, $\gamma = 1+\frac{1}{\alpha^2-\frac{3}{2}\alpha+1}$. □

Theorem 4.1 only claims a power law for the expected value of $N_{k,t}$, with no reference to the concentration of this random variable around its expected value. Proving the concentration for $N_{k,t}$ around $\mathbb{E}(N_{k,t})$ is a difficult open problem for the duplication model (see [**8**]), and it is open for the random ILT model (which includes the duplication model when $\alpha = 1$).

4.4. Simulation results

We simulated the Random ILT model using C++. See the Appendix for the code. We plotted the cumulative degree distribution for different values of α; namely, $\alpha = 0.25, 0.5, 0.75$, and 1 (see Figures 4.2, 4.3, 4.4, and 4.5). The plots seem to follow a power-law degree distribution for degrees up to some threshold. We note that since these plots are for the cumulative degree distribution, the slope of the line is $1-\gamma$, where γ is the power law exponent. The values found for the power-law exponent from the plots coincide with the results stated in Theorem 4.1. See Table 1 for a comparison of the power law exponents found from the plots (called γ_{plot}) and the power law exponents from Theorem 4.1 (called γ_{thm}).

We notice that there are some differences between γ_{plot} and γ_{thm} values. This may be due to the time t being too small as we ran the simulations up to $t = 10,000$ only. A larger t would likely give us a closer estimate for γ_{plot}. The other reason may be due to $N_{k,t}$ not being sufficiently concentrated.

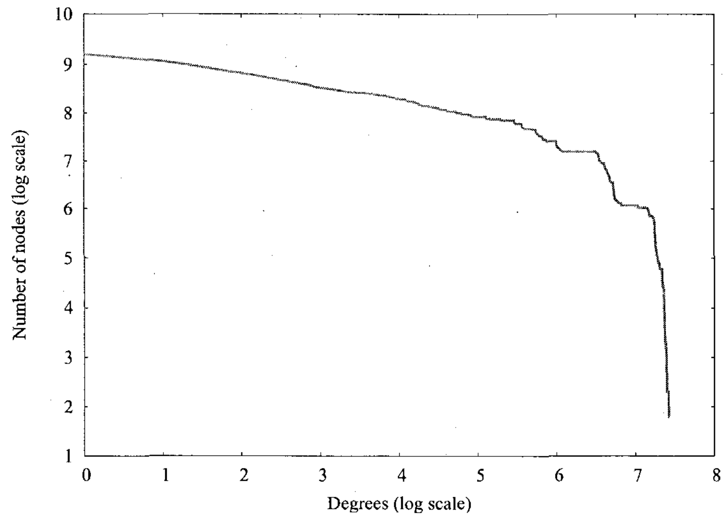


Figure 4.2: Cumulative degree distribution for G_{10000}, with $G_0 \cong K_3, \alpha = 0.25$.

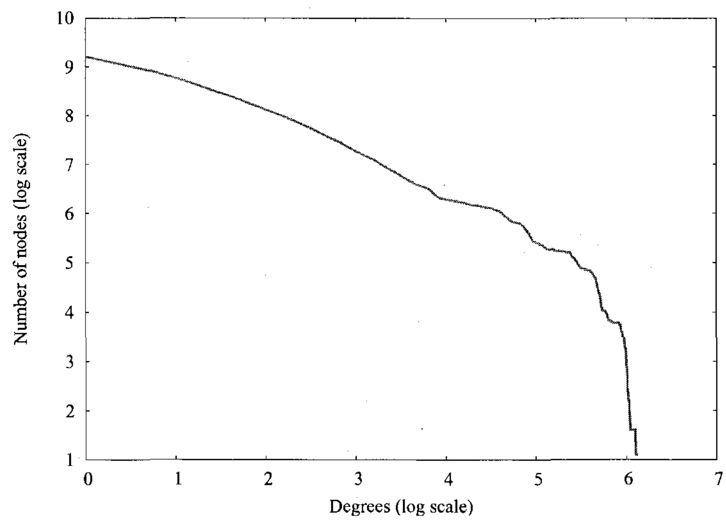


Figure 4.3: Cumulative degree distribution for G_{10000}, with $G_0 \cong K_3, \alpha = 0.50$.

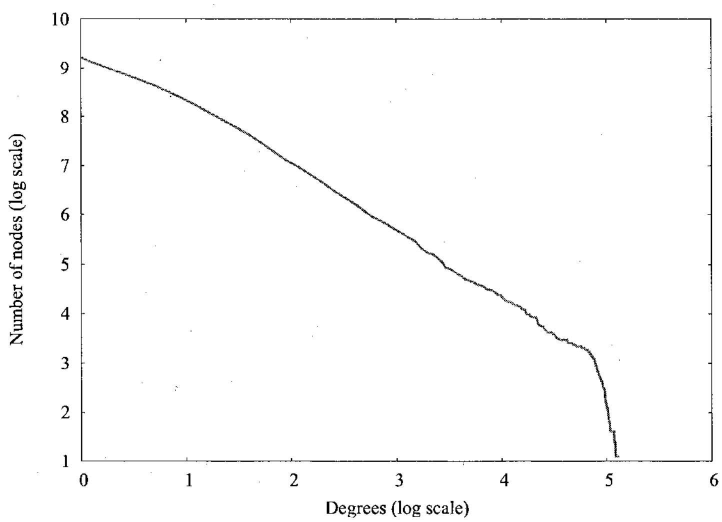


Figure 4.4: Cumulative degree distribution for G_{10000}, with $G_0 \cong K_3, \alpha = 0.75$.

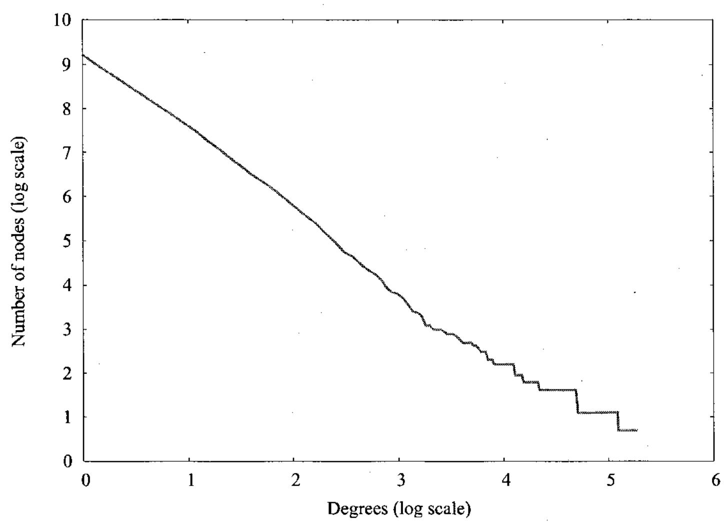


Figure 4.5: Cumulative degree distribution for G_{10000}, with $G_0 \cong K_3, \alpha = 1$.

Table 1: Comparison of power law exponents

α	γ_{plot}	γ_{thm}
0.25	1.18	2.33
0.5	3	3
0.75	2	2.28
1	2.66	3

CHAPTER 5

Open Problems

Several open problems remain pertaining to both the deterministic and random ILT models. Several of these problems—which were stated throughout the thesis—are listed below. We hope to address these problems in future work.

(1) Do the eigenvalues in the deterministic ILT model follow a binomial distribution? In Section 3.3, we presented a simulation for the distribution of eigenvalues, and this seemed to follow a binomial distribution.

(2) Does the degree distribution of graphs generated by the deterministic ILT model follow a binomial distribution? We noticed a binomial-like distribution from simulations presented in Section 2.2.

(3) Can we prove the concentration of $N_{k,t}$ around $\mathbb{E}(N_{k,t})$ in the random ILT model?

(4) In Theorem 4.1, we made the assumptions that $\mathbb{E}(N_{k,t}) = b_k t$ and $b_k \approx ck^{-\gamma}$. Can we prove both assumptions directly from the properties of the random ILT model?

(5) In Theorem 2.4, we proved that the deterministic ILT model has a densification power law exponent

$$\frac{\log 3}{\log 2} \approx 1.58.$$

Can we design a random ILT model where changing the parameters of the model gives variable densification power law exponents?

Appendix

The following original code was used to simulate the random ILT model.

```
/*****This file contains the main
 function for the simulation of
the Random ILT Model.*******
Author: Noor Hadi
Year: 2008
File name: test.cpp
**************************************/
#include <cstdlib>
#include<stdio.h>
#include<string.h>
#include <vector>
#include"mat.h"
#include"rand_model.h"
#include"amat.h"
#include"duplication_model.h"
```

```
#include"adjmat.h"
#include <time.h>

using namespace std;

int main(int argc, char** argv)
{
  //Declaring variables
  int i,j,rand_node,si;
  double alpha, beta;

  //Declaring and initializing instances
  // of the class adjmat
  adjmat adj_M(5,"adj_M");
  adjmat adj_M_t(5,"adj_M_t"); //for adjmat
  adjmat adj_M_r(5,"adj_M_r"); //for adjmat

  //Declaring s to hold the size of the matrix
  int* s=new int[2];

  //Declaring T to hold the time-steps that the
```

```
//user will give to the program
int T=atoi(argv[1]);

//Declaring pfile to point to the file to be read
FILE * pfile;

//Initializing the elements of the matrix adj_M
adj_M.set_elem(0,1,1);
adj_M.set_elem(0,2,1);
adj_M.set_elem(1,2,1);
adj_M.set_elem(2,3,1);
adj_M.set_elem(2,4,1);

srand(time(0));

//Initializing the probability alpha
// to the desired value
alpha=1;

//Looping T times and
//choosing between PA & duplication
```

```
  for(i=0;i<T;i++)
    {
      //Generating a random number beta
      beta = (double)(rand())/(RAND_MAX);

      //If beta<1-alpha, do duplication
      if(beta<1-alpha)
{
  //Get and store the size of the matrix adj_M in si
  si=adj_M.get_size();

  //Choosing a node randomly
        rand_node=rand()%(si);

  //Declaring adj_deg_v as a vector
// to store the degrees of the nodes
  vector<int> adj_deg_v(si);

  //Calling the function size and
//storing the degree of the nodes in adj_deg_v
  adj_M.degree(adj_deg_v);
```

```
  //
  delete [] adj_M_t.get_data();

  //Resizing adj_M_t to
//the size of the new adj_M matrix
  adj_M_t = adjmat(adj_M.get_size());

  //Copying the matrix adj_M
//to the matrix adj_M_t
  adj_M.copy(adj_M_t);

  //Resizing the matrix adj_M
  adj_M.resize();

  //Duplicating the chosen node
//"rand_node" and storing it in the matrix adj_M_t
  adj_M_t.dup_node(adj_M,rand_node);
  }
      //else if beta<alpha, do PA
  else
```

```
{
  //Initializing si to the size of the matrix adj_M
  si=adj_M.get_size();

  //Creating a new vector to
//store the degrees of the nodes
  vector<int> adj_deg_vc(si);

  //Storing the degrees of the
//nodes in adj_M in adj_deg_vc
  adj_M.degree(adj_deg_vc);

  //Calling the function b_preferential_choice
     //and storing the node chosen
// preferentially in rand_node
  rand_node=b_preferential_choice(adj_deg_vc);

  //
  delete [] adj_M_r.get_data();

  //Resizing the matrix adj_M_r
```

```
//to the size of the matric adj_M
  adj_M_r=adjmat(adj_M.get_size());

  //Copying the matrix adj_M
//to the matrix adj_M_r
  adj_M.copy(adj_M_r);

  //Resizing the matrix adj_M
  adj_M.resize();

  //Adding the node chosen
//preferentially to the matrix adj_M_r
  adj_M_r.add_node(adj_M,rand_node);
  }
    }

  //Declaring a new variable vs to store size
  int *vs=new int[2];

  //Declaring a new vector adj_vec
//of the same size as the matrix adj_M
```

```
  vector<int> adj_vec(adj_M.get_size());

  //Declaring a new variable counter
  int counter;

  //Setting the vector adj_vec to zeros
  for(counter=0; counter<adj_M.get_size(); counter++)
    {
      adj_vec.at(counter)=0;
    }

  //Storing the degrees of the
//nodes in the matrix adj_M into adj_vec
  adj_M.degree(adj_vec);

  //Finding the maximum degree in
//the vector adj_vec and storing it in adj_max_deg
  int adj_max_deg=b_max_deg(adj_vec);

  //Declaring a new vector adj_deg_dist
// to store the degree distribution
```

```
  vector<int> adj_deg_dist(adj_max_deg);

  //Declaring a new vector adj_cumul_deg_dist
// to store the cumulative degree distribution
  vector<int> adj_cumul_deg_dist(adj_max_deg);

  //Calling the function b_deg_dist
//and storing the degree distribution in adj_deg_dist
  b_deg_dist(adj_vec,adj_deg_dist);

  //Calling the function b_inverse_cumul to
//change the degree distribution to
// a cumulative degree distribution
  b_inverse_cumul(adj_deg_dist,adj_cumul_deg_dist);

  //Creating a new vector adj_deg_vc
//of the same size as the matrix adj_M
  vector<int> adj_deg_vc(adj_M.get_size());

  //Storing the degree of the
//matrix adj_M in the vector adj_deg_vc
```

```
  adj_M.degree(adj_deg_vc);


  //Storing the size of the
// vector adj_cumul_deg_dist in adj_cumul_size
  int adj_cumul_size=adj_cumul_deg_dist.size();


  //Declaring a variable p
  int p;


  //Opening a txt file called "alpha=1.txt"
  pfile=fopen("alpha=1.txt", "w");


  //Looping and writing the log-log cumulative
//degree distribution in the file "alpha=1.txt"
  for(p=1; p<=adj_cumul_size; p++)
    {
        fprintf(pfile,"%f\t %f\n",log((double)p),
  log(double(adj_cumul_deg_dist.at(p-1)+1)));
    }


  //Closing the file
```

```
  fclose(pfile);


  //Exiting main
  return 0;
}


/*****This is the header file for adjmat.cpp*********
Author: Noor Hadi
Year: 2008
File name: adjmat.h
**************************/
#ifndef ADJMAT_H
#define ADJMAT_H
#include <vector>
using namespace std;



class adjmat
{
  //Declaring variables
 private:
```

```
 int s;
 bool * data;
 char * name;

 //Declaring functions
public:
 adjmat(int N,char * Name="matrix");
 ~adjmat();
 void set_one();
 void set_zero();
 int get_size();
 bool get_elem(int i, int j);
 void set_elem(int i, int j, int value);
 void print();
 bool *get_data();
 int node_degree(int node);
 void degree(vector<int>& v);
 void fprint_adjmat(FILE *pFile);
 void resize();
 void copy(adjmat new_copy);
 void add_node(adjmat M,int node);
```

```
  void dup_node(adjmat M,int node);


};


#endif


/*****This file contains the

 implementation of the class adjmat *******

Author: Noor Hadi

File name: adjmat.cpp

*************************************************/


#include "adjmat.h"

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <math.h>

#include <vector>

using namespace std;


//constructor
```

```
adjmat::adjmat(int N,char* Name)
{
  s=N;
  data=new bool[(N*N+N)/2];
  set_zero();
  name=Name;
}


//destructor
adjmat::~adjmat()
{
  ;
}


//Set all the elements of the
// upper diagonal triangle of the matrix to 1
void adjmat::set_one()
{
  int i;
  for (i=0; i<(s*s+s)/2; i++)
    data[i]=1;
```

```
}


//Set all the elements of the
//upper diagonal triangle of the matrix to 0
void adjmat::set_zero()
{
  int i;
  for (i=0; i<(s*s+s)/2; i++)
    data[i]=0;
}


//Get a specific element in row i, column j
bool adjmat::get_elem(int i, int j)
{
  if(i<=j)
   return data[i*s+j-i*(i+1)/2];
  else
  return data[j*s+i-j*(j+1)/2];


}
```

```
//Set a specific element in row i, column j to value
void adjmat::set_elem(int i,int j, int value)
{
  if(i<=j)
  data[i*s+j-i*(i+1)/2]=value;
  else
     data[j*s+i-j*(j+1)/2]=value;
}

//Print the matrix
void adjmat::print()
{
  int i,j;
  printf(name);
  printf(":\n");
  for(i=0; i<s; i++)
    {
      for(j=0; j<i; j++)
{
  printf("%d\t",data[j*s+i-j*(j+1)/2]); //if i<=j
}
```

```
        printf("%d\t",0);
        for(j=i+1;j<s;j++)
{
  printf("%d\t",data[i*s+j-i*(i+1)/2]); //if i>j
}
        printf("\n");
    }
}

//Write the matrix to a file
void adjmat::fprint_adjmat(FILE *pFile)
{
  int i,j;
  for(i=0; i<s; i++)
    {
        for(j=0; j<i; j++)
{
  fprintf(pFile,"%d\t",data[j*s+i]);
}
        fprintf(pFile,"%d\t",0);
        for(j=i+1;j<s;j++)
```

```
{
  fprintf(pFile,"%d\t",data[i*s+j]);
}
      fprintf(pFile,"\n");
    }
}

//Get all the elements in the matrix
bool *adjmat::get_data()
{
  return data;
}

//Find the degree of a specific node
int adjmat::node_degree(int node)
{
  int sum=0;
  for(int i=0; i<s; i++)
    {
      if(get_elem(i,node))
      sum+=1;
```

```
    }

  return sum;
}

//Storing the degree of each node
void adjmat::degree(vector<int> &v)
{
  int i;

  for (i=0; i<s; i++)
    {
      v.at(i)=node_degree(i);
    }

}

//Resizing the matrix
void adjmat::resize()
{
```

```
  s=s+1;
  delete [] data;
  data=new bool[(s*s+s)/2];
  set_zero();
}

//Copying 2 matrices
void adjmat::copy(adjmat new_copy)
{
  int i;
  for (i=0; i<(s*s+s)/2; i++)
    {
      new_copy.data[i]=data[i];
    }
}

//Adding a new node
void adjmat::add_node(adjmat M,int node)
{
  int i,j;
  for( i=0;i<s; i++)
```

```
    {
      for(j=0;j<s; j++)
{
  M.set_elem(i,j,get_elem(i,j));
}
    }
  M.set_elem(node,s,1);


}


//Duplicating a node
void adjmat::dup_node(adjmat M,int node)
{
  int i,j;
  for( i=0;i<s; i++)
    {
      for(j=0;j<s; j++)
{
  M.set_elem(i,j,get_elem(i,j));
}
    }
```

```
  for(i=0;i<s;i++)
    M.set_elem(i,s,get_elem(i,node));
  M.set_elem(node,s,1);


}

//Getting the size of a matrix
int adjmat::get_size()
{
  return s;
}
/*****This is the header file
for duplication_model.cpp *****
Author: Noor Hadi
Year: 2008
File name: duplication_model.h*****************

#ifndef duplication_model_H
#define duplication_model_H
#include "amat.h"
```

```
int b_preferential_choice(vector <int> degree);
void b_deg_dist(vector<int> &A,
vector<int> &degree_distribution);
int b_max_deg(vector <int> a);
void b_inverse_cumul(vector<int>& A,
 vector<int>& ICA);

#endif

/*****This file contains the
implementation of the class duplication_model*****
We note that the code for the functions
to find the maximum degree,
degree distribution and inverse degree
distribution are not included here as
they are easy to implement.
Author: Noor Hadi
Year: 2008
File name: test.cpp
*************************************************/
```

```
#include "duplication_model.h"

#include <string.h>

#include <cstdlib>

#include <stdio.h>


//Choosing a node preferentially

int b_preferential_choice(vector<int> degree)

{

  int leng=degree.size();

  int max_deg=0;

  int sum=0;

  int i;

  double val;

  double alpha;

  double * cumulative_array=new double[leng];

  for (i=0; i<leng; i++)

    {

      sum+=degree.at(i);

    }

  cumulative_array[0]=((double)degree.at(0))/sum;

  for(i=1; i<leng; i++)
```

```
    {
      cumulative_array[i]=cumulative_array[i-1]+
    ((double)degree.at(i))/sum;
    }
  alpha=(double)rand()/RAND_MAX;
  for(i=0; i<leng; i++)
    {
      if(cumulative_array[i]>alpha)
return i;
    }
}
```

Bibliography

[1] Y. Ahn, S. Han, H. Jeong, H. Kwak, S. Moon, Analysis of topological characteristics of huge on-line social networking services, In: *Proceedings of the 16th International Conference on World Wide Web*, 2007.

[2] A. Barabási, R. Albert, Emergence of scaling in random networks, *Science* **286** (1999) 509-512.

[3] G. Bebek, P. Berenbrink, C. Cooper, T. Friedetzky, J. Nadeau, S.C. Sahinalp, The degree distribution of the generalized duplication model, *Theoretical Computer Science* **369** (2006) 234-249.

[4] B. Bhattacharjee, P. Druschel, M. Marcon, A. Mislove, Measurement and analysis of on-line social networks, In: *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, 2007.

[5] B. Bollobás, O. Riordan, J. Spencer, G. Tusnády, The degree sequence of a scale-free random graph process, *Random Structures and Algorithms* **18** (2001) 279-290.

[6] A. Bonato, *A Course on the Web Graph*, American Mathematical Society Graduate Studies Series in Mathematics, Providence, Rhode Island, 2008.

[7] A. Bonato, N. Hadi, P. Horn, P. Pralat, C. Wang, Dynamic models of online social networks, In: *Proceedings of the 6th Workshop on Algorithms and Models for the Web Graph (WAW2009)*, 2009.

[8] F. Chung, L. Lu, T. Dewey, D. Galas, Duplication models for biological networks, *Journal of Computational Biology* **10** (2003) 677–687.

[9] O. Frank, Transitivity in stochastic graphs and digraphs, *Journal of Mathematical Sociology* **7** (1980) 199-213.

[10] G.R. Grimmett, D.R. Stirzaker, *Probability and Random Processes*, 3rd Edition, Oxford University Press, 2001.

[11] B. Huberman, S. Golder, D. Wilkinson, Rhythms of social interaction: messaging within a massive on-line network, In: *3rd International Conference on Communities and Technologies*, 2007.

[12] R. Kumar, J. Novak, A. Tomkins, Structure and evolution of on-line social networks, In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[13] R. Kumar, J. Novak, P. Raghavan, A. Tomkins, On the bursty evolution of Blogspace, In: *Proceedings of the International World Wide Web Conference*, 2003.

[14] J. Leskovec, J. Kleinberg, C. Faloutsos, Graphs over time: densification Laws, shrinking diameters and possible explanations, In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2005.

[15] S. Milgram, The small world problem, *Psychology Today* **2** (1967) 60-67.

[16] J.P. Scott, *Social Network Analysis: A Handbook*, Sage Publications Ltd, London, 2000.

[17] J. Stewart, *Calculus*, Brooks/Cole, Pacific Grove, California, 1999.

[18] S.H. Strogatz, D.J. Watts, Collective dynamics of 'small-world' networks, *Nature* **393** (1998) 440-442.

[19] D.B. West, *Introduction to Graph Theory, 2nd edition*, Prentice Hall, 2001.

[20] H. White, S. Harrison, R. Breiger, Social structure from multiple networks. I.: Blockmodels of roles and positions, *American Journal of Sociology* **81** (1976) 730-780.