Analysing the Graph Expansion of Flipper Maintained Random Graphs

Daniel Baldin

2007
Department of Computer Science
Warburger Straße 100
33098 Paderborn

Analysing the Graph Expansion of Flipper Maintained Random Graphs

A Thesis Presented
by

Daniel Baldin

to

Computer Science

in partial fulfillment of the requirements
for the honors degree of
Bachelor of Computer Science

Prof. Dr. Friedhelm Meyer auf der Heide
and
Dr. Ekkart Kindler

Paderborn, 14th of March, 2007
Declaration

Hereby I ensure that the thesis at hand was created unassisted and without use of other sources than those listed. All phrases taken from other sources are quoted and listed in the section bibliography. This thesis has not been submitted identically or in any other form to an examination office yet.

Paderborn, 14th of March, 2007

Daniel Baldin
## Contents

1 Introduction and Preliminaries ........................................... 1
   1.1 Introduction ................................................................ 1
      1.1.1 Networks and Expanders .................................... 1
      1.1.2 Overview ....................................................... 2
   1.2 Definitions .................................................................. 2
      1.2.1 Graph Terminology and Notations ....................... 2
      1.2.2 Algebraic Terminology and Notations .................. 3

2 Fundamentals ................................................................. 5
   2.1 The Flipper Operation .................................................. 5
      2.1.1 The 1-Flipper Operation ..................................... 5
      2.1.2 Properties of the 1-Flipper Operation .................. 7
   2.2 Expander Graphs ...................................................... 8
      2.2.1 Graph Expansion ............................................... 8
      2.2.2 Expansion and the Spectral Gap ......................... 8
      2.2.3 Bounds of the Spectral Gap ................................. 10
      2.2.4 Expansion in Random Graphs .............................. 13
   2.3 Calculating the Spectral Gap ....................................... 13
      2.3.1 Eigenvalue Problems in General ......................... 13
      2.3.2 Efficient Calculation of the Spectral Gap ............. 14

3 Concept ................................................................. 17
   3.1 Candidate Graphs .................................................... 17
      3.1.1 The Torus Graph .............................................. 17
      3.1.2 The Circle Graph ........................................... 18
      3.1.3 The Candy Graph ............................................ 20
      3.1.4 The Egg Graph ............................................... 21
      3.1.5 Combined Graph Types ..................................... 22
   3.2 Simulation Environment ........................................... 23

4 Analysis ................................................................. 25
   4.1 Convergence Behavior ............................................... 25
      4.1.1 The Convergence Variance ................................ 25
      4.1.2 The Convergence Process .................................. 27
   4.2 The n-d Dependency ................................................ 29
      4.2.1 Incrementing the Node-Count ........................... 29
Chapter 1 Introduction and Preliminaries

1.1 Introduction

1.1.1 Networks and Expanders

Expander graphs and their properties have been a subject of great interest since the last three decades in the field of computer science and a lot of studies have revolved around it. One part of these studies covers the analysis of random processes in graphs. Those random processes are of great interest since many applications in computer science can benefit from it. Not only in the field of cryptography, coding and algorithms but also in the area of peer to peer networks these processes have been discovered to be very useful. But why are expander graphs so useful? Intuitively expander graphs are highly connected graphs which means that it is easy to get from one vertex to another vertex within very few steps. Such a property is of great importance if one thinks of designing large peer to peer networks. Although methods are known to construct expander graphs [LAP88] [Mar88] none of them may be used in networking concepts. However using the probabilistic method it has been shown that randomly chosen graphs are very good expander graphs with high probability.

This is where the 1-flipper operation comes into play. In order for peer to peer networks to benefit from the properties of expander graphs some process is needed to transform the graph into an expander and afterwards maintain this property. Construction of such a transformation is complicated by requirements such as d-regularity, connectivity and distributed execution. An operation which fulfills all these requirements is the flipper operation, a random process, defined by Schindelhauer and Mahlmann [MS05]. It has been proved that this operation transforms every d-regular connected graph into an expander graph with high probability. Unfortunately, few is known about the amount of operations needed by this transformation to create a graph with high expansion.

In order to answer this question some possibility is needed to measure the expansion. Fortunately, lots of measurements have been found to quantify the expansion property such as the edge expansion. Regrettably, all those measurements are NP-hard to compute. However it was shown that the expansion ratio is tied to
the spectrum of the graph and especially to the second eigenvalue of the adjacency matrix of the graph. This makes it possible to efficiently analyse graph transformations like the 1-flipper operation by the change of the expansion property resulting by the transformations made on the graph.

1.1.2 Overview

This thesis has the main goal to answer the question: "How many flipper operations are necessary to transform a bad graph into an expander graph?". This question will be answered by experimental analysis of the flipper operation.

In order to answer this question the reader is first introduced to the flipper operation as proposed by Schindelhauer and Mahlmann [MS05]. Afterwards expander graphs are introduced and the main properties of these graphs are been discussed especially the relation between the expansion property and the spectrum of the graph. The reader is also introduced to the mathematical techniques used to efficiently calculate the spectrum which is needed for the analysis. In chapter 3 the graph types are presented which have been used for testing and have been discovered to be bad conditioned. Chapter 4 covers the analysis of the tests made and shows how the flipper operation performs on the graphs introduced in chapter 3. In chapter 4 the conclusion will be given. The thesis is then finished with the summary.

1.2 Definitions

To avoid confusion about the terminology this section covers the main graph and algebraic terminology used in this thesis.

1.2.1 Graph Terminology and Notations

Definition 1.2.1. Throughout this work the following standard terminology will be used.

a) An undirected graph $G$ is a 2-tupel $G = (V, E)$ with an vertex set $V$ and an edge set $E \subset V \times V$. The size of the graph is denoted by $N = |V|$. The edge set does not contain self loops. This means $\forall x \in V : \{x, x\} \notin E$.

b) The degree of a vertex, denoted as $\text{deg}(x)$, is the amount of incident edges. A graph is called $d$-regular if all vertices have degree $d$.

c) For any subset $S \subset V$ the neighbor set $N(S)$ is defined as $N(S) = \{ j \in V | \exists i \in S \{i, j\} \in E \}$

d) For any subsets $S \subset V$ and $T \subset V$ let $E(S, T)$ define the set of edges between any nodes $v \in S$ and $w \in T$. If $T = S$ we denote the set as $E(S)$.
1.2 Definitions

e) A graph \( G = (V, E) \) is connected if for any pair of vertices \((i, j)\) there exists a path from \(i\) to \(j\).

f) Let \( p_{i,j} \) be a path from \(i\) to \(j\). The distance between \(i\) and \(j\) is defined as the length of a shortest path between \(i\) and \(j\): \( d(i, j) = \min_{p_{i,j}} |p_{i,j}| \)

g) The diameter \( K \) of a graph is defined as the longest distance between any pair of vertices on the graph: \( K = \max_{i,j} d(i, j) \).

h) A path \( p_{i,j} \) in a graph \( G = (V, E) \) may be given explicitly as a n-tupel \((v_1, v_2, v_3, .., v_n)\) with \( v_1 = i \) and \( v_n = j \). Hereby \( v_k \) denotes the k-th vertice on the path from \(i\) to \(j\).

i) The adjacency matrix of an n-vertex graph \( G \), denoted as \( A = A(G) \), is an \( n \times n \) matrix whose \((u, v)\) entry is the number of edges in \( G \) between vertex \(u\) and vertex \(v\).

Unless otherwise noted a graph will be a d-regular undirected graph throughout this thesis. Since every edge in an undirected graph \( G = (V, E) \) is bidirectional it follows that \( \forall i \in V : \forall j \in V : \{i, j\} \in E \rightarrow \{j, i\} \in E \).
To simplify the view on graphs with size \( N \) the vertices may also be labeled by the numbers 1..\( N \).

1.2.2 Algebraic Terminology and Notations

Definition 1.2.2. Since this work comprises of many algebraic theorems the following standard algebraic terminology will be used.

a) Let \( \mathbb{R}^{n \times m} \) denote the space of the \( n \times m \) real matrices. A vector \( v \in \mathbb{R}^n \) will be treated as a column vector out of \( \mathbb{R}^{n \times 1} \).

b) Let \( I_{n \times m} \) denote the \( n \times m \) identity matrix. \( I_n \) denotes the \( n \times n \) identity matrix. The notation \( I \) will be used whenever it is obvious which dimension is meant.

c) Let \( A \) be a matrix. \( A^t \) denotes the matrix transpose. A matrix \( A \) is symmetric if \( AA^t = I \).

d) The dot product \( \langle v, w \rangle \) of two vectors \( v \in \mathbb{R}^n \) and \( w \in \mathbb{R}^n \) is defined as \( \langle v, w \rangle = \sum_{i=1}^{n} v_i \cdot w_i \). The \( L_1 \)-norm of a vector \( x \in \mathbb{R}^n \) is denoted as \( |x|_1 \) and is defined as \( |x|_1 = \sum_{i=1}^{n} |x_i| \). The \( L_2 \)-norm is denoted as \( \|x\| \) and defined as \( \|x\| = \sqrt{\langle x, x \rangle} \).

In the following the term ”with high probability” (w.h.p) describes the probability \( p \geq 1 - n^{-c} \) and ”asymptotically almost surely” (a.a.s) the probability \( p \geq 1 - o(1) \).
Chapter 2 Fundamentals

In this chapter the fundamentals for this thesis are introduced. In the first part the flipper operation is introduced which performance is going to be analysed in the chapter analysis. In the second section expander graphs and its most important properties are presented. The following section will cover the mathematics for solving the specific eigenvalue problem involved in calculating the spectral gap.

2.1 The Flipper Operation

2.1.1 The 1-Flipper Operation

As it was already mentioned in the introduction random processes on graphs are of great importance in many areas. In the special case of peer to peer networks one is interested in constructing robust networks with small diameter so messages do not need to travel a long way and search queries can reach many nodes on the network within very few steps. These networks can be described as $d$-regular undirected graphs in many cases. By using a $d$-regular graph some fairness is induced since messages passed over a long random walk will pass every peer with nearly uniform probability. Naturally peer to peer networks need some process to guarantee these kind of standards. Todays widely used peer to peer networks do not posses such mechanisms. In this chapter the Flipper-Operation is introduced which is supposed to be a quickly converging transformation capable of guaranteeing good network properties. Experimental analysis on how the flipper operation really performs will be given in chapter 4.

The Flipper-Operation is a random transformation which generates a uniform probability distribution of regular connected graphs in the limit. The prove of this thesis can be found in [MS05]. At this point the 1-Flipper Operation by Schindelhauer and Mahlmann [MS05] and its randomized version are introduced.

![Figure 2.1: The 1-Flipper Operation $F^1_p$](image)
Chapter 2 Fundamentals

Definition 2.1.1 (1-Flipper).
Consider a d-regular undirected graph \( G = (V, E) \) and four distinct nodes \( u_1, u_2, u_3, u_4 \in V \) forming a path \( P = (u_1, u_2, u_3, u_4) \) in \( G \). Then, if \( \{u_1, u_3\}, \{u_2, u_4\} \notin E \) the 1-Flipper operation \( F^1_p \) transforms the graph \( G \) to the graph \( F^1_p(G) = (V, E') \) with

\[
E' := (E \setminus \{\{u_1, u_2\}, \{u_3, u_4\}\}) \cup \{\{u_1, u_3\}, \{u_2, u_4\}\}
\]

We denote the edges \( \{u_1, u_2\} \) and \( \{u_3, u_4\} \) as the flipping edges and \( \{u_2, u_3\} \) as the hub edge of the operation. Figure 2.1 illustrates the 1-Flipper operation.

In the following the randomized version of the 1-Flipper operation is given.

Algorithm 1 (Random 1-Flipper).

Choose random edge \( \{u_2, u_3\} \in E \)
Choose random node \( u_1 \in N(u_2) \setminus \{u_3\} \)
Choose random node \( u_4 \in N(u_3) \setminus \{u_2\} \)

if \( \{u_1, u_3\}, \{u_2, u_4\} \notin E \) then

\[
E := E \setminus \{\{u_1, u_2\}, \{u_3, u_4\}\}
\]

\[
E := E \cup \{\{u_1, u_3\}, \{u_2, u_4\}\}
\]

In contrast to the normal 1-Flipper operation the randomized version chooses the flipping edges randomly after choosing a random hub edge \( \{u_2, u_3\} \in E \). Hereby the random selection is supposed to be done uniformly.

In Figure 2.2: The Random 1-Flipper Operation illustrated on a circular graph after 0, 100 and 250 flips.

Example 2.1.1. Figure 2.2 shows the Random 1-Flipper Algorithm performing on a circular 4-regular graph with 50 nodes. At the beginning every node is connected with its closest 2 neighbors to the right and to the left. After 100 flips one can see that the flipper operation locally changed the neighboring set of the nodes and thus the graph looks a little bit more randomized. After 250 flips the graph looks totally different to the starting graph and the connections between the nodes look...
like randomly chosen. The last graph in fact can be considered as a good expander graph as we will see later on.

2.1.2 Properties of the 1-Flipper Operation

In the last section the 1-Flipper operation was defined. Now the properties of the transformation are analysed. The next Lemma will show that the flipper operation maintains connectivity which means that there is no chance to disconnect parts of the graph.

**Lemma 2.1.1.** The 1-Flipper operation preserves connectivity and d-regularity.

**Proof.** The d-regularity is maintained since every node looses one edge and receives a new one. Since the graph was connected before the operation there exists a path $p_{i,j}$ between any pair of vertices $i$ and $j$. This path may be destroyed if an edge on the path is chosen as a flipping edge in the operation but since all nodes affected by the operation stay connected there is a new path connecting $i$ and $j$.

A delimiting factor for applying the 1-flipper operation is the existence of a triangle such that for a hub edge $\{u_2, u_3\}$ nodes $v$ with $\{u_2, v\}, \{u_3, v\} \in E$ exist. In this case if one of the flipping edges is chosen as $\{u_2, v\}$ or $\{u_3, v\}$ the edge can not be flipped since one of the edges already exists. In this case we say the flipper operation failed.

![Figure 2.3: A flipper operation failing due to a triangle existing over the hub edge.](image)

This observation will be of importance later on when we have to choose some appropriate graphs for testing.

In the following let $G \xrightarrow{i} G'$ denote the predicate that $G'$ is derived from $G$ by applying $i$ flipper operations and $C_{n,d}$ denote the set of all connected d-regular graphs with $n$ nodes. The next Lemma is given without proof but the interested reader may find it in [MS05].

**Lemma 2.1.2.** Let $G_0$ be a d-regular connected graph with $n$ nodes and $d > 2$. Then in the limit the Random 1-Flipper operation constructs all connected d-regular graphs with the same probability

$$\lim_{t \to \infty} P \left[ G \xrightarrow{t} G' \right] = \frac{1}{|C_{n,d}|}$$
2.2 Expander Graphs

As it was already mentioned in the introduction a high expansion ratio is a desirable characteristic of a graph. In this section we are going to define what the expansion ratio is and thus what kind of graphs may be considered as expander graphs. It will soon be clear that some approximation of the expansion ratio is needed. For this reason the spectral gap will be introduced and analysed. Finally this section will discuss the question of optimal expansion and the relation of random graphs to it.

2.2.1 Graph Expansion

The Graph Expansion is defined differently in many papers. The mostly used expansion ratios are the vertex-expansion and the edge-expansion. Throughout this paper the latter one will be used.

Definition 2.2.1 (Expansion Ratio).

(1) The **Edge Boundary** of a set $S$, denoted as $\partial S$, is $\partial S = E(S, \overline{S})$. This denotes the set of edges emanating from the set $S$ to its complement $\overline{S}$.

(2) The **(edge) Expansion Ratio** of a graph $G$, denoted $h(G)$, is defined as:

$$h(G) = \min \{ S \mid |S| \leq \frac{n}{2} \} \frac{|\partial S|}{|S|}$$

(3) A graph $G$ is called a $\beta$-**expander** if $h(G) \geq \beta$ and $\beta > 0$.

As mentioned before there also exists a definition which counts the number of neighboring vertices instead of counting the amount of emanating edges from a set $S$. The definition of a $\beta$-expander means nothing else than that for every set $S$ of $G$, with $|S| \leq \frac{n}{2}$, the amount of emanating edges is $\beta$ times higher than the amount of nodes in the set $S$.

2.2.2 Expansion and the Spectral Gap

By definition 2.2.1 it is clear that it is in fact NP-hard to calculate the expansion ratio. But since the expansion ratio of a graph $G$ is a property of great interest, especially if one thinks of networks, it would be convenient to calculate some approximation of it. This is possible by analysing the adjacency matrix $A = A(G)$ of $G$ (compare 1.2.2 i). Since the matrix $A$ is real and symmetric the matrix has $n$ real eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$. The eigenvalues of $A(G)$ are called the **Spectrum** (denoted as $\sigma(G)$) of the graph $G$. Knowing the spectrum of the graph one can gather a lot of information about it :

- $\lambda_1 = d$ and the corresponding eigenvector is $v_1 = (1/\sqrt{n}, .., 1/\sqrt{n})^t$. 
2.2 Expander Graphs

- The graph is connected $\iff \lambda_1 > \lambda_2$

Not only these properties of a graph are reflected by the spectrum but also information about the expansion ratio can be calculated out of it. The next Theorem will show the relation of the spectrum to the expansion ratio.

**Theorem 2.2.1.** Let $G$ be a $d$-regular graph with spectrum $\lambda_1 \geq \ldots \geq \lambda_n$. Then

$$\frac{d - \lambda_2}{2} \leq h(G) \leq \sqrt{2d(d - \lambda_2)}$$

The Term $d - \lambda_2$ is also known as the **Spectral Gap**. Thus a large spectral gap implies a high expansion. Since this theorem is one of the most important ones for this thesis a proof for the lower-bound will be given in the following. The proof of the upper-bound is quite long and thus it is omitted here but it can be found in [LW03]. Now the lower-bound $\frac{d - \lambda_2}{2}$ of the expansion ratio is going to be proofed.

For some set $S$ we define $1_S$ to be the characteristic vector of $S$ (i.e., the $i$-th component of the vector is 1 if $i \in S$ and zero otherwise). For $S \subset V$ the vector $x$ is defined as :

$$x = \bar{\bar{S}} \cdot 1_S - |S| \cdot 1_{\bar{\bar{S}}}$$

$$\|x\|^2 = \|\bar{\bar{S}}\|^2 |S| + |S|^2 |\bar{\bar{S}}| = |\bar{\bar{S}}| |S| (|\bar{\bar{S}}| + |S|) = n |\bar{\bar{S}}| |S|$$

$$xAx^t = 2 |E(S)||\bar{\bar{S}}|^2 + 2 |E(\bar{\bar{S}})| |S|^2 - 2 |E(S, \bar{\bar{S}})| |\bar{\bar{S}}| |S|$$

Since $G$ is $d$-regular the following equations hold:

$$d |S| = 2 |E(S)| + |E(S, \bar{\bar{S}})|$$

$$\equiv d |S| - |E(S, \bar{\bar{S}})| = 2 |E(S)|$$  \hspace{1cm} (1)
\[ d |\bar{S}| = 2 |E(\bar{S})| + |E(S, \bar{S})| \]
\[ \equiv d |S| - |E(S, \bar{S})| = 2 |E(\bar{S})| \]  \hfill (2)

Using (1) and (2) it follows
\[ x A x^t = n d |\bar{S}| |S| - n^2 |E(S, \bar{S})| \]

Since \( \lambda_2 \geq x A x^t / \|x\| \) it follows
\[ \lambda_2 \geq \frac{n d |\bar{S}| |S| - n^2 |E(S, \bar{S})|}{n |S| |S|} = d - \frac{n |E(S, \bar{S})|}{|S| |S|} \]

Now we fix \( S \) to be a set for which holds
\[ h(G) = \frac{|E(S, \bar{S})|}{|S|} \]

Since \( |\bar{S}| \geq \frac{n}{2} \) it follows that
\[ \lambda_2 \geq k - \frac{n h(G)}{|S|} \geq d - 2 h(G) \]
\[ \equiv \frac{d - \lambda_2}{2} \leq h(G) \]

2.2.3 Bounds of the Spectral Gap

Now since we know what exactly the Spectral Gap is the question arises whether there exist some bounds for the spectral gap. Particularly with regard to some calculation of the spectral gap this information may be useful. For the following analysis the ”Gerschgorin Theorem” [KS06] will be needed:

**Theorem 2.2.2. (Gerschgorin)**
Let
\[ K_i := \left\{ z \in \mathbb{C} | z - a_{ii} \leq r_i := \sum_{k=1,k \neq i}^{n} |a_{ik}| \right\} \]
then every eigenvalue \( \lambda \) lies in \( \bigcup_{i=1}^{n} K_i \)

Because the only graphs we are considering are d-regular undirected graphs without self-loops it is clear that for all \( a_{ii} \) of \( A(G) \) it follows that \( a_{ii} = 0 \) and \( \forall i : \sum_{k=1,k \neq i}^{n} |a_{ik}| = d \). Out of this the next lemma directly follows:
2.2 Expander Graphs

Lemma 2.2.2. Let $G = (V, E)$ be a $d$-regular undirected graph without self-loops. Then for the spectrum $\sigma(G) = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ with $n = |V|$ it follows that $\lambda_i \in [-d, d].$

Proof. It is clear that $\forall i : \sum_{k=1, k \neq i}^n |a_{ik}| = d.$ Due to the fact that $A$ is symmetric it follows that all eigenvalues are real values. Thus it directly follows that $\lambda_i \in [-d, d]$ which is the real subset of $\bigcup_{i=1}^n K_i.$

Out of Lemma 2.2.2 we can infer that the spectral gap $d - \lambda_2$ is always bounded by $0 \leq d - \lambda_2 \leq 2d.$ In the first case $\lambda_2$ has to be exactly $d$ which means that the eigenvalue $\lambda = d$ has at last multiplicity 2. The latter case results from $\lambda_2 = -d$ which in turn leads to a multiplicity of $n - 1$ for the eigenvalue $\lambda = -d.$ This in fact is an unusual case and later on we will see that this actually never happens.

The upper bound of the spectral gap we just calculated of course does not satisfy our requirements. Thus we need to get some more precise information on the upper bound of the spectral gap. For this we will change our perspective in order to examine optimal expanders first and then draw a conclusion out of it. We are now going to ask the question ”What is the greatest expansion ratio that can be reached?”'. This question also revolves around the study of extremal problems and quite a lot is known about it. If infinite graphs are considered as well the infinite d-regular tree $T_d$ is clearly the ultimate expander.

Figure 2.4: A part of the 3-regular infinite tree $T_3$

Figure 2.4 illustrates the infinite 3-regular tree $T_3.$ For the d-regular infinite tree the following statements hold true.

1. The edge expansion of the d-regular infinite tree is $d - 2$

2. The spectrum of $T_d$ is the interval $[-2\sqrt{d - 1}, 2\sqrt{d - 1}]

The proofs of these statements can be found in [SH06]. Out of these statements it can be inferred that for any d-regular graph $G$ $h(G) \leq d - 2 - o(1).$ It also follows that the second eigenvalue of $G$ is at least $2\sqrt{d - 1} - o(1).$ The following theorem will underline this thesis [SH06].

Theorem 2.2.3. (Alon-Boppana) For every d-regular graph

$$\lambda \geq 2\sqrt{d - 1} - o_n(1),$$

where $\lambda = \max(|\lambda_2|, |\lambda_n|)$ and $o_n(1)$ a quantity that tends to zero for $n \to \infty.$
Chapter 2 Fundamentals

With theorem 2.2.3 it is now possible to give a better estimation of the spectral gap. We are now leaving out the term \( o(n) \) in the formula in theorem 2.2.3 since we are talking about graphs with \( n \) being big. Thus it follows that \( d - \lambda \leq d - 2\sqrt{d-1} \) which reduces the interval for the spectral gap from \([0, 2d]\) to \([0, d - 2\sqrt{d-1}]\) which is a much better estimation. Upon this knowledge we are now going to define 2 functions \( \rho_1(d), \rho_2(d) \).

**Definition 2.2.2. (upper-bounds for the lower and upper-bound of \( h(G) \))**

\[
\rho_1(d) = \frac{d - 2\sqrt{d-1}}{2}
\]

\[
\rho_2(d) = \sqrt{2d(d - 2\sqrt{d-1})}
\]

\( \rho_1(d) \) is now an approximation of the upper-bound for the changing lower-bound of the expansion ratio of a series of graphs \( G_1, G_2, \ldots \) and may be used to check whether a graph \( G_i \) may be considered as a good expander or not. Analogical \( \rho_2(d) \) is an upper-bound for the evolving upper-bound of the expansion ratio \( h(G) \). Since the computation of \( \rho_1(d) \) involves only 1 square root \( \rho_1(d) \) should be used.

![Figure 2.5: The functions \( \rho_1(d) \) and \( \rho_2(d) \) illustrated.](image)

Figure 2.5 illustrates these 2 functions. As one can see in the diagram the function \( \rho_2(d) \) is bigger than \( d \) for every \( d > 14 \) which means that the upperbound \( h(G) \leq \sqrt{2d(d - \lambda_2)} \) is an overestimation for good expander graphs with \( d > 14 \) since the expansion ratio is clearly bounded by \( d \) from above.
2.3 Calculating the Spectral Gap

2.2.4 Expansion in Random Graphs

Now that we analyzed the bounds of the spectral gap we are going to have a look at random graphs. Quite a lot is known about the expansion ratio of random graphs but we will only concentrate on the most important realization which is important for this thesis. The powerful theorem following next will categorize the expansion ratio of random $d$-regular graphs.

**Theorem 2.2.4.** Let $G$ be a random connected $d$-regular graph. Then $G$ is an $\Theta(d)$-expander graph a.a.s.

The proof of this thesis was given by B. Bollobas in the year 1988 and can be found in his paper [Bo88]. Since the flipper operation produces connected random $d$-regular graphs in the limit (compare Lemma 2.1.2) it directly follows:

**Corollary 2.2.1.** Consider any $d$-regular graph $G$ with $n$ nodes. Then in the limit the Random 1-Flipper operation establishes an expander graph after a sufficiently large number of applications a.a.s.

Now we know that the 1-Flipper operation generates a series of graphs $G_i$ for which there exists an index $k$ such that $G_k$ is an $\Theta(d)$-expander graph. The analysis on the amount of operations needed for this transformation is the main goal of this thesis and will be done in the following chapters.

2.3 Calculating the Spectral Gap

As it was shown in the last section the spectral gap gives away some vital information about the graph. However calculating the spectral gap involves calculating the eigenvalues of $A(G)$. Of course one is interested in calculating the eigenvalues as fast as possible which makes it important to use some sophisticated algorithm. This section will shortly cover eigenvalue problems in general and then give an algorithm to efficiently calculate the spectral gap.

2.3.1 Eigenvalue Problems in General

Eigenvalue Problems can be found in many applications and thus many algorithms are known to solve the specific problem to find values $\lambda$ for which the following equation holds: $Ax = \lambda x$. In many cases the matrix $A$ has some specific characteristics and sometimes not all eigenvalues of $A$ are needed to be calculated. Regarding these specific problems one needs to decide which algorithm to take to solve the eigenvalue problem. In many cases the matrix $A$ is symmetric which makes it possible to use algorithms specialised in solving those eigenvalue problems. The Jacobi Algorithm which can be found in [KS06] may be mentioned here. For unsymmetric matrices the QR-Transformation is definitely a good choice [KS06]. If one is only interested in calculating the absolut biggest eigenvalues $|\lambda_1| \geq |\lambda_2| \geq ..$ the **Vector-Iteration Algorithm** is the best choice here. Some advanced form of
it will be introduced in the next section in order to efficiently calculate the spectral gap.

2.3.2 Efficient Calculation of the Spectral Gap

The calculation of the spectral gap $d - \lambda_2$ of a graph $G$ involves the computation of the second biggest eigenvalue of $A = A(G)$. Due to the fact that $A$ is symmetric (since $G$ is undirected) and positive definite the Vector-Iteration Algorithm by F.L. Bauer is clearly the best method for this calculation. Another fact which makes this algorithm superior to other algorithms in this specific case is the fact that the graph $G$ is $d$-regular. This makes the implementation easier and the computation a lot faster since every row of $A$ only contains $d$ entries unequal to zero.

In the following the vector iteration by F.L. Bauer is given which calculates the first $m$ absolut biggest eigenvalues of a given matrix $A$ by iterative matrix multiplications.

**Algorithm 2 (Vector Iteration).**

1. Choose arbitrary but linear independent vectors
   \[
   U_0 = \begin{pmatrix}
   \ddots & \ddots & \ddots \\
   \ddots & u_1^{(0)} & \ddots \\
   \ddots & \ddots & \ddots \\
   \ddots & \ddots & \ddots \\
   \end{pmatrix} 
   \in \mathbb{R}^{n,m}
   \]
   Set $k = 0$.

2. Orthogonalize $U_k$ by determining an upper right triangle matrix $R_k \in \mathbb{R}^{n,m}$
   \[
   Z_k = U_k R_k \text{ with } Z_k^T Z_k = I_m
   \]
   (for example using the gram-schmidt orthogonalization algorithm [IDR86])

3. Multiply the matrices:
   \[
   U_{k+1} = AZ_k.
   \]

4. Check for convergence:
   if $\sum_{i \neq j} |r_{i,j}| < \epsilon$ stop
   else set $k = k + 1$ and go to (2).

Since $A$ is positive definite, it can be shown [Bau69] that:

\[
\lim_{k \to \infty} Z_k = V
\]

14
2.3 Calculating the Spectral Gap

\[ \lim_{k \to \infty} R_k = D \quad (n \times m \text{ diagonal matrix}) \]

with \( V \) being the matrix containing the eigenvectors of \( A \) and \( D \) containing the eigenvalues. The algorithm yields the following stable convergence rate:

**Theorem 2.3.1.** Let \( A \) have the eigenvalues \( \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p \geq \lambda_{p+1} \geq \ldots > 0 \).
Let \( v_i \) be an appropriately chosen eigenvector to the eigenvalue \( \lambda_i \) then

\[ \|v_i - x_j^{(k)}\| = O(q^k) \]

where \( q = \max(\lambda_{j+1}/\lambda_j, \lambda_j/\lambda_{j-1}) \)

This theorem states that every column \( x_j \) is linearly converging towards the nearest eigenvector with \( q \). This may in fact take longer if the eigenvalue \( \lambda_i \) is close to one of the eigenvalues \( \lambda_{i+1} \) or \( \lambda_{i-1} \).

Still open is the question about the performance of the algorithm. For this we first need to take a deeper look at the convergence behavior. One problem of this algorithm is the occurrence of unstable convergence. Unstable convergence means that the columns of \( U_k \) do not converge to the eigenvectors \( v_1, \ldots, v_m \). This may happen if the space spanned by the columns of \( U_k \) is orthogonal to at least one of the eigenvectors. Fortunately the probability for this to happen is very low. Even if it happens roundoff-errors usually turn the unstable convergence into delayed stable convergence. However it is possible to avoid this situation by adding another column to \( U_k \) which is initialized randomly every iteration and orthonormalized against the rest of the columns.

Another question is the amount of operations needed for this calculation. Clearly the matrix multiplications take the most time each step. But since the graph was \( d \)-regular the multiplication time is bounded by \( O(d \cdot n^2 \cdot m) \) using a proper implementation (i.e. storing the matrix as an adjacency list).

The considerate reader may now annotate that the algorithm 2 given above may not calculate the second biggest eigenvalue \( \lambda_2 \). In fact the algorithm calculates the \( m \) absolut biggest eigenvalues. However the algorithm can be easily adopted to calculate the \( m \) biggest eigenvalues instead. To accomplish this task we make use of a spectral shift. From chapter 2.2.3 we know that all eigenvalues of \( A(G) \) lie in the interval \([-d, d]\) thus it is possible to shift all eigenvalues so they lie in the range \([0, 2d]\). Step (3) of the algorithm is then modified to

\[ U_{k+1} = (A + d \cdot I)Z_k \]

The resulting eigenvalues \( \lambda_{\text{shift}} \) will then be the shifted eigenvalues of \( A \) by \( d \) into positive direction. To receive the real eigenvalues \( \lambda_A \) we only need to subtract \( d \) afterwards : \( \lambda_A = \lambda_{\text{shift}} - d \).
Chapter 3 Concept

In the last chapter the fundamentals have been introduced which are needed to analyse the performance of the flipper operation. We got to know that the expansion ratio is a main property of good network topologies and how to measure the expansion ratio using the spectral gap. Now it is also clear that the flipper operation generates a random distribution over all possible d-regular graphs and thus in the limit creates expander graphs but until now it is still unclear how long it really takes to turn a graph into an expander graph using the flipper operation. Before we can try answering this question by performing tests it is important to identify bad graphs with properties that are totally contradictorily to properties of good expander graphs. With these kind of graphs we may then simulate the flipper operation and afterwards analyse the results. Therefor the next step is to find suitable candidate graphs.

3.1 Candidate Graphs

For the testing process it is important to choose a good set of candidate graphs in order to get meaningful test results. Finding these graphs is anything but easy. The difficulty in finding these graphs is founded in the restriction that the graphs have to be regular. We are now going to present some graphs and their properties which have been created based on the knowledge from the last chapter and have been identified as good test graphs for the flipper operation.

One of the good properties of expander graphs is the small mixing time of random walks [LW03] and the logarithmic diameter [SH06]. Thus it is a good idea to identify graphs with high mixing times and high diameter in order to test the flipper operation. One of the graphs with a high mixing time and high diameter is the Torus which will be introduced next. Another property of a graph, which may slow down the convergence process of the flipper operation, is the amount of triangles in the graph. As we saw in chapter 2.1 a triangle in a graph may make the flipper operation fail. Thus the amount of triangles is another important property of a bad graph.

3.1.1 The Torus Graph

The Torus Graph is one of the standard graphs in mathematics or network topologies. Analysis showed that the Torus is not rapidly mixing and lacks other proper-
ties expander graphs possess which makes this graph a suitable graph to test the flipper operation. Another property, which makes this graph a good test candidate, is the fact that the diameter of a Torus is not logarithmic but lies in $O(\sqrt{n})$.

Figure 3.1: The Torus Graph illustrated.

Figure 3.1 illustrates two 2-dimensional Torus Graphs. Graph a) shows a Torus with $10 \times 20$ nodes and makes it obvious why this graph topology is called a Torus. Graph b) illustrates a Torus rolled out onto a plane with $8 \times 10$ nodes. We say the first parameter defines the height the latter one the width of the Torus. One of the properties which makes this graph better conditioned than the graphs presented next is the fact that the Torus has no triangles at all. Thus in the beginning there is no chance for a failing flipper operation on this graph type.

We are only going to test the flipper operation on the 2-dimensional Torus since adding more dimensions will reduce the diameter leading to a better convergence speed.

### 3.1.2 The Circle Graph

We already got to know the Circle Graph in chapter 2. Within the Circle Graph each node is only connected to its closest neighbors to the left and to the right. Figure 3.2 illustrates a Circle Graph with 20 nodes and degree 4. One of the properties of the Circle Graph is the high diameter which stands in contrast to the logarithmic diameter of expander graphs. The following table will show the evolution of the diameter with increasing amount of nodes.

Figure 3.1 shows that the diameter is linearly increasing with increasing amount of nodes.

As we mentioned at the beginning of this chapter the amount of triangles plays a major role for the possibility of a failing flipper operation (compare 2.1.2). Thus
3.1 Candidate Graphs

<table>
<thead>
<tr>
<th>Degree</th>
<th>Nodes</th>
<th>Diameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>200</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>300</td>
<td>75</td>
</tr>
<tr>
<td>4</td>
<td>400</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>500</td>
<td>125</td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>150</td>
</tr>
</tbody>
</table>

Table 3.1: Table showing the linear increasing diameter of the Circle Graph.

we will now analyse the graph on the amount of triangles. The next Lemma will give us a nice possibility to calculate the amount of triangles in a graph.

**Lemma 3.1.1.** Let $A = A(G)$ be the adjacency matrix of graph $G = (V, E)$. Let $A(3) = A \cdot A \cdot A$. Then $\Delta = \frac{1}{6} \sum_{i=1}^{N} a_{ii}^{(3)}$ is the amount of triangles in the graph $G$.

**Proof.** Since the element $a_{ij}$ of $A^{(3)}$ is the number of paths $p_{i,j}$ from $i$ to $j$ with length 3 the diagonal elements $a_{ii}$ denote the amount of paths from the node $i$ to itself with length 3. Since every path $p_{i,i}$ with length 3 is a triangle the sum of the diagonal elements is the number of triangles times 6 since every triangle can be described as the following paths: $(v_i, v_j, v_k, v_i)$, $(v_i, v_k, v_j, v_i)$, $(v_j, v_k, v_i, v_j)$, $(v_i, v_j, v_k, v_i)$ and $(v_k, v_j, v_i, v_k)$. \qed

Using Lemma 3.1.1 the amount of triangles in the Circle Graph calculates to the following values.

Figure 3.2 shows that the Circle Graph comprises of many triangles compared to the amount of edges. With higher degree the amount of triangles grows even larger than the amount of edges giving a very high probability for a failing flipper operation. We can also see that the amount of triangles linearly increases with increasing amount of nodes and polynomial with increasing degree. In the next chapter we will see how this property influences the flipper operation.
3.1.3 The Candy Graph

The next graph, which seems to be a good candidate, is the Candy Graph. Figure 3.3 shows a Candy Graph with 60 nodes and degree 6. The graph mainly consists out of cliques of size $d$ where $d$ is also the degree of the graph. Within a clique all nodes are connected with each other having a degree of $d - 1$ inside of the clique. To regain the $d$-regularity each of the nodes inside a clique is adjacent to a connecting node. The connecting node connects two cliques with each other. Figure 3.3 b) illustrates this and makes it obvious why we chose to call this graph Candy. Similarly to the Circle Graph this graph does not have a logarithmic diameter. On top of that the chance for the 1-flipper operation to fail on this graph type is pretty high since this graph type comprises of a lot of triangles as we will see next.

Table 3.2: Table showing the amount of triangles in the Circle Graph under some selected graph parameters.

| Nodes | Degree | $\Delta$ | $|E|$ |
|-------|--------|----------|------|
| 100   | 4      | 100      | 200  |
| 120   | 4      | 120      | 240  |
| 140   | 4      | 140      | 280  |
| 100   | 6      | 300      | 300  |
| 120   | 6      | 360      | 360  |
| 100   | 8      | 600      | 400  |

Figure 3.3: The Candy Graph illustrated.

Analogues to the Circle Graph we want to examine the diameter and the amount of triangles of this graph topology. The following table shows the calculated values for some selected parameters.

As we can see in table 3.3 the amount of triangles is even higher than the amount of
3.1 Candidate Graphs

| Nodes | Degree | $\Delta$ | $|E|$ | Diameter |
|-------|--------|----------|------|----------|
| 100   | 4      | 120      | 200  | 30       |
| 120   | 4      | 144      | 240  | 36       |
| 140   | 4      | 168      | 280  | 42       |
| 98 $^1$ | 6   | 364      | 294  | 21       |

Table 3.3: Table showing the amount of triangles in the Candy Graph under some selected graph parameters.

The degree of the graph calculates to $d = k - 1$ where $k$ is the size of the cliques. The amount of nodes can only be a multiple of $k$. Let $\{u_{k,i}, u_{k,j}\}$ denote this edge. Afterwards the edges $\{u_{k-1,j}, u_{k,i}\}$ and $\{u_{k,j}, u_{k+1,i}\}$ are established. Figure 3.4 demonstrates the construction.

The exact value of $100$ can not be constructed due to the topology of the graph.

3.1.4 The Egg Graph

The Egg Graph is quite similar to the Candy Graph but differs in an important kind of way. The Egg Graph consists of multiple cliques but no explicit connecting node is used this time. To maintain connectivity between the cliques a random edge is deleted inside the clique with index $k$. Let $\{u_{k,i}, u_{k,j}\}$ denote this edge. Afterwards the edges $\{u_{k-1,j}, u_{k,i}\}$ and $\{u_{k,j}, u_{k+1,i}\}$ are established. Figure 3.4 demonstrates the construction.

The degree of the graph matches the diameter of the Candy Graph with the same amount of nodes and the same degree. Additionally the growth of the amount of triangles shows the same behavior as it does on the Circle Graph. The diameter is also quite high making this graph topology a good candidate for testing the flipper operation.

The degree of the graph calculates to $d = k - 1$ where $k$ is the size of the cliques. The amount of nodes can only be a multiple of $k$. Let $\{u_{k,i}, u_{k,j}\}$ denote this edge. Afterwards the edges $\{u_{k-1,j}, u_{k,i}\}$ and $\{u_{k,j}, u_{k+1,i}\}$ are established. Figure 3.4 demonstrates the construction.

The degree of the graph calculates to $d = k - 1$ where $k$ is the size of the cliques. The amount of nodes can only be a multiple of $k$. Let $\{u_{k,i}, u_{k,j}\}$ denote this edge. Afterwards the edges $\{u_{k-1,j}, u_{k,i}\}$ and $\{u_{k,j}, u_{k+1,i}\}$ are established. Figure 3.4 demonstrates the construction.

The exact value of $100$ can not be constructed due to the topology of the graph.
Chapter 3 Concept

| Nodes | Degree | Δ  | $|E|$ | Diameter |
|-------|--------|----|-----|---------|
| 100   | 4      | 140| 200 | 30      |
| 120   | 4      | 168| 240 | 36      |
| 140   | 4      | 196| 280 | 42      |
| 98 $^1$ | 6    | 420| 294 | 21      |
| 99 $^1$ | 8    | 847| 396 | 16      |

Table 3.4: Table showing the amount of triangles in the Egg Graph under some selected graph parameters.

edges $\{u_{k-1,j}, u_{k,j}\}$ and $\{u_{k,j}, u_{k+1,i}\}$. Later on we will see how this affects the flipper operation.

3.1.5 Combined Graph Types

Now that we got to know all these graph types another idea is to combine the graphs in hope for the negative effects to sum up. We call these kind of graphs combined graph types. The basic idea is to either substitute nodes by whole graphs or for example cliques by other graphs. The combinations are fairly limited since the d-regularity has to be maintained. One combination we want to present here is the graph we will call the Double Layered Egg Graph or the Flower Graph.

Figure 3.5: The Double Layered Egg Graph illustrated.

Figure 3.5 illustrates the Double Layered Egg Graph with degree 6, four cliques in layer 1 and six cliques in layer 2. Basically the graph consists of two layers. The

$^1$The exact value of 100 can not be constructed due to the topology of the graph.
first layer, denoted by the green circle in the figure, defines the amount of subgraphs in the second layer. The second layer, denoted by the blue circles, consists of independent Egg subgraphs. The subgraphs are only connected with each other by a clique in layer one. Thus a clique in the first layer is also part of a subgraph in layer two.

Let us have a look at the diameter and the amount of triangles in this graph as well. Since the same amount of nodes may result from different amount of cliques in each layer we will look at it under some selected parameters.

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Degree</th>
<th>L1 Cliques</th>
<th>L2 Cliques</th>
<th>Δ</th>
<th>E</th>
<th>Diameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>4</td>
<td>2</td>
<td>10</td>
<td>134</td>
<td>200</td>
<td>31</td>
</tr>
<tr>
<td>100</td>
<td>4</td>
<td>10</td>
<td>2</td>
<td>110</td>
<td>200</td>
<td>19</td>
</tr>
<tr>
<td>120</td>
<td>4</td>
<td>2</td>
<td>12</td>
<td>162</td>
<td>240</td>
<td>37</td>
</tr>
<tr>
<td>120</td>
<td>4</td>
<td>12</td>
<td>2</td>
<td>132</td>
<td>240</td>
<td>22</td>
</tr>
<tr>
<td>98(^1)</td>
<td>6</td>
<td>2</td>
<td>7</td>
<td>410</td>
<td>294</td>
<td>21</td>
</tr>
<tr>
<td>98(^1)</td>
<td>6</td>
<td>7</td>
<td>2</td>
<td>385</td>
<td>294</td>
<td>13</td>
</tr>
</tbody>
</table>

Table 3.5: Table showing the amount of triangles in the Double Layered Egg Graph under some selected graph parameters.

As we can see in table 3.5 the amount of triangles and the diameter can differ strongly with different parameters. Whenever the amount of layer 1 cliques rises the amount of triangles goes down as well as the diameter. This is feasible since each of the layer 1 cliques looses two edges inside of the clique in order to establish connections to other cliques whereas a clique in layer 2 looses only one edge. Thus it seems like the Double Layered Egg Graph with only two cliques in layer 1 is the graph with the worst properties in this graph type. Compared to the Egg Graph 3.4 the amount of triangles is slightly lower. However the diameter is slightly higher on small graphs.

### 3.2 Simulation Environment

In this section we will shortly introduce the simulation environment which was programmed as part of this thesis to test the flipper operation. The environment called FSim (Flipper Simulator) was written in C++ with cross-platform support in mind. It has been successfully compiled under windows, linux and macosx. The Simulation Environment uses the algorithm 1 introduced in chapter 2.3 to efficiently calculate the spectral gap. For easier generation of test cases the graphs have directly been integrated and are produced online. Test series can be generated by command line parameters or by using an external xml file. For detailed information see the documentation of the code provided with this thesis.

\(^1\)The exact value of 100 can not be constructed due to the topology of the graph.
Chapter 4 Analysis

In the last chapter we presented graphs which have been identified to possess bad properties compared to properties of good expander graphs. We introduced graphs on which the flipper operation has a higher probability to fail and thus may slow down the convergence process to an expander graph. In this chapter we will present the results of the tests made using the FSim tool introduced in the last chapter. We will first have a look at the general convergence process of the operation and its distribution on the different graphs. We will then analyse the evolution of the amount of operations needed for convergence with changing parameters of the graph types. Finally we will compare the results on the graphs with each other but before we get there we will need the following definition, which is used throughout this chapter:

Definition 4.0.1. Let \( G, G' \) be two graphs with \( G \xrightarrow{\mathcal{F}} G' \). We say the flipper operation turned the graph \( G \) into an expander graph when the lower-bound \( \frac{d - \lambda_2}{2} \) of \( h(G') \) satisfies:

\[
\frac{d - \lambda_2}{2} \geq 0.98 \rho_1(d).
\]

Whenever a graph \( G' \) was turned into an expander graph by the flipper operation we say the graph series induced by the flipper operation converged into a good expander graph.

4.1 Convergence Behavior

In this section we will have a deeper look at the convergence behavior of the flipper operation. Since the flipper operation is a random algorithm it is interesting to analyse the variation of the amount of operations needed for the convergence process.

4.1.1 The Convergence Variance

In the following figure 4.1 the distribution of the number of 1-flipper operations for convergence on two graphs is shown. Picture a) shows the distribution on the Circle Graph with 2000 nodes and degree 10. The test was repeated 8000 times and the number of operations have been sorted into 30 bins in order to see the distribution. The mean value lies at 70539 operations. The expected value \( E(X) \), where \( X \) denotes the amount of operations needed, lies at 70691 and thus being pretty close to the mean value. The variance computes to 1459.
Figure 4.1: The convergence distribution of the 1-flipper operation on the Circle Graph a) and the Candy Graph b). The needed amount of operations have been sorted into bins $P(\text{Ops})$.

In comparison to that b) shows the distribution on the Candy Graph with 1980 nodes and degree 10. The mean value is 77213 and the expected value 77738. It is easy to see that the expected value is much farther away from the mean value than it is in the Circle Graph. Underlying this observation the variance calculates to 1787. This observation is our first indication that the flipper operation performs worse on the Candy Graph than on the Circle.

The distributions as shown in the figure 4.1 are typical for the 1-flipper operation and look the same on the torus and the Egg Graph. With this tests one may infer that the distribution will be very similar on other graphs and may only differ by variance, expected value and mean value. The following table gives some information on the evolution of the variance with increasing amount of nodes. The tests have been repeated 2000 times each giving a high enough resolution.

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Degree</th>
<th>Mean</th>
<th>$E(X)$</th>
<th>$VAR(X)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2200</td>
<td>10</td>
<td>8.728661e+004</td>
<td>8.761324e+004</td>
<td>1.865595e+003</td>
</tr>
<tr>
<td>2420</td>
<td>10</td>
<td>9.748513e+004</td>
<td>9.779534e+004</td>
<td>2.006000e+003</td>
</tr>
<tr>
<td>2640</td>
<td>10</td>
<td>1.078373e+005</td>
<td>1.081932e+005</td>
<td>2.130541e+003</td>
</tr>
<tr>
<td>2860</td>
<td>10</td>
<td>1.183060e+005</td>
<td>1.187855e+005</td>
<td>2.360606e+003</td>
</tr>
<tr>
<td>3080</td>
<td>10</td>
<td>1.288527e+005</td>
<td>1.295116e+005</td>
<td>2.551360e+003</td>
</tr>
</tbody>
</table>

Table 4.1: Table showing the evolution of the expected value and the variance with increasing amount of nodes on the Candy Graph.

As we can see in table 4.1 the variance is linearly increasing with linearly increasing amount of nodes.
4.1 Convergence Behavior

Let’s have a look at the evolution of the variance on the Egg Graph. The tests on the Egg Graph have also been repeated 2000 times each.

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Degree</th>
<th>Mean</th>
<th>$E(X)$</th>
<th>$VAR(X)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2200</td>
<td>10</td>
<td>1.139063e+005</td>
<td>1.145213e+005</td>
<td>7.369837e+003</td>
</tr>
<tr>
<td>2420</td>
<td>10</td>
<td>1.272598e+005</td>
<td>1.277330e+005</td>
<td>8.085523e+003</td>
</tr>
<tr>
<td>2640</td>
<td>10</td>
<td>1.409014e+005</td>
<td>1.414499e+005</td>
<td>8.927024e+003</td>
</tr>
<tr>
<td>2860</td>
<td>10</td>
<td>1.540829e+005</td>
<td>1.548155e+005</td>
<td>9.295579e+003</td>
</tr>
<tr>
<td>3080</td>
<td>10</td>
<td>1.676225e+005</td>
<td>1.683719e+005</td>
<td>1.032467e+004</td>
</tr>
</tbody>
</table>

Table 4.2: Table showing the evolution of the expected value and the variance with increasing amount of nodes on the Egg Graph.

Similarly to the Candy Graph the variance is also linearly increasing here. However the variance is nearly a power of ten higher. We can also see that the variance is strongly increasing. For instance the Egg Graph with 3080 nodes and degree 10 already has a variance which is nearly about a tenth part of the expected value. Thus we need to consider this development of the variance in our tests and have to repeat each test case more often if necessary to get an appropriate mean value.

### 4.1.2 The Convergence Process

We are now going to have a look at the convergence process. Figure 4.2 shows the evolution of the lower bound in a test case on the Circle Graph with 1800 nodes and degree 8. The upper-bound $\rho_1(8)$ for the lower-bound has been marked as a red line at the top. Basically a flipper maintained graph is situated inside one of three phases. The first one is the initial phase in which the lower-bound is not changing much or falling from time to time. This region is marked with 1 in the picture. The second phase is the expanding phase. In this phase the lower-bound for the expansion ratio is rapidly rising although it might fall and rise again from time to time as seen in the graph. This is feasible since the flipper operation is still a random transformation. The expanding phase is denoted with 2 in the figure. The last phase denoted with 3 is the stable phase. At this time the graph series converged into a good expander graph and this property is maintained through the flipper operation afterwards as we will see in 4.5. Due to Theorem 2.2.3 the lower-bound can not increase any more and thus stays around the value $\rho_1(d) \pm O(1)$.

The convergence behavior is very similar on the graphs introduced in chapter 3. Figure 4.3 shows the evolution of the lower-bound in a test case on the Circle, Candy and the Egg Graph. For better comparison the parameters of the graphs have been chosen in a way that the amount of nodes and the degree correspond to each other. Obviously the functions look very similar, only the 3 phases of the convergence process are differently. We can see that phase two of the Candy Graph convergence process starts later than phase two of the Circle Graph convergence process and phase two of the Egg Graph starts latest. In addition to that the phase
Figure 4.2: The three convergence phases demonstrated on the Circle Graph with 1800 nodes and degree 8.

Figure 4.3: The evolution of the lower-bound on the Candy, Circle and the Egg Graph with degree 8 and 1800 nodes.
lengths are varying. As we can see the second phase of the Egg Graph takes longer than the phases of the other two graphs. Later on we will see why this is the case.

4.2 The n-d Dependency

In this section we will have a deeper look at the dependency of the convergence process on the parameters \( n \) and \( d \) with \( n \) denoting the amount of nodes in the graph and \( d \) the degree. We will have a look at the different graphs and change the amount of nodes in the graphs to analyse the dependency on the amount of nodes in the convergence process. Afterwards we will change the degree and have a look at the evolution of the amount of operations needed in that case.

4.2.1 Incrementing the Node-Count

We will first have a look at the Circle Graph and the Candy Graph. Figure 4.4 shows the evolution of the needed operations for convergence with increasing amount of nodes. The blue graph shows the amount of operations for the Candy Graph. The magenta one the Circle Graph. Both graphs had degree 8 and 1080 nodes at the beginning. Each step the amount of nodes has been increased by 180. Each of these test cases has been repeated 20 times and the mean value was taken to form the continuous lines. For each test case the maximum value and minimum value is denoted by a triangle pointing downwards and upwards respectively.

![Graph showing the evolution of the amount of operations needed by incrementation of the node count.](image)

Figure 4.4: Graph showing the evolution of the amount of operations needed by incrementation of the node count.
Chapter 4 Analysis

It is obvious that the amount of operations linearly increases on both graphs. To underline this the function \( f(x) = 38 \cdot x \) has been drawn as well which bounds the amount of operations from above in our test set. We can also infer that the 1-flipper operation performs worse on the Candy Graph since the slope is a little bit higher.

![Graph showing the increasing amount of operations with increasing amount of nodes on the Egg Graph.](image)

Figure 4.5 shows that the number of flipper operations needed on the Egg Graph is also linearly increasing with linearly increasing amount of nodes although quite a lot more operations are needed than on the Candy Graph. The degree in this tests has been fixed to 8. Since the variance is much higher on the Egg Graph (compare 4.1) each test case has been repeated 60 times to get an appropriate mean value. The maximum and the minimum value of each test case has been marked as triangles again. The test results on the worst and the best performing Double Layered Egg Graph (alias Flower Graph) has been drawn as well. We can observe that the Flower Graph behaves very similar to the Egg Graph. This is probably due to the fact that both graph types consist of similar topologies.

Analysing the Torus is quite a little bit more complicated since the node set may be changed by increasing either the width or the height of it or even both. Since the Torus Graph is symmetric \(^{1}\) it is sufficient to increase just one of the parameters in the first case. Table 4.3 shows the test cases used in the tests on the Torus.

---

\(^{1}\)A 10x20 Torus Graph is just a 20x10 Torus Graph rotated and thus yields the same properties.
same amount of nodes could not be produced on all Torus Graphs in the test cases as a result of the different topologies. However the deviation is kept so small that no noticeable effect is produced.

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Torus 3 × n</th>
<th>Torus 10 × n</th>
<th>Torus n × n</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>3 × 133</td>
<td>10 × 40</td>
<td>20 × 20</td>
</tr>
<tr>
<td>625</td>
<td>3 × 208</td>
<td>10 × 63</td>
<td>25 × 25</td>
</tr>
<tr>
<td>850</td>
<td>3 × 283</td>
<td>10 × 85</td>
<td>29 × 29</td>
</tr>
<tr>
<td>1075</td>
<td>3 × 358</td>
<td>10 × 108</td>
<td>33 × 33</td>
</tr>
<tr>
<td>1300</td>
<td>3 × 433</td>
<td>10 × 130</td>
<td>36 × 36</td>
</tr>
<tr>
<td>1525</td>
<td>3 × 508</td>
<td>10 × 153</td>
<td>39 × 39</td>
</tr>
</tbody>
</table>

Table 4.3: Table showing the test settings for the Torus Graph with increasing amount of nodes.

As we can see in figure 4.6 the Torus behaves like the other graphs only the slope is differently. We can see that the Square-Torus has the smallest slope. The bigger the ratio between the height and the width of the Torus grows the bigger the slope gets as well. Compared to the other graphs the Torus performs better as we can see at the slope of the function \( f(x) = 27 \cdot x \) which bounds our test results from above. This is our first indication that the Torus seams to be better conditioned for the the problem of transforming the graph into an expander graph using the flipper operation.
4.2.2 Incrementing the Degree

Changing the degree is the other option in order to alter the graphs. We want to find out how the amount of flipper operations needed changes if the degree of the graph is changed. For this we fix the amount of nodes to 1080 and linearly increase the degree of the graph. The topology of the Candy Graph forces us to use the settings as shown in 4.4 for the tests.

<table>
<thead>
<tr>
<th>Clique-Size</th>
<th>Num-Cliques</th>
<th>Resulting Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>120</td>
<td>1080</td>
</tr>
<tr>
<td>10</td>
<td>98</td>
<td>1078</td>
</tr>
<tr>
<td>12</td>
<td>83</td>
<td>1079</td>
</tr>
<tr>
<td>14</td>
<td>72</td>
<td>1080</td>
</tr>
<tr>
<td>16</td>
<td>64</td>
<td>1088</td>
</tr>
</tbody>
</table>

Table 4.4: Table showing the test settings for the Candy Graph.

Since we can only change the amount of cliques to reach the node count of 1080 we chose the closest possible value but because the deviation has been kept small the results have not been influenced much. Figure 4.7 shows the result of this test.

As we can see in figure 4.7 the amount of operations linearly increases with the degree. The Circle Graph has also been tested and the result is shown as the red
4.2 The n-d Dependency

graph in figure 4.7. Supporting the results we already got from the tests before the flipper operation performs better on the Circle Graph than on the Candy Graph. We can also see that the slope of the Circle Graph is smaller than the slope of the Candy Graph. The cyan line shows the function \( f(x) = 3000 \cdot x + 14000 \) which again bounds the amount of operations from above.

Next we will have a look at the behavior of the Egg Graph. For this purpose the Egg Graph was fixed to 1800 nodes and the degree was linearly increased. The amount of cliques has been altered accordingly to come up with 1800 nodes or the closest possible value. In order to overcome the problem of the high variance the tests have been repeated 100 times.

![Figure 4.8: The operations needed for convergence on the Egg Graph with increasing degree.](image)

Similarly to the test results on the other graphs the amount of operations needed on the Egg Graph also increases linearly with linearly increasing degree. However the slope of the curve has more than doubled in comparison to the Candy Graph (4.7). The observation that the difference between the maximum and the minimum value of each test case is also growing underlines the observations made in 4.1. We also draw the evolution of the amount of needed operations on the two selected Flower Graphs we used for comparison in the last section. As one may already have supposed the evolution is again very similar to the Egg Graph but the slope is slightly lower. This is probably due to the fewer amount of triangles in the graph (compare 3.4 and 3.5).
4.2.3 Incrementing both Parameters

In this section we are going to analyse the amount of needed operations if we increase both parameters at the same time. From the last section we got to know that the amount of needed operations linearly depends on the parameters $d$ and $n$. Thus we expect that the amount of operations needed for convergence resembles some polynomial function lying in $O(n \cdot d)$. We tested the flipper operation on the Egg Graph by increasing the degree each time by one and increasing the amount of cliques in a way such that the amount of nodes increased by 200 each test case.

![Graph showing the evolution of the amount of operations needed with increasing amount of nodes and degree.](image)

Figure 4.9: Graphs showing the evolution of the amount of operations needed with increasing amount of nodes and degree.

We can see the result of the test in figure 4.9. As expected the graph resembles a polynomial function as we can see in a). In figure 4.9 b) the log-log diagram of the test results is shown. As we can see the function behaves like a perfect straight line in b). The maximum and minimum value of each test case has been marked as triangles again. Underlining our tests made in section 4.1 the variance seams to be growing in $O(n \cdot d)$ as well.

4.3 Comparative Analysis

In the last two sections the flipper operation was analysed on the different graph types. We are now going to deepen our knowledge of the reason why the flipper operation behaves so differently. Therefor we will first have a look at the failure behavior of the flipper operation on the different graph types.
4.3 Comparative Analysis

4.3.1 The Failure Behavior

The amount of failed operations is an important criteria for the performance of the flipper operation. We already saw in 4.1 and 4.2 that the amount of operations for convergence can differ enormously. We will find a reason for this on the following pages. Let us first compare the amount of failed operations in the convergence process on the different graph types with a fixed amount of nodes and degree with each other.

![Graph showing the amount of failed flipper operations on some test graphs in the convergence process.](image)

Figure 4.10: Graph showing the amount of failed flipper operations on some test graphs in the convergence process.

Figure 4.10 shows the failed operations in the convergence process on the different graph types. The test was made on graphs with 1800 nodes and degree 8 except the Torus Graphs, which had degree 4. Every 150 flipper operations the amount of failed flipper operations since the last measurement is drawn. As we can see the Egg Graph performs worst again. At the beginning of the process nearly every operation fails (> 95%). This is due to the high amount of triangles in the graph. We will compare the evolution of the amount of triangles with the failing operations next. The Candy and the Circle Graph perform nearly as bad as the Egg Graph but both of them start with a lower chance for a failing operation. Interesting is the development on the Torus Graph. At the beginning the amount of failed operation starts to rise. This is due to the fact that the torus has no triangles at all at the beginning (compare 3.1.1). All in all the amount of flipper-operations needed seems to be closely related to the amount of failing flipper operations.

As we already stated before it seems like the evolution of the amount of triangles
plays the major role for the amount of failing operations. This is plausible since
the flipper operation is a random process and its utilization is only limited through
the existence of triangles. Thus we are now going to compare the evolution of the
failing flipper operations with the amount of triangles in the graph.

![Graph comparing the amount of triangles with the amount of failed operations in the Egg and the Torus Graph.]

Figure 4.11: Graph comparing the amount of triangles with the amount of failed operations in the Egg and the Torus Graph.

Figure 4.11 shows the evolution of the amount of triangles and the failing flipper operations on the Egg Graph with 450 nodes and degree 4 as well as the worst performing Torus Graph with 450 nodes. The thick painted lines represent the amount of triangles. As supposed we can see the direct connection between the amount of triangles and the amount of failing flipper operations. We can also observe the increasing amount of triangles at the beginning of the convergence process of the Torus which leads to a higher amount of failing operations. In contrast to that the amount of triangles is rapidly falling in the Egg Graph. At some point both graphs possess nearly the same amount of triangles but in the end the Egg Graph still takes longer to converge due to its initial amount of failed operations and its topology.

### 4.3.2 The Diameter

We just found out that the amount of failed operations is tied to the amount of
triangles in the graph. However in the last chapter we also chose the graph types
because the diameter was known to be high. Therefor it is interesting to observe
the diameter of the graph during the convergence process. Since we know that
expander graphs possess logarithmic diameter \[O(\log(n))\] we expect the diameter of
the test graphs to be reduced to some value lying in \(O(\log(n))\). The following test
was made on the Egg Graph and the $3 \times k$ Torus with 450 nodes (thus $k = 150$) and degree 4.

4.3.3 The Number of Needed Operations

Until now we only analysed the total amount of flipper operations needed. But how does the flipper operation perform on the graphs if we only count the successful operations? Let's compare the amount of successful flipper operations on the different graphs with each other.
Chapter 4 Analysis

<table>
<thead>
<tr>
<th>Degree</th>
<th>Nodes</th>
<th>Egg</th>
<th>Candy</th>
<th>Circle</th>
<th>Torus $3 \times k$</th>
<th>Torus $k \times k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Ops}_S$</td>
<td>4</td>
<td>900</td>
<td>18215</td>
<td>16678</td>
<td>15845</td>
<td>15454</td>
</tr>
<tr>
<td>$\text{Ops}_T$</td>
<td>4</td>
<td>900</td>
<td>21351</td>
<td>19507</td>
<td>18916</td>
<td>17079</td>
</tr>
<tr>
<td>$\text{Ops}_S$</td>
<td>4</td>
<td>1800</td>
<td>47172</td>
<td>41510</td>
<td>42925</td>
<td>37295</td>
</tr>
<tr>
<td>$\text{Ops}_T$</td>
<td>4</td>
<td>1800</td>
<td>50768</td>
<td>44464</td>
<td>45830</td>
<td>40152</td>
</tr>
</tbody>
</table>

Table 4.5: Table comparing the amount of successful operations and total operations for convergence on the different graph types.

Table 4.5 shows the amount of successful flipper operations needed for convergence in the row denoted with $\text{Ops}_S$. The row below denoted with $\text{Ops}_T$ shows the total amount of operations. Each test has been repeated 60 times and the mean value was taken. As we can observe the difference between the total amount of operations and the amount of successful operations stays the about the same on the Egg, Candy and Circle Graph. This is feasible since the amount of triangles linearly grows with the amount of nodes (compare chapter 3) and thus the graphs yield a similar probability for a failing flipper operation on graphs with higher amount of nodes.

The following table compares the amount of successful operations and the total amount of operations with increasing degree.

<table>
<thead>
<tr>
<th>Degree</th>
<th>Nodes</th>
<th>Egg</th>
<th>Candy</th>
<th>Circle</th>
<th>Flower $2 \times k$</th>
<th>Flower $k \times 2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Ops}_S$</td>
<td>4</td>
<td>1800</td>
<td>47172</td>
<td>41510</td>
<td>42925</td>
<td>42320</td>
</tr>
<tr>
<td>$\text{Ops}_T$</td>
<td>4</td>
<td>1800</td>
<td>50768</td>
<td>44464</td>
<td>45830</td>
<td>48603</td>
</tr>
<tr>
<td>$\text{Ops}_S$</td>
<td>8</td>
<td>1800</td>
<td>55724</td>
<td>45045</td>
<td>43712</td>
<td>52800</td>
</tr>
<tr>
<td>$\text{Ops}_T$</td>
<td>8</td>
<td>1800</td>
<td>76820</td>
<td>60315</td>
<td>54540</td>
<td>72392</td>
</tr>
</tbody>
</table>

Table 4.6: Table comparing the amount of successful operations and total operations for convergence on the different graph types with increasing degree.

Table 4.6 shows the test results on the different graphs with increasing degree. As we can see the Flower Graphs behave very similar to the Egg Graph which is due to the similar topology of both graph types. The Flower Graph performs better again as we already saw in the sections before. On all graphs presented here the difference between the total amount of operations and the amount of successful operations strongly increases with increasing degree. Again we can accredit this behavior with the strongly increasing amount of triangles in the graphs (compare chapter 3).

4.4 Intermediate Results

In this chapter we analysed the flipper operation on the different graph types presented in chapter 3. We found out that the amount of flipper operations needed linearly depends on the amount of nodes and degree on the graph types we used. We also analysed the behavior of the flipper operation on similar graphs like the
Double Layered Egg Graph (alias Flower Graph) and the Egg Graph. The important observation made here is that the flipper operation behaves similar on similar graphs (Figure 4.5, Figure 4.8, Table 4.6). Only this observation allows us to generalize the results made here onto other graphs that have not been tested. We also found out that the major factor influencing the performance of the flipper operation is the amount of triangles in the graph. Thus if one wants to forecast the performance of the flipper operations on graph types not covered here the amount of triangles is an important factor to consider. Next we will have a look at the longtime behavior of the flipper operation.

4.5 Longtime Behavior

In this section we will analyse the longtime behavior of the flipper operation. We got to know that the flipper operation turns a graph into an expander graph after a sufficient amount of operations. But what happens afterwards? To get some more information on the behavior we set up the following test case. The Egg Graph has been tested on 500 nodes and degree 4 over a period of $8 \cdot 10^6$ operations. Every 250 operations the actual lower-bound of the spectral gap has been drawn. Assuming the flipper operation is performed concurrently on all peers every 20 seconds, which is a realistic time period, then this test represents a total time period of about 7 month. Figure 4.13 shows the result of this test.

![Figure 4.13: The lower-bound of the Egg Graph with 500 nodes and degree 4 over a period of $8 \cdot 10^6$ operations.](image)

We can see that the lower-bound stays around the value $\rho_1(4)$ nearly all the time. Twice in this long period the lower-bound drops down close to 0. These places are marked with 1. and 2. respectively. Although the lower-bound drops this does not imply that the expansion ratio gets lower as well since the lower-bound is still just a bound. Important is the fact that the flipper operation reestablishes a high lower-bound directly afterwards. The reason for the reduction of the lower-bound is probably a series of bad chosen flipper operations. Since the lower-bound was settled around 0.28 in this case small absolute changes in the lower-bound led to high relative changes. This rises the question whether we can observe these phenomena in graphs with higher degree as well. The following test was
performed on the Egg Graph with 500 nodes again but instead of degree 4 we chose the degree 9.

Figure 4.14: The lower-bound of the Egg Graph with 500 nodes and degree 9 over a period of \(8 \cdot 10^6\) operations.

Figure 4.14 shows the result of this test case. We can observe that the lower-bound is subject to the same absolute changes in this test case. But since the lower-bound lay around 1.7 this time the relative changes have been fairly small. During the whole time period the lower-bound never dropped down to 0. This leads to the conclusion that graphs with higher degree behave much more stable under the influence of the flipper operation. This is feasible since after the flipper operation established a high expansion for the first time a much longer series of bad chosen flipper operations is needed to strongly reduce the lower-bound since the amount of edges is higher as well. Putting it all together we can say that the flipper operation is capable of continuously maintaining a high expansion ratio.
Chapter 5  Conclusion

In the last chapter the convergence behavior of the flipper operation and the dependency on the amount of nodes and the degree on different graph types have been analysed. Now we will draw the conclusion on what we learned.

5.1 Results

Referring to chapter 4.2 the amount of flipper operations needed for convergence linearly depends on the amount of nodes and the degree of the graph. In all graph types tested the amount of operations increased linearly with increasing amount of nodes or degree although with different slopes. Since the graph types chosen have been graphs with bad properties such as high diameter or high amount of triangles we assume that on most of the normal network topologies the amount of flipper operations needed is also bounded by some function lying in $O(n \cdot d)$. We also saw that the flipper operation behaves similar on similar graph topologies. This allows the assumption that other graph topologies which are like the topologies presented here will show a similar behavior.

The 1-flipper operation is also supposed to be applied concurrently which of course leads to some proper implementation to ensure connectivity. This is not covered here but can be found in [MS05]. With the concurrency the amount of operations needed splits up nearly equally. For example, if we consider table 4.5 again, the amount of needed operations for every peer in the network on the Egg Graph, which turned out to be the worst graph found, with 1800 nodes calculates to about 30 executions. This calculates to only 10 minutes if the flipper operation is performed every 20 seconds. Thus the 1-flipper operation turns out to be a good operation to quickly transform a graph into an expander graph.

At this point we also want to reference the topology of today’s typical peer to peer networks as for example the gnutella network [Gnu]. Figure 5.1 shows a snapshot of a gnutella network. The snapshot was taken from [B.S00]. The gnutella network topology is a tree like topology and such topologies are way better conditioned than our topologies we chose for the tests. Thus we may infer that the flipper operation transforms graphs we can find in today’s peer to peer networks even faster into an expander graph than it does on the graphs chosen in this thesis and consequently improves the networks overall performance.
Chapter 5 Conclusion

Figure 5.1: A Snapshot of the Gnutella Network

5.2 Future Work

As a result of this thesis some questions arise. If we reconsider the evolution of the amount of failed flipper operations \[4.10\] it is clear that there is a relation between the actual lower-bound of the expansion ratio and the amount of failed flipper operations in the last time period. Now the question arises if it is still possible to infer this relation by locally observing the amount of failed operations at only one peer without global information. This might be of interest for the implementation of the algorithm in order to allow some reduction of the amount of flips needed per time period.

Another work in the future might compare the k-flipper [MS05], which is the extension of the 1-flipper that was not part of this thesis, and the 1-flipper on some real network basis. Since flipping edges may induce lots of overhead traffic in the network some comparative analysis regarding the efficiency based on the overall network traffic produced may be of great interest.
Chapter 6 Summary

In this chapter the contents of this thesis is summarized. This paper started with an introduction stating the main goal of this thesis and introducing some graph and algebraic terminologies used throughout this thesis. The following chapter then covered the fundamentals for this thesis. The 1-flipper operation was introduced and its properties have been presented. Shortly afterwards expander graphs have been presented in depth. We got to know the expansion ratio and its relation to the spectral gap. We found some bounds for the spectral gap which have been of great importance for the testing process afterwards. In that section we also realized that the 1-flipper operation transforms a graph $G$ into an expander graph after a sufficient amount of operations. The next section covered the efficient calculation of the spectral gap which was implemented in the Simulation Environment programmed for this thesis. In the chapter concept we introduced the testing graphs and analysed their properties. We chose graph types with bad properties compared to properties of good expander graphs. At the end of that chapter we shortly introduced the Simulation Environment $FSim$. The next chapter covered the analysis of the tests made. First the convergence process of the 1-flipper operation was examined. We found out that the convergence variance can differ enormously thus giving the responsibility to repeat test cases often enough to get some meaningful information. Afterwards we analysed the convergence process. We observed that a graph $G_i$ of the series $G_0, G_1, ..., G_n$ induced by the flipper operation is situated in one of three phases. The subsequent sections yielded that the amount of operations needed for convergence is linearly depending on the degree and the amount of nodes on all graphs tested. Soon afterwards we found the relationship between the amount of failing flipper operations and the amount of triangles in the graph. At the end of that chapter we covered the long time behavior of the flipper operation and saw that the flipper operation is capable of maintaining a high expansion ratio. In the chapter conclusion the analysis was summed up and a relation to todays typical network topologies was stated. The thesis then finished with some interesting topics for future works.
Bibliography


## List of Tables

3.1 Table showing the linear increasing diameter of the Circle Graph. ........................................... 19
3.2 Table showing the amount of triangles in the Circle Graph under some selected graph parameters. ......................... 20
3.3 Table showing the amount of triangles in the Candy Graph under some selected graph parameters. ......................... 21
3.4 Table showing the amount of triangles in the Egg Graph under some selected graph parameters. ......................... 22
3.5 Table showing the amount of triangles in the Double Layered Egg Graph under some selected graph parameters. ......................... 23
4.1 Table showing the evolution of the expected value and the variance with increasing amount of nodes on the Candy Graph. ......................... 26
4.2 Table showing the evolution of the expected value and the variance with increasing amount of nodes on the Egg Graph. ......................... 27
4.3 Table showing the test settings for the Torus Graph with increasing amount of nodes. ................................. 31
4.4 Table showing the test settings for the Candy Graph. ......................... 32
4.5 Table comparing the amount of successful operations and total operations for convergence on the different graph types. ......................... 38
4.6 Table comparing the amount of successful operations and total operations for convergence on the different graph types with increasing degree. ................................. 38
## List of Figures

2.1 The 1-Flipper Operation $F_1^k$. .............................................. 5
2.2 The Random 1-Flipper Operation illustrated on a circular graph after 0,100 and 250 flips. .............................................. 6
2.3 A flipper operation failing due to a triangle existing over the hub edge. ......................................................... 7
2.4 A part of the 3-regular infinite tree $T_3$. ........................................ 11
2.5 The functions $\rho_1(d)$ and $\rho_2(d)$ illustrated. ............................... 12

3.1 The Torus Graph illustrated. .................................................. 18
3.2 The Circle Graph with 20 nodes and degree 4. ................................ 19
3.3 The Candy Graph illustrated. .................................................. 20
3.4 The construction of the Egg Graph. .......................................... 21
3.5 The Double Layered Egg Graph illustrated. ................................. 22

4.1 The convergence distribution of the 1-flipper operation on the Circle Graph a) and the Candy Graph b). The needed amount of operations have been sorted into bins $P(Ops)$. .............................................. 26
4.2 The three convergence phases demonstrated on the Circle Graph with 1800 nodes and degree 8. .............................................. 28
4.3 The evolution of the lower-bound on the Candy, Circle and the Egg Graph with degree 8 and 1800 nodes. .............................................. 28
4.4 Graph showing the evolution of the amount of operations needed by incrementation of the node count. .............................................. 29
4.5 Graph showing the increasing amount of operations with increasing amount of nodes on the Egg Graph. ................................. 30
4.6 Graph showing the amount of flipper operations needed for convergence with increasing amount of nodes on 3 different Torus Graphs. .............................................. 31
4.7 Evolution of the number of flipper operations needed for convergence with increasing degree on the Candy Graph and the Circle Graph. .............................................. 32
4.8 The operations needed for convergence on the Egg Graph with increasing degree. ......................................................... 33
4.9 Graphs showing the evolution of the amount of operations needed with increasing amount of nodes and degree. .............................................. 34
4.10 Graph showing the amount of failed flipper operations on some test graphs in the convergence process. .............................................. 35
4.11 Graph comparing the amount of triangles with the amount of failed operations in the Egg and the Torus Graph. ................................. 36
List of Figures

4.12 Comparison between the diameter and the lower-bound of the expansion ratio on the Egg and Torus Graph.

4.13 The lower-bound of the Egg Graph with 500 nodes and degree 4 over a period of $8 \cdot 10^6 + 006$ operations.

4.14 The lower-bound of the Egg Graph with 500 nodes and degree 9 over a period of $8 \cdot 10^6 + 006$ operations.

5.1 A Snapshot of the Gnutella Network