A PILOT STUDY OF MODULE INTERCONNECTEDNESS

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Prasanth Vanguru

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science
Statistics

October 2010

Fargo, North Dakota

North Dakota State University
Graduate School

Title

A Study of Modules Interconnectedness

By

Prasanth Vanguru

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

ABSTRACT

Vanguru, Prasanth, M.S., Department of Computer Science, Department of Statistics, College of Science and Mathematics, North Dakota State University, October 2010. A Pilot Study of Module Interconnectedness. Major Professors: Dr. Kenneth Magel, Dr. Rhonda Magel.

Complexity plays an important role in understanding and working with a program, and has been measured in many different ways for software applications. The use of statistical analysis is one of the ways to predict the pattern of complexity among the modules present in a software application. A random sample of twelve software applications was selected for this study to examine complexity. A single pair of complexity measures was evaluated. This pair of complexity measures was the in-degrees and out-degrees for each module of an application. The next step was to try to fit suitable statistical distributions to the in-degrees and to the out-degrees. By using various statistical distributions such as the normal, log-normal, exponential, geometric, uniform, poisson and the chi-square, we try to determine the type of distribution for the in-degrees and the type of distribution for out-degrees of the modules present in the software applications so that the pattern of complexity can be derived. The chi-square goodness of fit test was used to test various null hypotheses about the distributions for the in-degrees and for the out-degrees. Results showed that the pattern of in-degrees and the pattern of out-degrees both followed chi-square distributions.

ACKNOWLEDGEMENT

I would like to thank Dr. Kenneth Magel, Dr. Rhonda Magel for their continued support, help, and direction. My sincere thanks to Dr. Fu-Chih Cheng and Dr. Edward L.Deckard, for serving on the committee. I would like to thank my family and friends who supported and motivated me to accomplish this task.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

Table Page

Table | Page

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

When a computer application is constructed it is easily susceptible to change. In order to accommodate growing functions and changing requirements, modifying or updating an application tends to lead to a more complex system. Is it possible to predict the error rate in our code and the complexity that arises due to alteration of the code? Can the code be re-used in future applications? If we could make such predictions, we could reduce the code complexity and errors by avoiding changes that are more complex than required. When a software application is evaluated, error rate and re-usability are considered. A low error rate and high re-usability are desired. Software complexity should be tailored in such a way that software systems which use reusable components should have reduced complexity.

A software program is divided into smaller units called modules. Each module is a separate compilation unit. These modules vary in size and number for each program. When a program is executed, sequences of statements are executed and this execution of statements from different modules is known as control flow. Interaction among modules occurs when a module shares information with another module, which is known as data flow. Data flows usually includes parameters being passed from one module to another and return values and side effects coming back

The problem is trying to understand application complexity. If we understand complexity, we can predict the error rate, difficulty to change and reusability. Application complexity is more than module complexity. Module complexity depends on the following factors: the number and size of modules, the control flow; and the data flow. Inter-module interaction plays an important role in complexity. One factor which plays an important role in inter-module interaction is the number of inter-connections between the modules (i.e. the number of in-degrees and out-degrees between the

modules). The number of possible execution paths in the system can be extremely large and the resulting performance can be very complex.

Our objective is to find the type of statistical distribution the in-degrees and the type of statistical distribution out-degrees follow among the software modules within an application as an approximate measure of complexity. By trying to find a distribution among the modules present, we might be able to draw conclusions on the relationship of module distribution in programs and how it affects application complexity.

One of the approaches used in trying to determine the probability distributions is to first take a sample of the in-degrees and out-degrees of all the functions present in the software modules. This is done using the call graph method. A call graph is used to map the interactions of various components in a program. The out-degrees are defined to be the number of calls going out from a module to a function in a different module. The in-degrees are defined to be the number of calls being received by a module from a different module.

Example: In terms of modules, if a function in module A, accesses only one function in module B, and no other module accesses B, the out-degree for A is 1 and the accessed module B has in-degree 1. In this way, different functions in modules access functions in various modules due to which the in-degree and out-degree grows for each module. We determine the distribution of in-degrees and the distribution of out-degrees by measuring a sample of applications

CHAPTER 2: LITERATURE REVIEW

Software complexity plays an important role in the software development of a program. Software complexity is influenced by many attributes including the program construction, application size, branching complexity, module size and number of modules used to construct the program.

Phukan, Kalava & Prabhu (2005) attempted to show the importance and influence of inter-module complexity on a software application and how interactions between the modules are one of the reasons for increasing complexity of an application. Attempts were made to quantify complexity of Enterprise Resource Planning (ERP) software. The metrics used to calculate the complexity of the system were the inter-module interactions and the process complexity. Inter-module complexity of the application was calculated by taking into account the fan-in + fan-out, where fan-in is defined as the number of calls received by the module. The number of calls that emanate from a module is called as fan-out of the module. The other metric used was process complexity which is defined as intra-module complexity multiplied by square of inter-module complexity. These two metrics were evaluated for the sales and order process module of the ERP software. The complexity that was calculated from the process (121,296) exceeded the reference complexity which was in the range of 10,000 to 20,000.

In order to reduce the application complexity the Erlang (1986) software proposes the two rules: The first rule is to export minimum number of functions from a module. The complexity of a module depends upon the number of functions which are exported from the module. For example, a module which exports one or two functions is less complex and usually easier to understand than a module which exports dozens of functions. The second rule is to try to reduce inter-module dependencies. Reducing the

interdependencies between modules simplifies the problem of maintaining these modules. This approach has been followed especially for designing the telephone software which contained many modules.

Schneberger and McLean (2003) identified factors that make an information system complex and prescribed ways this complexity can be reduced. The components used for the construction of the application, different interactions, and changes to the software by the programmers are some of the factors responsible for the complexity of the system. They suggest the following three ways to reduce complexity in an application: lowering the component curve, lowering the system curve; and finally moderate the distribution of modules. Moderating the distribution of modules means to optimize the number of software modules, minimize the number of dependent relationships, and have more data available directly to other components centrally rather than depending on other modules. This last step will the developer time to suggest improvements to the application. A moderate distribution of modules in an application advocates that there should be a balance between the number of modules created, complexity of the system and also the costs associated with each module.

Schneberger (1997) discusses the effect of a distributed computing environment on the maintenance of a software application. Over 150 programmers, system analysts, and project leaders in four large organizations were interviewed and surveyed. All these organizations had centralized and distributed systems. One of the important conclusions of this research was that some computing complexity factors appear to have greater effects than others. In particular, the variety of components and interactions between them has a greater effect on the overall complexity of the system. The other factor taken into consideration was the rate at which the system changes. Rapid change tends to increase interactions between components which caused ripple effects throughout the

system thus increasing the overall complexity of the system. The results showed that software for distributed systems seems harder to maintain than for centralized systems. The greater the complexity factor, the greater the difficulty maintaining software.

Lungu and Lanza (2007) used visualization as a technique to observe the inter-module relationships in large evolving systems. In order to use the visualization technique, a semantic dependency matrix is used. This matrix takes into account the dependencies; function calls from one module to another and arranges them in a hierarchical order starting from the class which has the least dependencies. Patterns of relationships between the modules were studied and were categorized into relationships, namely life-term, fossil, stable and unstable. Life-term explains the relation that had existed between the first and final versions of the application. Fossil relations are relations among modules that are not present in the current systems version and could have happened due to new relationships that have taken place in order to increase the functions of a system. Stable relations occur when there are no new dependencies among modules when the versions of the system change. Unstable relations are those which change qualitatively and quantitatively during the system evolution, giving rise to new inter-dependencies among modules. Having unstable relations implies that the system needs changes to the design to bring it to a stable state.

The results showed that the azereus which has been through five versions has been stable and has not changed much in its evolution. The argo UML assessed in this paper reported unstable relations between the modules. This method of observing the inter-module relationships in software applications helps in the recovering structure information and also for proposing a pattern language for inter-module relationships.

Ma, He & Du (2005) proposed a qualitative measure based on the "structure entropy that measures the amount of uncertainty of structural information and on the

linking weight that measures the influences of interactions or relationships between components of software systems". This measure takes into account the uncertainty of a structure in an application, relationships and interactions between components (classes, sub-classes) which form the base for structural complexity. The relationships between modules have been represented by the call graph method, namely the in-degrees and out-degrees. In-degrees represent incoming calls to the node and out-degrees represent out-going calls from nodes. The measures which were used are: connectivity, ripple degree and abstraction degree. Connectivity is the sum of in-degrees and out-degrees of a node. Ripple degree measures the influence of node on other nodes. The whole graph and abstraction degree is calculated by the number of interactions or relationships between the components. Weight of an edge is defined by the significance of its interaction with other components. The interactions considered among components ranked in increasing order are the following: procedure call; streaming data access; and linkage.

Ma et al. (2005) calculated structural complexity by using the connectivity, ripple degree and abstraction degree of the components. To measure the structural complexity different types of sample systems were selected. Systems which were used are lattice, subsystems, aime, yahoopops, blender, gtk, xmms, jdk-b, cs,striker ,linux, star. It was observed that star network which had many interactions showed that as information between the components increases, the structural complexity increases.

Khimta, Sandhu & Brar (2008) attempted to find complexity measure for Java beans software components in order to make the software components more reusable. A component consists of classes, derived classes, base classes etc. Based on these attributes Khitma et al. (2008) proposed a metric. The proposed metric takes in to account the following measures in order to calculate the complexity among

components: component complexity, interface complexity, coupling factor and cyclometric complexity.

Khitma et al. (2008) defines component complexity as "Component complexity takes into account total number of the simple, medium, complex variables in a component and also the weight of the variables which consist of integers, char, double, string, date, link list stack and queue which is given by the following equation:

$$V(Component_x) = \sum_{i=1}^{n1} W_{simlpei} + \sum_{j=1}^{n2} W_{meidumj} + \sum_{k=1}^{n3} W_{complexk}$$

where n1, n2, n3 are the total number of simple, medium and complex variables in the components and $w_{simplei}$, $w_{mediumj}$, $w_{complexk}$ are the weight value of simple, medium, complex variables". "The interface complexity component is one which requests information from another component is given by:

$I(Component_x) = \sum_{i=1}^{m1} W_{simlpei} + \sum_{j=1}^{m2} W_{mediumj} + \sum_{k=1}^{m3} W_{complexk}$ where m1, m2, m3 are the total number of interface methods of simple, medium and complex $w_{simplei}$, $w_{mediumj}$, $w_{complexk}$ are the weight value of simple, medium, complex nature of interface methods". Coupling factor is given by the equation C ($component_x$) =o/d "Where *O* is total number of other methods being called in the methods of the Componentx and *D* is the total number of declared methods in the Componentx. Cyclometric complexity (component) measures the number of paths taken by a component in a program represented by CC (Component). The complexity is measured as the sum of all the above methods. These metrics were tested for twenty JavaBeans components collected from open repositories. The rate of component customizability is also calculated for all the components of JavaBeans defined by Washizaki (2003). A correlation was carried out between complexity and the rate of component customizability using Karl Pearson

co-efficient of correlation which indicated that high complexity between the components leads to high maintenance and low customizability. To reduce this complexity the components need to be reviewed i.e. their structure, interactions and their dependency.

Component based development is one of the practices that is gaining popularity among software developers. Gill & Balkishan (2008) presents metrics which help in controlling the complexity among the components. One of the metrics proposed is component dependency metric (CDM) that measures the interactions and the number of dependencies among components. CDM measures dependencies in two ways, for example, if a component is directly connected to another component it takes into measure the single path to reach the component otherwise takes into account all the paths required to connect with the other component. The value of CDM was estimated to be between 0 and 1 indicating that 0 being minimum dependency and 1 representing maximum dependency. A second metric presented by Gill and Balkishan (2008) is the component interaction density metric (CIDM) measures the dependency, coupling aspects of software components and interactions between the components. It was calculated by taking the ratio of interactions between the components divided by total number of components. Values above 1 indicate high dependencies among components and thus making the structure more complex.

These metrics were applied to four cases: i) first application had 5 components and 5 interactions, ii) the second had 5 interactions and 8 components and lot more interactions between the components; iii) the third case had 7 components and 9 interactions; and the fourth had 7 components and 6 interactions. The results showed that the higher values of CIDM indicated higher dependencies between the components

and dependency oriented complexity showed that if the interactions between the components is less the application complexity is lower.

Sharma, Kumar & Grover (2007) proposed a complexity metric based on different constituents of the components like inheritance of classes, methods and attributes. This metric is known as component complexity. Classes in the component are derived into base-classes and derived classes. Base classes are imported from other reused library or packages. Derived classes are identified classes during component design in a domain. Methods are categorized on the basis of their arguments and return types. Complexity is calculated by taking into account the following attributes namely: variables which contribute to the complexity are denoted by (Cv), number of methods in a component which contribute to complexity are denoted by (Cm) and the interface complexity between the components is denoted by (Ci). This metric was applied to java bean components. Results were obtained for each of the components. The results have been validated by another metric called rate of component customizability (RCC) defined by Washizaki (2003). A correlation analysis was conducted for the complexity and rate of customizability using the Karl Pearson coefficient of correlation indicating high complexity among components leads to high maintainability.

Haesool, Tsujino & Tokura (1991) evaluated the conventional measures of complexity which measure complexity based on program size (Halstead 1977) program control structure (McCabe 1976) data structure and data flow of the program (Henry 1981) (Chapine 1979) against the proposed measures of complexity which are module complexity metrics. Module Complexity metrics consists of inter-module complexity and interface complexity. A sample of 30 programs was evaluated for complexity. The correlations among the proposed module complexity metrics were high implying that module complexity metrics represented the module dependencies and interactions in a

comprehensive way where as the conventional measures of complexity could not represent the interactions or the dependencies in a significant way.

Following are some of the measures which are used to measure complexity: Cyclomatic complexity is a software metric measurement developed by McCabe (1976) and is used to indicate the complexity of a program. It measures the number of linearly independent paths through a program's source code and thereby placing a numerical value on the complexity. In practice it is a count of the number of test conditions in a program. It can also be applied to individual functions, modules, methods or classes within a program. Cyclomatic complexity measuring allows us to evaluate the quality of the program code and detect high-complexity procedures. The McCabe metric is: m = e - n + x where: m is the McCabe Cyclomatic Complexity (MCC) metric, e is the number of edges in the control flow graph of the program, n is the number of nodes or decision points in the graph of the program and x is the number of connected components. MCC also is useful in determining the testability of a program. The higher the value, the more difficult and risky the program is to test and maintain. An example would be a Java method. The McCabe's Cyclomatic complexity would be applied in the following way, start with a count of one for the method. Add one for each of the following flow-related elements that are found in the method: Methods: Each return that isn't the last statement of a method, Selection: if, else, case, default, Loops: for, while, do-while, break, and continue, Exceptions: catch, finally, throw, or throws clause, Threads: start () call on a thread.

McCabe suggests that complexity under 5 is good, from 5-10 is ok, and over 10 is too complex. The advantages that are associated with this metric are that it can be calculated earlier in the life cycle of an application. It also estimates the minimum effort

and the best areas that the programmer needs to concentrate for testing. It is easy to apply to an application to get the complexity.

Henry and Kafura (1981) introduced the software metrics based on information flow in an application which measures complexity as a function of fan-in and fan-out. The fan-in of a procedure is defined as the number of calls into that procedure including the number of data structures from which that procedure retrieves information. Fan-out is defined as the number of calls out of that procedure plus the number of data structures that the procedure updates. The data flow from components to components is taken into account in this method and a count is maintained for the number of times data flow occurs between components. Typically some of the attributes that are taken into account are field declarations, formal parameters, return types, and local variables. The formula given is Complexity = Length x (fan-in x fan-out) ^{2}. This metric was validated using an entire UNIX operating system. Faulty components were identified.

The Halstead (1977) complexity tries to estimate the programming effort needed to develop an application. Some of the properties this metric uses to calculate complexity are the number of distinct operators, operands and the total number of operands and operators. Using these properties the program length, difficulty and effort are calculated. The mathematical formulas for the above attributes are as follows:

"From these numbers, five measures can be calculated: Program length: N = N1+N2, Program vocabulary: n = n1+n2, Volume: V = n x $\log_2$ n, Difficulty: D = $\frac{n1}{2}$ x$\frac{N2}{n2}$, Effort: E = D x V, where n_1= number of unique or distinct operators appearing in that implementation, n_2= number of unique or distinct operands appearing in that implementation, N_1= total usage of all of the operators appearing in that implementation, N_2= total usage of all of the operands appearing in that implementation (Halstead, 1977)".

CHAPTER 3: SAMPLING

This chapter explains the data collection and the tests performed to evaluate the probability distributions as predictions of module in-degrees and out-degrees. As the interactions between the modules in a software application increase, the complexity of the application increases. Our goal is to predict the pattern of the complexity based on the modules that an application has. We use the in-degrees and out-degrees of each module as a prediction of complexity of that module.

To help determine the types of distributions the in-degrees and out-degrees of a module follow, the in-degrees and out-degrees were collected from twelve java applications which were all open-source applications from the website www.sourceforge.net (1999) consisting of graphical packages and interactive games. In this study, we used the lines of code in the modules of each application as the metric to find the pattern of the distributions exhibited by the in-degrees and out-degrees of a module so as to predict complexity.

Our applications had an average of five to thirty modules each. For each of the applications, the in-degrees and out-degrees were found by using the eclipse software (2007). The out-degree is the number of modules called by the module. The in-degrees is the number of calls made to the module.

The applications which were used are;

1. Sudoku-"Marc's sudoku is an interactive computer game program with an advanced puzzle creation and solving engine. With Marc's Sudoku you can play alone or in duel/battle". The goal of a Sudoku problem is to complete the published grid so that no number appears more than once in any row, column or 3x3 sub-grids. This application has 8 modules. Table 1 gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net. 1999).

Table 1. In-degrees and out-degrees of Sudoku application

Modules	in-degree	out-degree
1	0	1
2	1	2
3	1	0
4	4	1
5	2	2
6	2	2
7	3	3
8	3	4

2. Snort alert monitor (SAM) - "SAM is a near real-time Snort alert monitor and is a program to monitor network security. SAM provides many ways to indicate that you may be experiencing an intrusion attempt on your network including audio/visual warnings, email warnings". This Tool provides reporting and real-time statistics .This application has 8 modules. Table 2 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

Table 2. In-degrees and out-degrees of Snort alert monitor application

Modules	in-degree	out-degree
1	0	0
2	2	0
3	2	0
4	2	3
5	2	3
6	3	4
7	4	5
8	5	5

3. Java game maker-"Java game maker is a simple tool that allows the user to make java games writing few lines of code". Some of the features are, it can run on Windows, Mac and Linux and supports images with color. It has an internal game editor which is extendible and has a sub-engine that speeds up performances without drawing. This application has 33 modules. Table 3 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

Table 3. In-degrees and out-degrees of Java game maker application

Modules	in-degree	out-degree
1	0	0
2	1	0
3	1	0
4	1	0
5	1	2
6	1	2
7	1	2
8	1	2
9	1	2
10	4	1
11	4	1
12	4	1
13	4	1
14	4	1
15	4	2
16	2	2
17	2	2
18	2	2
19	2	2
20	2	2
21	2	2
22	2	3
23	2	3
24	2	3
25	2	3
26	3	3
27	3	3
28	4	4
29	4	4
30	5	4
31	6	5
32	6	7
33	6	10

4. Jippy-snake-"This is an application written in java and an interactive game". It runs on Windows, Mac and Linux and supports images with color. It has an internal game editor which is extendible. This application has 21 modules. Table 4 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

Table 4. In-degrees and out-degrees of Jippy-snake application

Modules	in-degree	out-degree
1	0	0
2	1	0
3	1	1
4	1	1
5	1	1
6	1	1
7	1	1
8	1	1
9	2	1
10	2	1
11	2	2
12	2	2
13	2	2
14	2	2
15	3	3
16	3	3
17	3	4
18	4	5
19	4	6
20	6	6
21	8	7

5. Pocket-basket-"Podcast client supporting pocket pc and smartphone". This is a standalone basket ball game designed for devices such as cellular phones and handheld devices. The objective of this game is to shoot as many baskets as possible while preventing your opponent shoot to your basket. This application has 9 modules. Table 5 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

Table 5. In-degrees and out-degrees of Pocket-basket application

Modules	in-degree	out-degree
1	0	1
2	1	1
3	1	1
4	1	1
5	1	1
6	1	1
7	1	1
8	2	1
9	8	8

6. Lines &dot-"Computer version of a simple but addictive, single-player pen &paper game". The game is simple and learning to play it takes only a few minutes, but it can still offer entertainment for a long time. This application has 18 modules. Table 6 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

Table 6. In-degrees and out-degrees of Lines &dot application

Modules	in-degree	out-degree
1	0	0
2	1	0
3	1	0
4	1	0
5	1	0
6	1	0
7	1	0
8	1	1
9	1	1
10	1	1
11	2	2
12	2	2
13	2	2
14	2	3
15	2	3
16	3	3
17	3	4
18	4	7

7. Thief-script-"The ultimate nuke zone Thief-script. It helps you pick the right thief target in the online game 'nuke zone'. The program is written in java to ensure cross-platform compatibility". This application has 27 modules. Table 7 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

8. J-controller-"An open source framework for building java web applications based on the model-view-controller (MVC) design paradigm". This application has 29 modules. Table 8 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

Table 7. In-degrees and out-degrees of Thief-script application

Modules	in-degree	out-degree
1	0	0
2	1	0
3	1	0
4	1	0
5	1	1
6	1	1
7	1	1
8	1	1
9	1	1
10	1	1
11	1	1
12	1	1
13	1	1
14	2	1
15	2	2
16	2	2
17	2	2
18	3	2
19	3	2
20	3	3
21	3	3
22	3	3
23	4	3
24	5	4
25	4	5
26	5	6
27	5	11

8. J-controller-"An open source framework for building java web applications based on the model-view-controller (MVC) design paradigm". This application has 29 modules. Table 8 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

9. Evver-games-"An Evver game is the open source project used to power the gaming site, evver.com. This project contains an ajax web renderer and framework for creating card games; along with the card games golf and no peeking". This application has 7 modules. Evver games also features several games written with the framework and

playable online at evver.com Table 9 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

Table 8. In-degrees and out-degrees of J-controller application

Modules	in-degree	out-degree
1	0	0
2	1	0
3	1	0
4	1	0
5	1	2
6	1	2
7	1	2
8	1	2
9	1	1
10	1	1
11	1	1
12	2	1
13	2	1
14	2	1
15	2	2
16	2	2
17	2	2
18	2	2
19	2	3
20	3	3
21	3	3
22	3	3
23	3	3
24	3	3
25	4	3
26	4	4
27	5	6
28	5	7
29	5	12

Table 9. In-degrees and out-degrees of Evver-games application

Modules	in-degree	out-degree
1	0	0
2	0	0
3	1	0
4	1	0
5	1	1
6	1	1
7	1	2

10. Free-cell-"A Java / J2ME implementation of the popular free-cell card game and the popular sudoku game. It supports MIDP enabled mobile phones, and plan to expand and implement other games for mobile phones in the future". This application has 9 modules. Table 10 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

Table 10. In-degrees and out-degrees of Free-cell application

Modules	in-degree	out-degree
1	0	0
2	1	0
3	1	0
4	2	0
5	2	2
6	2	2
7	2	2
8	2	4
9	3	5

11. Minesweeper-"J2Minesweeper is minesweeper game can be played at portable media (cell phone, pda, etc) with J2ME supported". This application has 17 modules. Table 11 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

12. Black-jack analyst.-"This is a gui version of the game blackjack, written in python and pygame. It is easy to play, and can be very fast-Pac". This application has 21 modules. Table 12 gives the gives the number of in-degrees and out-degrees for each module. (www.sourceforge.net, 1999).

We would like to determine the distributions that the number of in-degrees and the number of out-degrees in a module follow. This will be done by testing the null hypothesis that the number of in-degrees follows a specific distribution. A null hypothesis is also tested for the number of out-degrees. If the null hypotheses are not

rejected, there is not enough evidence to indicate they do not follow the distributions. This does not prove that they do, but it is the best we can do.

Table 11. In-degrees and out-degrees of Minesweeper application

Modules	in-degree	out-degree
1	1	0
2	1	0
3	1	0
4	1	0
5	2	0
6	2	1
7	2	1
8	2	1
9	2	2
10	2	2
11	2	2
12	3	3
13	3	3
14	3	4
15	3	5
16	4	5
17	7	12

Table 12. In-degrees and out-degrees of Black-jack analyst application

Modules	in-degree	out-degree
1	0	0
2	1	0
3	1	0
4	1	0
5	1	1
6	1	1
7	1	1
8	4	2
9	4	2
10	4	2
11	2	2
12	2	2
13	2	3
14	2	3
15	3	4
16	3	4
17	3	4
18	3	4
19	3	4
20	3	4
21	4	4

The following null hypotheses were tested for the in-degrees:

H_0: The distribution of in-degrees follows normal distribution.

H_0: The distribution of standardized in-degrees for each module follows standard-normal distribution.

H_0: The distribution of standardized in-degrees for each module follows standard-normal distribution

H_0: The distribution of in-degrees follows log- normal distribution.

H_0: The distribution of standardized in-degrees for each module follows standard log-normal distribution.

H_0: The distribution of in-degrees follows exponential distribution.

H_0: The distribution of standardized in-degrees for each module follows standard-exponential distribution.

H_0: The distribution of in-degrees follows geometric distribution

H_0: The distribution of in-degrees follows a discrete uniform distribution.

H_0: The distribution of in-degrees follows poisson distribution.

H_0: The distribution of in-degrees follows chi-square distribution.

The following null hypotheses were tested for the out-degrees:

H_0: The distribution of out-degrees follows normal distribution.

H_0: The distribution of standardized out-degrees for each module follows standard-normal distribution

H_0: The distribution of out-degrees follows log- normal distribution.

H_0: The distribution of standardized out-degrees for each module follows standard log-normal distribution

H_0: The distribution of out-degrees follows exponential distribution.

H_0: The distribution of standardized out-degrees for each module follows standard-exponential distribution.

H_0: The distribution of out-degrees follows geometric distribution

H_0: The distribution of out-degrees follows a discrete uniform distribution.

H_0: The distribution of out-degrees follows poisson distribution.

H_0: The distribution of out-degrees follows chi-square distribution.

Note that the type of distribution was specified in each case but not the parameter values. We would expect the parameter values to change depending on the size of module. Some of the distributions we could assume they follow could be the normal, log-normal, exponential, geometric, uniform, poisson and the chi-square. This helps us in categorizing the data we have got. These distributions types were used for the test because the normal distribution is commonly found in many applications, the log-normal and exponential deal with positive values and are non-symmetric distributions as was anticipated in these cases. The uniform distribution is one in which the probability of occurrence is same for all the intervals. Graphs of the shapes of each of the distributions are shown:

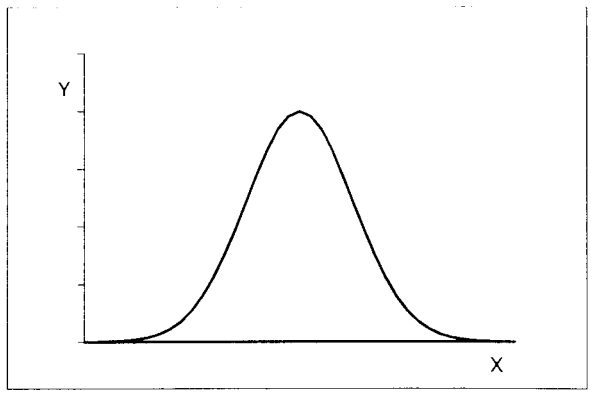

Figure 1. Graphical representation of the normal distribution curve

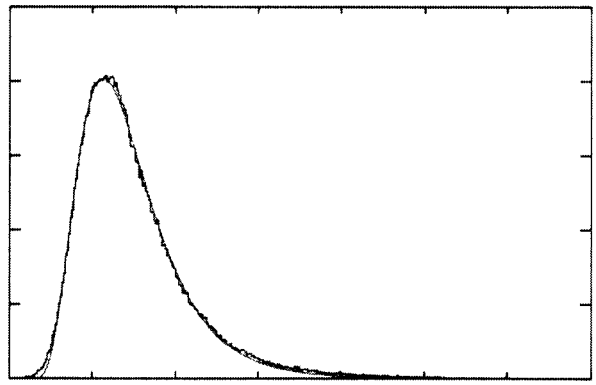

Figure 2. Graphical representation of the log-normal distribution curve

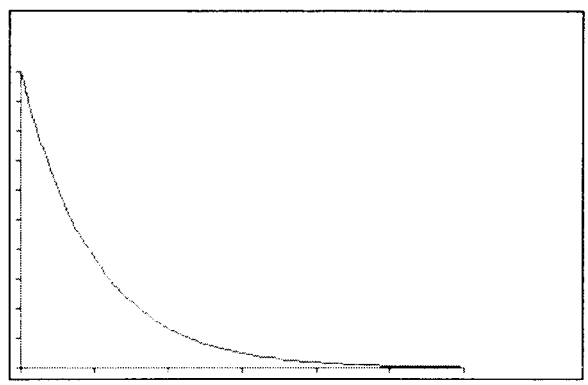

Figure 3. Graphical representation of the exponential distribution curve

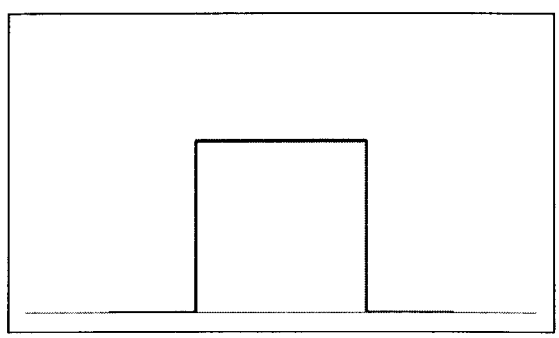

Figure 4. Graphical representation of the uniform distribution curve

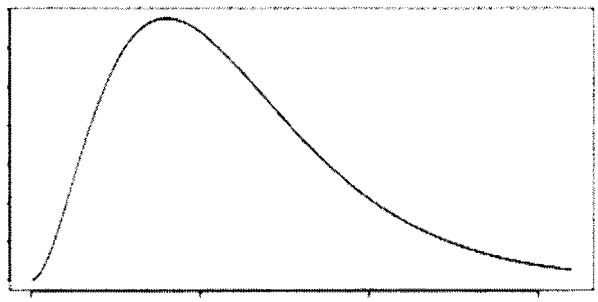

Figure 5. Graphical representation of the chi-square distribution curve

The chi-square goodness of fit test was used since it is a known test that could be used to test the null hypotheses that the number of in-degrees or the number of out-degrees follows a specified distribution. The chi-square goodness of fit test can be used for both continuous and discrete distributions and it has a correction procedure when the parameters are unknown.

Results are given in Chapter 4.

CHAPTER 4: RESULTS

In this chapter we wish to determine the type of distribution that the in-degrees has, and the type of distribution the out-degree has. It is not possible to prove that the in-degrees, or the out-degrees has a given a distribution. It is only possible to show that there is not enough evidence to reject the hypothesis that the in-degrees, or the out-degrees, have a given distribution. Namely, the null hypothesis is always going to be that the in-degrees has a specified distribution and the alternate hypothesis is going to be that it does not. The same is true for testing the out-degrees distribution. We will begin by testing for the following distributions for the in-degrees: normal, standard-normal, log-normal, exponential, standard-exponential, geometric, uniform, poisson and the chi-square. We will then test for the same distributions using the out-degrees.

We are only assuming that the in-degrees and out-degrees follow one of the above distributions. We will then conduct statistical tests to determine if there is evidence they don't follow a given distribution. We are assuming that the parameters change (mean and variance) depending on the size of the program. We are just testing for the type of distribution, not with specific parameters. A chi-square goodness of fit test will be used in all cases. In using a chi-square goodness of fit test, distinct categories for the range of values of the observations should be decided on ahead of time.

In order to use the chi-square goodness of fit test, the data must first be divided up into C categories. If the null hypothesis is true, the chi-square statistic will have an asymptotic chi-square distribution with C-1 degrees of freedom. If the parameters must be estimated, degrees of freedom will also be subtracted for the number of estimated parameters. In Chapter 3, we described how the data was collected. There were twelve applications with the number of modules in the range of 5- 30 for which the number of

in-degrees and the number of out-degrees were taken. The type of distribution for the in-degrees should be the same regardless of the application size, but the parameters will change depending on the size of the application. To begin with, we tested to see if the in-degrees followed the normal distribution. The sample mean and the standard deviation were calculated for the number of in-degrees (for a module) for each application. The data were then standardized based on their program size. A chi-square goodness of fit test was performed on the standardized data, testing whether the data followed a standard normal distribution.

To get the standardized value, the following was the method used. The mean and standard deviation were calculated for the in-degrees, and then for the out-degrees of each module of an application. The number of in-degrees was subtracted from mean and divided by the standard deviation to get the standardized value for each in-degree observation. Standardized values for out-degree observations were found in the same way. The standardized values for the out-degrees and in-degrees are given in Tables 13-36.

The null hypothesis that the distribution of in-degrees and the distribution of out-degrees have a log-normal distribution was also tested. In order to test for this distribution, the natural logarithm of each of the values is taken, the values are standardized and the null hypothesis that these transformed values follow a standard normal distribution is tested. The log-normal value of 0 was taken as 0 for the purpose of calculations. The mean and standard deviation was calculated for the transformed log-values. To get the log-normalized value, the transformed in-degrees and out-degrees of each application were subtracted from the mean and divided by the standard deviation. The mean, standard deviation of the log values and the standardized log-

normal value for the out-degrees and the mean, standard deviation of the log values and the standardized log-normal value for the out-degrees 13-36.

Following are the standardized values for out-degrees.

Table 13. Standardized out-degree values for Sudoku application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-1.50430417	0	-1.07659009
2	2	0.100286945	0.30102	0.231347612
3	1	-0.702008612	0	-1.07659009
4	1	-0.702008612	0	-1.07659009
5	2	0.100286945	0.30102	0.231347612
6	2	0.100286945	0.30102	0.231347612
7	3	0.902582502	0.47712	0.996442121
8	4	1.704878059	0.6020	1.539285314
Mean	1.875		0.24778	
Standard Deviation	1.246423455		0.23015	

Table 14. Standardized out-degree values for Snort alert monitor application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-1.13435651	0	-1.164494141
2	0	-1.13435651	0	-1.164494141
3	0	-1.13435651	0	-1.164494141
4	3	0.226871303	.47712	0.340067259
5	3	0.226871303	.47712	0.340067259
6	4	0.68061391	.60205	0.734050966
7	5	1.134356516	0.69897	1.03964847
8	0	-1.13435651	0	-1.164494141
9	0	-1.13435651	0	-1.164494141
10	0	-1.13435651	0	-1.164494141
11	5	1.134356516	.69897	1.03964847
Mean	2.5		.3692	
Standard Deviation	2.20389266		.3171	

Table 15. Standardized out-degree values for Java game maker application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-1.216944863	0	-1.253284679
2	0	-1.216944863	0	-1.253284679
3	0	-1.216944863	0	-1.253284679
4	0	-1.216944863	0	-1.253284679
5	2	-0.22536016	.3010	-0.103194015
6	2	-0.22536016	.3010	-0.103194015
7	2	-0.22536016	.3010	-0.103194015
8	2	-0.22536016	.3010	-0.103194015
9	2	-0.22536016	.3010	-0.103194015
10	1	-0.721152511	0	-1.253284679
11	1	-0.721152511	0	-1.253284679
12	1	-0.721152511	0	-1.253284679
13	1	-0.721152511	0	-1.253284679
14	1	-0.721152511	0	-1.253284679
15	2	-0.22536016	.3010	-0.103194015
16	2	-0.22536016	.3010	-0.103194015
17	2	-0.22536016	.3010	-0.103194015
18	2	-0.22536016	.3010	-0.103194015
19	2	-0.22536016	.3010	-0.103194015
20	2	-0.22536016	.3010	-0.103194015
21	2	-0.22536016	.3010	-0.103194015
22	3	0.270432192	.47712	0.569565896
23	3	0.270432192	.47712	0.569565896
24	3	0.270432192	.47712	0.569565896
25	3	0.270432192	.47712	0.569565896
26	3	0.270432192	.47712	0.569565896
27	3	0.270432192	.47712	0.569565896
28	4	0.766224543	.60205	1.046896649
29	4	0.766224543	.60205	1.046896649
30	4	0.766224543	.60205	1.046896649
31	5	1.262016895	.68897	1.417143145
32	7	2.253601598	.84509	1.975428007
33	10	3.740978653	1	2.567233809
34	1	-0.721152511	0	-1.253284679
35	1	-0.721152511	0	-1.253284679
36	2	-0.22536016	.3010	-0.103194015
Mean	2.454545455		.3280	
Standard Deviation	2.01697343		.0261	

Table 16. Standardized out-degree values for Jippy-snake application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-1.155255194	0	-0.891876785
2	0	-1.155255194	0	-0.891876785
3	1	-0.670048013	0	-0.891876785
4	1	-0.670048013	0	-0.891876785
5	1	-0.670048013	0	-0.891876785
6	1	-0.670048013	0	-0.891876785
7	1	-0.670048013	0	-0.891876785
8	1	-0.670048013	0	-0.891876785
9	1	-0.670048013	0	-0.891876785
10	1	-0.670048013	0	-0.891876785
11	2	-0.184840831	.3010	0.070128686
12	2	-0.184840831	.3010	0.070128686
13	2	-0.184840831	.3010	0.070128686
14	2	-0.184840831	.3010	0.070128686
15	3	0.30036635	.47712	0.632865812
16	3	0.30036635	.47712	0.632865812
17	4	0.785573532	.60205	1.032134157
18	5	1.270780713	.69897	1.341830746
19	6	1.755987895	.7781	1.594871283
20	6	1.755987895	.7781	1.594871283
21	7	2.241195076	.8450	1.808814009
Mean	2.380952381		.2790	
Standard Deviation	2.060975266		.3129	

Table 17. Standardized out-degree values for Pocket-basket application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	1	-0.333333333	0	-0.333333333
2	1	-0.333333333	0	-0.333333333
3	1	-0.333333333	0	-0.333333333
4	1	-0.333333333	0	-0.333333333
5	1	-0.333333333	0	-0.333333333
6	1	-0.333333333	0	-0.333333333
7	1	-0.333333333	0	-0.333333333
8	1	-0.333333333	0	-0.333333333
9	8	2.666666667	.9030	2.666666667
Mean	1.777777778		.100	
Standard Deviation	2.333333333		.3010	

Table 18. Standardized out-degree values for Lines &dot application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-0.855601105	0	-0.778838422
2	0	-0.855601105	0	-0.778838422
3	0	-0.855601105	0	-0.778838422
4	0	-0.855601105	0	-0.778838422
5	0	-0.855601105	0	-0.778838422
6	0	-0.855601105	0	-0.778838422
7	0	-0.855601105	0	-0.778838422
8	1	-0.32453835	0	-0.778838422
9	1	-0.32453835	0	-0.778838422
10	1	-0.32453835	0	-0.778838422
11	2	0.206524405	.3010	0.337132049
12	2	0.206524405	.3010	0.337132049
13	2	0.206524405	.3010	0.337132049
14	3	0.73758716	.47712	0.989932927
15	3	0.73758716	.47712	0.989932927
16	3	0.73758716	.47712	0.989932927
17	4	1.268649915	.60205	1.45310252
18	7	2.86183818	.8450	2.354086773
Mean	1.611111111		.2100	
Standard Deviation	1.883016631		.269	

Table 19. Standardized out-degree values for Evver-games application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-0.846114112	0	-0.540061725
2	0	-0.846114112	0	-0.540061725
3	0	-0.846114112	0	-0.540061725
4	0	-0.846114112	0	-0.540061725
5	1	0.282038037	0	-0.540061725
6	1	0.282038037	0	-0.540061725
7	2	1.410190187	.3010	1.620185175
8	2	1.410190187	.3010	1.620185175
Mean	0.75		.0752	
Standard Deviation	0.88640526		.1393	

Table 20. Standardized out-degree values for Thief-script application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-0.9277009	0	-0.809569411
2	0	-0.9277009	0	-0.809569411
3	0	-0.9277009	0	-0.809569411
4	0	-0.9277009	0	-0.809569411
5	1	-0.495840136	0	-0.809569411
6	1	-0.495840136	0	-0.809569411
7	1	-0.495840136	0	-0.809569411
8	1	-0.495840136	0	-0.809569411
9	1	-0.495840136	0	-0.809569411
10	1	-0.495840136	0	-0.809569411
11	1	-0.495840136	0	-0.809569411
12	1	-0.495840136	0	-0.809569411
13	1	-0.495840136	0	-0.809569411
14	1	-0.495840136	0	-0.809569411
15	2	-0.063979372	.3010	0.197442506
16	2	-0.063979372	.3010	0.197442506
17	2	-0.063979372	.3010	0.197442506
18	2	-0.063979372	.3010	0.197442506
19	2	-0.063979372	.3010	0.197442506
20	3	0.367881392	.47712	0.786506715
21	3	0.367881392	.47712	0.786506715
22	3	0.367881392	.47712	0.786506715
23	3	0.367881392	.47712	0.786506715
24	4	0.799742155	.60205	1.204454423
25	5	1.231602919	.09897	1.528639851
26	6	1.663463683	.7781	1.793518632
27	11	3.822767503	1.0413	2.674119455
Mean	2.148148148		.2420	
Standard Deviation	2.315561133		.2989	

Table 21. Standardized in-degree values for Evver-games application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	0	-1.36533162	0	-0.35355339
2	0	-1.36533162	0	-0.35355339
3	1	0.195047374	0	-0.35355339
4	1	0.195047374	0	-0.35355339
5	1	0.195047374	0	-0.35355339
6	1	0.195047374	0	-0.35355339
7	1	0.195047374	0	-0.35355339
Mean	0.875		.0376	
Standard Deviation	0.640869944		.1064	

Table 22. Standardized out-degree values for J-controller application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-1.015878207	0	-1.079932248
2	0	-1.015878207	0	-1.079932248
3	0	-1.015878207	0	-1.079932248
4	0	-1.015878207	0	-1.079932248
5	2	-0.197531874	.3010	-0.038497789
6	2	-0.197531874	.3010	-0.038497789
7	2	-0.197531874	.3010	-0.038497789
8	2	-0.197531874	.3010	-0.038497789
9	1	-0.60670504	0	-1.079932248
10	1	-0.60670504	0	-1.079932248
11	1	-0.60670504	0	-1.079932248
12	1	-0.60670504	0	-1.079932248
13	1	-0.60670504	0	-1.079932248
14	1	-0.60670504	0	-1.079932248
15	2	-0.197531874	.3010	-0.038497789
16	2	-0.197531874	.3010	-0.038497789
17	2	-0.197531874	.3010	-0.038497789
18	2	-0.197531874	.3010	-0.038497789
19	3	0.211641293	.47712	0.570702316
20	3	0.211641293	.47712	0.570702316
21	3	0.211641293	.47712	0.570702316
22	3	0.211641293	.47712	0.570702316
23	3	0.211641293	.47712	0.570702316
24	3	0.211641293	.47712	0.570702316
25	3	0.211641293	.47712	0.570702316
26	4	0.62081446	.60205	1.002936669
27	6	1.439160794	.7781	1.612136775
28	7	1.84833396	.84509	1.843743906
29	0	-1.015878207	0	-1.079932248
30	0	-1.015878207	0	-1.079932248
31	0	-1.015878207	0	-1.079932248
32	0	-1.015878207	0	-1.079932248
33	0	-1.015878207	0	-1.079932248
34	1	-0.60670504	0	-1.079932248
35	1	-0.60670504	0	-1.079932248
36	1	-0.60670504	0	-1.079932248
37	1	-0.60670504	0	-1.079932248
Mean	2.482758621		.3121	
Standard Deviation	2.443953028		.2890	

Table 23. Standardized out-degree values for Free-cell application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-0.890870806	0	-0.907124126
2	0	-0.890870806	0	-0.907124126
3	0	-0.890870806	0	-0.907124126
4	0	-0.890870806	0	-0.907124126
5	2	0.178174161	.3010	0.207898724
6	2	0.178174161	.3010	0.207898724
7	2	0.178174161	.3010	0.207898724
8	4	1.247219129	.60205	1.322921574
9	5	1.781741613	.69897	1.681878756
Mean	1.666666667		.244	
Standard Deviation	1.870828693		.2699	

Table 24. Standardized out-degree values for Minesweeper application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-0.80359337	0	-0.866262562
2	0	-0.80359337	0	-0.866262562
3	0	-0.80359337	0	-0.866262562
4	0	-0.80359337	0	-0.866262562
5	0	-0.80359337	0	-0.866262562
6	1	-0.470396119	0	-0.866262562
7	1	-0.470396119	0	-0.866262562
8	1	-0.470396119	0	-0.866262562
9	2	-0.137198868	.3010	0.031761325
10	2	-0.137198868	.3010	0.031761325
11	2	-0.137198868	.3010	0.031761325
12	3	0.195998383	.47712	0.557071624
13	3	0.195998383	.47712	0.557071624
14	4	0.529195634	.60205	0.929785212
15	5	0.862392885	.69897	1.218884331
16	5	0.862392885	.69897	1.218884331
17	12	3.194773643	1.07918	2.353119397
Mean	2.411764706		.2903	
Standard Deviation	3.00122524		.3352	

Table 25. Standardized out-degree values for Black-jack analyst application.

Module	out-degree	Standardized normal value	log-values of the out-degree	Standardized log-normal value
1	0	-1.404878717	0	-1.365572509
2	0	-1.404878717	0	-1.365572509
3	0	-1.404878717	0	-1.365572509
4	0	-1.404878717	0	-1.365572509
5	1	-0.936585812	0	-1.365572509
6	1	-0.936585812	0	-1.365572509
7	1	-0.936585812	0	-1.365572509
8	2	-0.468292906	.3010	-0.359098018
9	2	-0.468292906	.3010	-0.359098018
10	2	-0.468292906	.3010	-0.359098018
11	2	-0.468292906	.3010	-0.359098018
12	2	-0.468292906	.3010	-0.359098018
13	3	0	.47712	0.229651817
14	3	0	.47712	0.229651817
15	4	0.468292906	.60205	0.647376472
16	4	0.468292906	.60205	0.647376472
17	4	0.468292906	.60205	0.647376472
18	4	0.468292906	.60205	0.647376472
19	4	0.468292906	.60205	0.647376472
20	4	0.468292906	.60205	0.647376472
21	4	0.468292906	.60205	0.647376472
22	5	0.936585812	.69897	0.971388888
23	6	1.404878717	.7781	1.236126307
24	6	1.404878717	.7781	1.236126307
25	7	1.873171623	.8450	1.459958606
26	7	1.873171623	.8450	1.459958606
Mean	3		.4084	
Standard Deviation	2.13541565		.2990	

Table 26. Standardized in-degree values for Sudoku application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	0	-1.527525232	0	-1.105803863
2	1	-0.763762616	0	-1.105803863
3	1	-0.763762616	0	-1.105803863
4	4	1.527525232	0.60205	1.361845632
5	2	0	.3010	0.128020884
6	2	0	.3010	0.128020884
7	3	0.763762616	.47712	0.849762094
8	3	0.763762616	.47712	0.849762094
Mean	2		0.2697	
Standard Deviation	1.309307341		0.24398	

Table 27. Standardized in-degree values for Snort alert monitor application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	0	-1.653594569	0	-1.718444332
2	2	-0.330718914	.3010	-0.330768298
3	2	-0.330718914	.3010	-0.330768298
4	2	-0.330718914	.3010	-0.330768298
5	2	-0.330718914	.3010	-0.330768298
6	3	0.330718914	.47712	0.480970146
7	4	0.992156742	0.60205	1.056907737
8	5	1.653594569	.69897	1.503639639
9	2	-0.330718914	.3010	-0.330768298
10	2	-0.330718914	.3010	-0.330768298
Mean	2.5		.372	
Standard Deviation	1.511857892		.216	

Table 28. Standardized in-degree values for Lines &dot application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log- normal value
1	0	-1.64646459	0	-0.80965595
2	1	-0.62452105	0	-0.80965595
3	1	-0.62452105	0	-0.80965595
4	1	-0.62452105	0	-0.80965595
5	1	-0.62452105	0	-0.80965595
6	1	-0.62452105	0	-0.80965595
7	1	-0.62452105	0	-0.80965595
8	1	-0.62452105	0	-0.80965595
9	1	-0.62452105	0	-0.80965595
10	1	-0.62452105	0	-0.80965595
11	2	0.397422488	.3010	0.623374004
12	2	0.397422488	.3010	0.623374004
13	2	0.397422488	.3010	0.623374004
14	2	0.397422488	.3010	0.623374004
15	2	0.397422488	.3010	0.623374004
16	3	1.419366029	.47712	1.461642793
17	3	1.419366029	.47712	1.461642793
18	4	2.44130957	.60205	2.056403964
19	1	-0.62452105	0	-0.80965595
20	1	-0.62452105	0	-0.80965595
Mean	1.611111111		.2100	
Standard Deviation	0.978527639		.26974	

Table 29. Standardized in-degree values for Java game maker application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal-value
1	0	-1.63593889	0	-1.32487592
2	1	-1.02935480	0	-1.32487592
3	1	-1.02935480	0	-1.32487592
4	1	-1.02935480	0	-1.32487592
5	1	-1.02935480	0	-1.32487592
6	1	-1.02935480	0	-1.32487592
7	1	-1.02935480	0	-1.32487592
8	1	-1.02935480	0	-1.32487592
9	1	-1.02935480	0	-1.32487592
10	4	0.790397443	.60205	0.903125936
11	4	0.790397443	.60205	0.903125936
12	4	0.790397443	.60205	0.903125936
13	4	0.790397443	.60205	0.903125936
14	4	0.790397443	.60205	0.903125936
15	4	0.790397443	.60205	0.903125936
16	2	-0.42277072	.3010	-0.21087499
17	2	-0.42277072	.3010	-0.21087499
18	2	-0.42277072	.3010	-0.21087499
19	2	-0.42277072	.3010	-0.21087499
20	2	-0.42277072	.3010	-0.21087499
21	2	-0.42277072	.3010	-0.21087499
22	2	-0.42277072	.3010	-0.21087499
23	2	-0.42277072	.3010	-0.21087499
24	2	-0.42277072	.3010	-0.21087499
25	2	-0.42277072	.3010	-0.21087499
26	3	0.183813359	.47712	0.440773775
27	3	0.183813359	.47712	0.440773775
28	4	0.790397443	.60205	0.903125936
29	4	0.790397443	.60205	0.903125936
30	5	1.396981527	.69897	1.261754134
31	6	2.003565611	.7781	1.554774707
32	6	2.003565611	.7781	1.554774707
33	6	2.003565611	.7781	1.554774707
Mean	2.696969697		.358	
Standard Deviation	1.648576061		.270	

Table 30. Standardized in-degree values for Jippy-snake application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	0	-1.264102297	0	-1.034909711
2	1	-0.733179332	0	-1.034909711
3	1	-0.733179332	0	-1.034909711
4	1	-0.733179332	0	-1.034909711
5	1	-0.733179332	0	-1.034909711
6	1	-0.733179332	0	-1.034909711
7	1	-0.733179332	0	-1.034909711
8	1	-0.733179332	0	-1.034909711
9	2	-0.202256368	.3010	0.033589021
10	2	-0.202256368	.3010	0.033589021
11	2	-0.202256368	.3010	0.033589021
12	2	-0.202256368	.3010	0.033589021
13	2	-0.202256368	.3010	0.033589021
14	2	-0.202256368	.3010	0.033589021
15	3	0.328666597	.47712	0.658620711
16	3	0.328666597	.47712	0.658620711
17	3	0.328666597	.47712	0.658620711
18	4	0.859589562	.60205	1.102087752
19	4	0.859589562	.60205	1.102087752
20	6	1.921435492	.7781	1.727119442
21	8	2.983281422	.9030	2.170586483
Mean	2.380952381		.291	
Standard Deviation	1.883512423		.281	

Table 31. Standardized in-degree values for Pocket-basket application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	0	-0.744992315	0	-0.438397299
2	1	-0.325934138	0	-0.438397299
3	1	-0.325934138	0	-0.438397299
4	1	-0.325934138	0	-0.438397299
5	1	-0.325934138	0	-0.438397299
6	1	-0.325934138	0	-0.438397299
7	1	-0.325934138	0	-0.438397299
8	2	0.093124039	.3010	0.547996624
9	8	2.607473104	.9030	2.520784472
Mean	1.777777778		.133	
Standard Deviation	2.386303511		.305	

Table 32. Standardized in-degree values for Thief-script application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	0	-1.47152277	0	-0.93606415
2	1	-0.78650355	0	-0.93606415
3	1	-0.78650355	0	-0.93606415
4	1	-0.78650355	0	-0.93606415
5	1	-0.78650355	0	-0.93606415
6	1	-0.78650355	0	-0.93606415
7	1	-0.78650355	0	-0.93606415
8	1	-0.78650355	0	-0.93606415
9	1	-0.78650355	0	-0.93606415
10	1	-0.78650355	0	-0.93606415
11	1	-0.78650355	0	-0.93606415
12	1	-0.78650355	0	-0.93606415
13	1	-0.78650355	0	-0.93606415
14	2	-0.10148432	.3010	0.168045642
15	2	-0.10148430	.3010	0.168045642
16	2	-0.10148432	.3010	0.168045642
17	2	-0.10148432	.3010	0.168045642
18	3	0.583534894	.47712	0.81390847
19	3	0.583534894	.47712	0.81390847
20	3	0.583534894	.47712	0.81390847
21	3	0.583534894	.47712	0.81390847
23	4	1.268554118	.60205	1.272155439
24	5	1.953573342	.69897	1.627599402
25	4	1.268554118	.60205	1.272155439
26	5	1.953573342	.69897	1.627599402
Mean	2.148148148		.1700	
Standard Deviation	1.459813047		.2100	

Table 34. Standardized in-degree values for Free-cell application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	0	-1.924500897	0	-1.260799513
2	1	-0.769800359	0	-1.260799513
3	1	-0.769800359	0	-1.260799513
4	2	0.384900179	.3010	0.462398701
5	2	0.384900179	.3010	0.462398701
6	2	0.384900179	.3010	0.462398701
Mean	1.666666667		.2202	
Standard Deviation	0.866025404		.1746	

Table 33. Standardized in-degree values for J-controller application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	0	-1.60773429	0	-1.10267968
2	1	-0.87922969	0	-1.10267968
3	1	-0.87922969	0	-1.10267968
4	1	-0.87922969	0	-1.10267968
5	1	-0.87922969	0	-1.10267968
6	1	-0.87922969	0	-1.10267968
7	1	-0.87922969	0	-1.10267968
8	1	-0.87922969	0	-1.10267968
9	1	-0.87922969	0	-1.10267966
10	1	-0.87922969	0	-1.10267968
11	1	-0.87922969	0	-1.10267968
12	2	-0.15072509	.3010	0.086498492
13	2	-0.15072509	.3010	0.086498492
14	2	-0.15072509	.3010	0.086498492
15	2	-0.15072509	.3010	0.086498492
16	2	-0.15072509	.3010	0.086498492
17	2	-0.15072509	.3010	0.086498492
18	2	-0.15072509	.3010	0.086498492
19	2	-0.15072509	.3010	0.086498492
20	3	0.577779511	.47712	0.782123133
21	3	0.577779511	.47712	0.782123133
22	3	0.577779511	.47712	0.782123133
23	3	0.577779511	.47712	0.782123133
24	3	0.577779511	.47712	0.782123133
25	4	1.306284113	.60205	1.27567667
26	4	1.306284113	.60205	1.27567667
27	5	2.034788714	.69897	1.658506536
29	5	2.034788714	.69897	1.658506536
Mean	2.206896552		.2791	.
Standard Deviation	1.372674926		.253	

The standard normal test for the in-degrees was conducted in the following way. The standardized in-degrees of all the applications were taken together to perform the test. The intervals were divided symmetrically with a range of (0.5) for each interval. Following is the standard normal test for in-degrees. The following set of hypotheses is tested:

H_0: The standardized in-degrees for each module follow a standard-normal distribution.

H_1: The standardized in-degrees for each module do not follow a standard-normal distribution.

Table 35. Standardized in-degree values for Minesweeper application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	1	-0.966791393	0	-1.381150286
2	1	-0.966791393	0	-1.381150286
3	1	-0.966791393	0	-1.381150286
4	1	-0.966791393	0	-1.381150286
5	2	-0.281980823	.3010	-0.08731165
6	2	-0.281980823	.3010	-0.08731165
7	2	-0.281980823	.3010	-0.08731165
8	2	-0.281980823	.3010	-0.08731165
9	2	-0.281980823	.3010	-0.08731165
10	2	-0.281980823	.3010	-0.08731165
11	2	-0.281980823	.3010	-0.08731165
12	3	0.402829747	.47712	0.669535433
13	3	0.402829747	.47712	0.669535433
14	3	0.402829747	.47712	0.669535433
15	3	0.402829747	.47712	0.669535433
16	4	1.087640318	.60205	1.206526985
17	7	3.142072029	.8450	2.251113976
Mean	2.411764706		.3213	
Standard Deviation	1.460257834		.2320	

The calculation of the test statistic is given in Table 37. The data categories used for this test and the calculations are given in Table 37. The calculated test statistic is compared to the upper value on a chi-square table with 5 degrees of freedom with alpha equal to 0.05. The reason 5 degrees of freedom was used is because we had 8 categories for the data and the mean and standard deviation has to be estimated for each program size. Since we estimated two parameters mean and standard deviation for each of the modules, the degrees of freedom would be 8 minus 1 minus 2 degrees of freedom. This estimation was done individually for each application type since the mean and variance would change with the size of the application.

Table 36. Standardized in-degree values for Black-jack analyst application.

Module	in-degree	Standardized normal value	log-values of the in-degree	Standardized log-normal value
1	0	-1.18338544	0	-1.29638048
2	1	-0.82973002	0	-1.29638048
3	1	-0.82973002	0	-1.29638048
4	1	-0.82973002	0	-1.29638048
5	1	-0.82973002	0	-1.29638048
6	1	-0.82973002	0	-1.29638048
7	1	-0.82973002	0	-1.29638048
8	4	0.231236237	.60205	0.58808009
9	4	0.231236237	.60205	0.58808009
10	4	0.231236237	.60205	0.58808009
11	2	-0.47607460	.3010	-0.35415019
12	2	-0.47607460	.3010	-0.35415019
13	2	-0.47607460	.3010	-0.35415019
14	2	-0.47607460	.3010	-0.35415019
15	3	-0.12241918	.47712	0.197019188
16	3	-0.12241918	.47712	0.197019188
17	3	-0.12241918	.47712	0.197019188
18	3	-0.12241918	.47712	0.197019188
19	3	-0.12241918	.47712	0.197019188
20	3	-0.12241918	.47712	0.197019188
21	4	0.231236237	.60205	0.58808009
22	4	0.231236237	.60205	0.58808009
23	6	0.93854708	.7781	1.139249474
24	7	1.292202501	.8450	1.348794349
25	9	1.999513343	.9542	1.690418858
26	13	3.414135029	1.1139	2.190285891
Mean	3.346153846		.4141	
Standard Deviation	2.827611115		.3194	

The chi-square value at 5 degrees of freedom and .05 level of significance is 11.070 which is less than the calculated value 33.1967. We can reject the null hypothesis and conclude that the data of in-degrees did not come from a standard normal distribution. We followed the same procedure for the out-degrees that was followed for the in-degrees. The following set of hypotheses is tested:

H_0: The standardized out-degrees for each module follow a standard-normal distribution.

H_1: The standardized out-degrees for each module do not follow a standard-normal distribution.

Table 37. Calculation of chi-square statistic for the standardized in-degree values to test for the standard normal distribution

	Range	Relative frequency	Observed frequency	Expected frequency	chi-square
1	<-1.3	0.0968	9	20.6184	6.54693
2	-1.3 to-0.8	0.1151	30	24.5163	1.22657
3	-0.8 to-0.3	0.1702	57	36.2526	11.87376
4	-0.3 to 0.2	0.1972	41	42.0036	0.023979
5	0.2 to 0.7	0.1787	33	38.0631	0.673486
6	0.7 to 1.2	0.1269	14	27.0297	6.280983
7	1.2 to 1.7	0.0705	12	15.0165	0.605952
8	>1.7	0.0446	17	9.4998	5.92149
			213		33.15315

The calculation of the test statistic is given in Table 38. The data categories used for this test and the calculation are given in Table 38. The calculated test statistic is compared to the upper value on a chi-square table with 5 degrees of freedom with alpha equal to 0.05. The reason 5 degrees of freedom was used is because we had 8 categories for the data and the mean and standard deviation has to be estimated for each program size. Since we estimated two parameters, mean and standard deviation for each of the modules, the degrees of freedom would be 8 minus 1 minus 2 degrees of freedom. The chi-square value at 5 degrees of freedom and .05 level of significance is 11.070 which is less than the calculated value 36.7282. We can reject the null hypothesis and conclude that the data of out-degrees did not come from a standard normal distribution. We can reject the null hypothesis and conclude that the data of out-degrees did not come from a standard normal distribution.

Table 38. Calculation of chi-square statistic for the standardized out-degree values test for the standard normal distribution

	Range	Relative frequency	Observed frequency	Expected Frequency	chi-square
1	<-1.3	0.0968	5	20.6184	11.83091
2	-1.3 to -0.8	0.1151	40	24.5163	9.779003
3	-0.8 to -0.3	0.1702	50	36.2526	5.213171
4	-0.3 to 0.2	0.1972	45	42.0036	0.213753
5	0.2 to 0.7	0.1787	33	38.0631	0.673486
6	0.7 to 1.2	0.1269	14	27.0297	6.280983
7	1.2 to 1.7	0.0705	12	15.0165	0.605952
8	>1.7	0.0446	14	9.4998	2.131
			213		36.7282

We can reject the null hypothesis and conclude that the data of out-degrees did not come from a standard normal distribution.

We again tested the hypothesis that the in-degrees followed a normal distribution. This time we did not standardize values according to the module size, but we assumed that the module size would not affect the mean and standard deviation. In this case the actual values of the number of in-degrees for each module were combined together (without standardizing first) to test for a normal distribution. In this case the parameters had to be estimated first. A common mean and the standard deviation were calculated. The mean was found to be (2.30) and the standard deviation (1.77). All the in-degrees were subtracted from the mean and divided by the standard deviation to get the normalized value. The intervals were divided symmetrically with a range of (0.5).

Following is the test for in-degrees: The following set of hypotheses is tested:

H_0: The number of in-degrees follows a normal distribution.

H_1: The number of in-degrees does not follow normal distribution.

The calculation of the test statistic is given in Table 39.

Table 39. Calculation of chi-square statistic for the in-degrees to test for normal distribution

	Range	Relative frequency	Observed frequency	Expected frequency	chi-square
1	<-1.3	0.0968	0	20.6184	20.6184
2	-1.3 to -0.8	0.1151	12	24.5163	6.389943
3	-0.8 to -0.3	0.1702	69	36.2526	29.58111
4	-0.3 to 0.2	0.1972	57	42.0036	5.354113
5	0.2 to 0.7	0.1787	31	38.0631	1.310649
6	0.7 to 1.2	0.1269	23	27.0297	0.600764
7	1.2 to 1.7	0.0705	8	15.0165	3.278478
8	>1.7	0.0446	13	9.4998	1.2896
			213		68.423

The data categories used for this test and the calculations are given in Table 39. The calculated test statistic is compared to the upper value on a chi-square table with 5 degrees of freedom with alpha equal to 0.05. The reason 5 degrees of freedom was used is because we had 8 categories for the data and the mean and standard deviation has to be estimated for all the in-degrees of the modules. Since we estimated two parameters mean (2.30) and standard deviation (1.77) from data of in-degrees, the degrees of freedom would be 8 minus 1 minus 2 degrees of freedom. The chi-square value at 5 degrees of freedom and .05 level of significance is 11.070 which is less than the calculated value 68.423. We can reject the null hypothesis and conclude that the data of in-degrees did not come from a normal distribution.

We followed the same procedure for the out-degrees that were followed for the in-degrees. The mean was found to be (2.25) and the standard deviation (2.18). All the out-degrees were subtracted from the mean and divided by the standard deviation to get the normalized value. The following set of hypotheses is tested:

H_0: The number of out-degrees follows a normal distribution.

H_1: The number of out-degrees does not follow a normal distribution.

The calculation of the test statistic is given in Table 40.

Table 40. Calculation of chi-square statistic for the out-degrees to test for normal distribution

	Range	Relative frequency	Observed frequency	Expected frequency	chi-square
1	<-1.3	0.0968	0	20.6184	20.6184
2	-1.3 to -0.8	0.1151	40	24.5163	9.779003
3	-0.8 to -0.3	0.1702	50	36.2526	5.213171
4	-0.3 to 0.2	0.1972	48	42.0036	0.856041
5	0.2 to 0.7	0.1787	29	38.0631	2.15799
6	0.7 to 1.2	0.1269	18	27.0297	3.016515
7	1.2 to 1.7	0.0705	11	15.0165	1.074303
8	>1.7	0.0446	17	9.4998	5.921
			213		48.6364

The data categories used for this test and the calculations are given in Table 40. The calculated test statistic is compared to the upper value on a chi-square table with 5 degrees of freedom with alpha equal to 0.05. The reason 5 degrees of freedom was used is because we had 8 categories for the data and the mean and standard deviation has to be estimated for all the in-degrees of the modules. Since we estimated two parameters mean (2.25) and standard deviation (2.18) from data of in-degrees, the degrees of freedom would be 8 minus 1 minus 2 degrees of freedom. The chi-square value at 5 degrees of freedom and .05 level of significance is 11.070 which is less than the calculated value 48.6364. We can reject the null hypothesis and conclude that the data of in-degrees did not come from a normal distribution.

The next test was conducted was to test whether the number of in-degrees followed a log-normal distribution. The data of in-degrees of all the applications were first converted into respective log-values. All the log-values were taken to perform the test. A common mean and the standard deviation were calculated from the log-values. The mean was found to be (0.2828) and the standard deviation (0.2911). All the transformed log-values were subtracted from the mean and divided by the standard deviation to get the normalized value. The intervals are divided symmetrically with a

range of (0.5). The same procedure has been followed for the in-degrees. The following set of hypotheses is tested:

H_0: The number of out-degrees follows a log-normal distribution.

H_1: The number of out-degrees does not follow a log-normal distribution.

The calculation of the test statistic is given in Table 41. The data categories used for this test and the calculation are given in Table 41. The calculated test statistic is compared to the upper value on a chi-square table with 6 degrees of freedom with alpha equal to 0.05. The reason 6 degrees of freedom was used is because we had 9 categories for the data and the mean and standard deviation has to be estimated for each program size.

Table 41. Calculation of chi-square statistic for the out-degrees to test for log-normal distribution

	Range	Observed frequency	Relative frequency	Expected frequency	Chi-square
1	< -1.3	0	0.0968	20.6184	20.6184
2	-1.3 to -0.8	92	0.1151	24.5163	185.5163
3	-0.8 to -0.3	0	0.1702	36.2526	36.2526
4	-0.3 to 0.2	48	0.1972	42.0036	0.8560
5	0.2 to 0.62	0	0.1453	30.9489	30.9489
6	0.62 to 1.12	47	0.1362	29.0106	11.15518
7	1.12 to 1.62	9	0.0788	16.7844	3.61031
8	1.6 to 2.12	12	0.0356	7.5828	2.5731
9	>2.12	5	0.0248	5.2824	0.0150
		213			291.7706

Since we estimated two parameters mean and standard deviation, for each application, the degrees of freedom would be 9 minus 1 minus 2 degrees of freedom. The chi-square value at 6 degrees of freedom and .05 level of significance is 12.592

which is less than the calculated value 291.7706. We can reject the null hypothesis and conclude that the data of out-degrees did not come from a log-normal distribution.

A similar test was conducted for the in-degrees. The following set of hypotheses is tested:

H_0: The number of in-degrees follows a log-normal distribution.

H_1: The number of in-degrees does not follow a log-normal distribution.

The calculation of the test statistic is given in Table 42.

Table 42. Calculation of chi-square statistic for the in-degrees to test for log-normal distribution

	Range	Observed frequency	Relative frequency	Expected frequency	Chi-square
1	< -1.3	0	0.0968	20.6184	20.6184
2	-1.3 to -0.8	83	0.1151	24.5163	139.513
3	-0.8 to -0.3	0	0.1702	36.2526	36.2526
4	-0.3 to 0.2	57	0.1972	42.0036	5.35411
5	0.2 to 0.62	0	0.1453	30.9489	30.9489
6	0.62 to 1.12	32	0.1362	29.0106	0.30804
7	1.12 to 1.62	30	0.0788	16.7844	10.4056
8	1.6 to 2.12	7	0.0356	7.5828	0.04479
9	>2.12	4	0.0248	5.2824	0.31132
		213			243.7568

The data categories used for this test and the calculation are given in Table 42. The calculated test statistic is compared to the upper value on a chi-square table with 6 degrees of freedom with alpha equal to 0.05. The reason 6 degrees of freedom was used is because we had 9 categories for the data and the mean and standard deviation has to be estimated for the transformed log-values. Since we estimated two parameters mean (0.2850) and standard deviation(0.2683), for each application, the degrees of freedom would be 9 minus 1 minus 2 degrees of freedom. The chi-square value at 6 degrees of freedom and .05 level of significance is 12.592 which is less than the calculated value

243.7568. We can reject the null hypothesis and conclude that the data of out-degrees did not come from a log-normal distribution.

The next test that was conducted was to test whether standardized in-degrees values followed a standard log-normal distribution. Following was the method used: The data of in-degrees of all the applications were first converted into respective log-values. The mean and standard deviation were calculated for the log-values for each of the applications. In order to get the standardized value, the log-values of in-degrees in each application were subtracted from the mean and divided by the standard deviation of that application, in this way all the in-degrees of all applications were standardized. The standardized in-degree values of all the applications were taken together to perform the test. The intervals are divided symmetrically with a range of (0.5). The same procedure has been followed for the out-degrees. The standardized values for each application are given in Tables 13-36. The following set of hypotheses is tested:

H_0: The standardized out-degrees for each module follow a standard log-normal distribution.

H_1: The standardized out-degrees for each module do not follow a standard log-normal distribution.

The calculation of the test statistic is given in Table 43. The data categories used for this test and the calculation are given in Table 43. The calculated test statistic is compared to the upper value on a chi-square table with 6 degrees of freedom with alpha equal to 0.05. The reason 6 degrees of freedom was used is because we had 9 categories for the data and the mean and standard deviation has to be estimated for each program size. Since we estimated two parameters mean and standard deviation, for each application, the degrees of freedom would be 9 minus 1 minus 2 degrees of freedom.

Table 43. Calculation of chi-square statistic for the standardized out-degree values to test for the standard log-normal distribution

	Range	Relative frequency	Observed frequency	Expected frequency	chi-square
1	<-1.3	0.0968	7	20.6184	8.994918
2	-1.3 to -0.8	0.1151	61	24.5163	54.29287
3	-0.8 to -0.3	0.1702	29	36.2526	1.450936
4	-0.3 to 0.2	0.1972	35	42.0036	1.167767
5	0.2 to 0.62	0.1453	34	30.9489	0.300793
6	0.62 to 1.12	0.1362	19	29.0106	3.454327
7	1.12 to 1.62	0.0788	18	16.7844	0.088039
8	1.6 to 2.12	0.0356	4	7.5828	1.692839
9	>2.12	0.0248	6	5.2824	0.097484
			213		71.53998

The chi-square value at 6 degrees of freedom and .05 level of significance is 12.592 which is less than the calculated value 71.539. We can reject the null hypothesis and conclude that the data of out-degrees did not come from a standard log-normal distribution.

Following is the log-normal test for the in-degrees. The following set of hypotheses is tested:

H_0: The standardized in-degrees for each module follow a standard log-normal distribution.

H_1: The standardized in-degrees for each module do not follow a standard log-normal distribution.

The calculation of the test statistic is given in Table 44. The data categories used for this test and the calculation are given in Table 44. The calculated test statistic is compared to the upper value on a chi-square table with 6 degrees of freedom with alpha equal to 0.05. The reason 6 degrees of freedom was used is because we had 9 categories

for the data and the mean and standard deviation has to be estimated for each program size.

Table 44. Calculation of chi-square statistic for the standardized in-degree values to test for the standard log-normal distribution

	Range	Relative frequency	Observed frequency	Expected frequency	chi-square
1	<-1.3	0.0968	5	20.6184	11.83091
2	-1.3 to -0.8	0.1151	54	24.5163	35.45758
3	-0.8 to -0.3	0.1702	17	36.2526	10.22444
4	-0.3 to 0.2	0.1972	46	42.0036	0.380234
5	0.2 to 0.62	0.1453	26	30.9489	0.791356
6	0.62 to 1.12	0.1362	35	29.0106	1.236545
7	1.12 to 1.62	0.0788	16	16.7844	0.036658
8	1.6 to 2.12	0.0356	9	7.5828	0.26487
9	>2.12	0.0248	5	5.2824	0.015097
			213		60.23769

Since we estimated two parameters mean and standard deviation for each application, the degrees of freedom would be 9 minus 1 minus 2 degrees of freedom. The chi-square value at 6 degrees of freedom and .05 level of significance is 12.592 which is less than the calculated value 60.2379. We can reject the null hypothesis and conclude that the data of out-degrees did not come from a log-normal distribution. Since the data are actually discrete data, we next tested several hypotheses that involved discrete distribution such as poisson, uniform, geometric.

The next step was to try and test the data of in-degrees and the data of out-degrees followed a poisson distribution. For the poisson distribution, the in-degrees of all the modules were taken were taken together to perform this test. The mean was estimated for the in-degrees which was (2.30). The intervals for both the in-degrees and out-degrees were grouped according to the observations. The function $F(x) = (m^x.e^{-m}/x!)$ where m= mean of the in-degrees, x= observations of the in-degrees. This function

was evaluated for each of the observations present to get the relative frequencies for in-degrees. The following set of hypotheses is tested:

H_0: The number of in-degrees follows a poisson distribution.

H_1: The number of in-degrees does not follow a poisson distribution.

The calculation of the test statistic is given in Table 45. The data categories used for this test and the calculation are given in Table 45. The calculated test statistic is compared to the upper value on a chi-square table with 4 degrees of freedom with alpha equal to 0.05.

Table 45. Calculation of chi-square statistic for the in-degrees to test for poisson distribution

	Observations	Observed frequency	Relative frequency	Expected frequency	chi-square
1	0	12	0.10026	21.3551	4.09824
2	1	71	0.2306	49.1168	9.7497
3	2	57	0.2652	56.4843	0.00471
4	3	31	0.2033	43.3047	3.49626
5	4	23	0.1169	24.9002	0.14501
6	>5	19	0.08375	17.8389	0.07557
		213			17.5695

The reason 4 degrees of freedom was used is because we had 6 categories for the data and the mean has to be estimated, which is (2.30) from the data, the degrees of freedom would be 6 minus 2 degrees of freedom. The critical value at 4 degrees of freedom is 9.48 which is less than the calculated value 17.5695. We can reject the null hypothesis and conclude that the data did not come from a poisson distribution. The same procedure was followed for the out-degrees that had been followed for the in-degrees. The mean was estimated to be (2.25) for the out-degrees. The following set of hypotheses is tested:

H_0: The number of out-degrees follows a poisson distribution.

H_1: The number of out-degrees does not follow a poisson distribution.

The calculation of the test statistic is given in Table 46. The data categories used for this test and the calculation are given in Table 46. The calculated test statistic is compared to the upper value on a chi-square table with 4 degrees of freedom with alpha equal to 0.05. The reason 4 degrees of freedom was used is because we had 6 categories for the data and the mean has to be estimated, which is (2.25) from the data, the degrees of freedom would be 6 minus 2 degrees of freedom.

Table 46. Calculation of chi-square statistic for out-degrees to test for poisson distribution

	Observations	Observed frequency	Relative frequency	Expected frequency	chi-square
1	0	42	0.105399	22.45003	17.02452
2	1	50	0.237148	50.51258	0.005201
3	2	48	0.266792	56.82665	1.371007
4	3	29	0.200094	42.61999	4.352513
5	4	18	0.112553	23.97374	1.488529
6	>5	26	0.078014	16.61701	5.298223
		213			29.54

The critical value at 4 degrees of freedom is 9.48 which is less than the calculated value 29.54. We can reject the null hypothesis and conclude that the data did not come from a poisson distribution.

The next test that was carried out was whether the data of in-degrees had come from a uniform distribution on categories (0, 1,2,3,4 or more) and also (0, 1, 2, 3 or more). The intervals for this test were decided before hand on categories (0, 1,2,3,4 or more) and also (0, 1, 2, 3 or more) and data of in-degrees were grouped according to

their frequencies. The probabilities of (0.20) and (0.25) were assumed as relative frequencies for the observations. Following are tests for in-degrees and out-degrees with probability (0.20). The same procedure mentioned above has been followed to test whether the data of out-degrees had come from a uniform distribution. The following set of hypotheses is tested:

H_0: The number of in-degrees follows a uniform distribution with a probability equal to 0.20 for each category.

H_1: The number of in-degrees does not follow a uniform distribution with a probability equal to 0.20 for each category.

The calculation of the test statistic is given in Table 47.

Table 47. Calculation of chi-square statistic for the in-degrees to test for uniform distribution with probability (0.20)

Observations	Observed frequency	Relative frequency	Expected frequency	chi-square
0	12	0.20	42.6	21.98028
1	71	0.20	42.6	18.93333
2	57	0.20	42.6	4.867606
3	31	0.20	42.6	3.158685
$\geq$ 4	42	0.20	42.6	0.008451
	213			48.94836

The data categories used for this test and the calculation are given in Table 47. The calculated test statistic is compared to the upper value on a chi-square table with 4 degrees of freedom with alpha equal to 0.05. The chi-square value at 4 degrees of freedom and .05 level of significance is 9.48 which is less than the calculated value 48.94. Hence we reject the null hypothesis and conclude that the data of in-degrees does not follow a uniform distribution with probability of (0.20) of falling in each category.

The following set of hypotheses is tested for the out-degrees:

H_0: The number of out-degrees follows a uniform distribution with a probability equal to 0.20 for each category.

H_1: The number of out-degrees does not follow a uniform distribution with a probability equal to 0.20 for each category.

The calculation of the test statistic is given in Table 48.

Table 48. Calculation of chi-square statistic for the out-degrees to test for uniform distribution with probability (0.20)

Observations	Observed frequency	Relative frequency	Expected frequency	chi-square
0	42	0.20	42.6	0.008451
1	50	0.20	42.6	1.285446
2	48	0.20	42.6	0.684507
3	29	0.20	42.6	4.341784
≥4	44	0.20	42.6	0.046009
	213			6.366197

The data categories used for this test and the calculation are given in Table 48. The calculated test statistic is compared to the upper value on a chi-square table with 4 degrees of freedom with alpha equal to 0.05. The chi-square value at 4 degrees of freedom and .05 level of significance is 9.48 which is greater than the calculated value 6.366. Hence we reject the null hypothesis and conclude that the data of out-degrees does not follow a uniform distribution with probability of (0.20) of falling in each category.

The next step was to conduct to test whether the data comes from a population with a probability of 0.25 falling in each category of (0, 1, 2, 3 or more). The data of in-degrees were taken together to perform the test. The observations were grouped

respective to their frequencies. Following is the test for in-degrees and out-degrees. The following set of hypotheses is tested:

H_0: The number of in-degrees follows a uniform distribution with a probability equal to 0.25 for each category.

H_1: The number of in-degrees does not follow a uniform distribution with a probability equal to 0.25 for each category.

The calculation of the test statistic is given in Table 49.

Table 49. Calculation of chi-square statistic for the in-degrees to test for uniform distribution with probability (0.25)

Observations	Observed frequency	Relative frequency	Expected frequency	chi-square
0	12	0.25	53.25	31.95423
1	71	0.25	53.25	5.916667
2	57	0.25	53.25	0.264085
$\geq$ 3	53	0.25	53.25	0.001174
	213			38.13615

The data categories used for this test and the calculation are given in Table 49. The calculated test statistic is compared to the upper value on a chi-square table with 2 degrees of freedom with alpha equal to 0.05. The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is less than the calculated value 38.13. Hence we reject the null hypothesis and conclude that the data of in-degrees does not follow a uniform distribution with probability of (0.25) of falling in each category.

A similar test was carried out for the out-degrees. The following set of hypotheses is tested:

H_0: The number of data of out-degrees follows a uniform distribution with a probability equal to 0.25 for each category.

H_1: The number of data of out-degrees does not follow a uniform distribution with a probability equal to 0.25 for each category.

The calculation of the test statistic is given in Table 50.

Table 50. Calculation of chi-square statistic for the out-degrees to test for uniform distribution with probability (0.25)

Observations	Observed frequency	Relative frequency	Expected frequency	chi-square
0	42	0.25	53.25	2.37676
1	50	0.25	53.25	0.19835
2	48	0.25	53.25	0.51760
≥3	73	0.25	53.25	7.32511
	213			10.4178

The data categories used for this test and the calculation are given in Table 50. The calculated test statistic is compared to the upper value on a chi-square table with 3 degrees of freedom with alpha equal to 0.05. The chi-square value at 3 degrees of freedom and .05 level of significance is 7.81 which is less than the calculated value 10.41. Hence we reject the null hypothesis and conclude that the data of out-degrees does not follow a uniform distribution with probability of (0.25) of falling in each category.

The next step was to try and test the data of in-degrees and the data of out-degrees followed a geometric distribution. The mean for all the in-degrees was estimated to be 2.30. The expected value of a geometric distribution random variable x is 1/p which is the mean of geometric distribution. The values of p and q were estimated from the above formula. The function $f(x) = p.q^x$ was evaluated using p = (0.4) and q =

(0.6) to get the relative frequencies. The same procedure has been followed for the out-degrees with a mean of 2.25. The following set of hypotheses is tested:

H_0: The number of in-degrees follows a geometric distribution.

H_1: The number of in-degrees does not follow a geometric distribution.

The calculation of the test statistic is given in Table 51.

Table 51. Calculation of chi-square statistic for the in-degree values to test for geometric distribution

Observations	Observed frequency	Relative frequency	Expected Frequency	chi-square
0	12	0.40	85.20	62.890
1	71	0.24	51.12	7.731
2	57	0.144	30.67	22.599
3	31	0.086	18.40	8.622
4	42	0.051	11.04	86.809
	213			188.651

The data categories used for this test and the calculation are given in Table 51. The calculated test statistic is compared to the upper value on a chi-square table with 2 degrees of freedom with alpha equal to 0.05. The reason 2 degrees of freedom was used is because we had 4 categories for the data and the mean has to be estimated, since we estimated one parameter mean for each application from the data, the degrees of freedom would be 4 minus 2 degrees of freedom. The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is less than the calculated value 188.651. We can reject the null hypothesis and conclude that the data of in-degrees did not come from a geometric distribution.

A similar test was conducted for the out-degrees: The following set of hypotheses is tested:

H_0: The number of out-degrees follows a geometric distribution.

H_1: The number of out-degrees does not follow geometric distribution.

The calculation of the test statistic is given in Table 52.

Table 52. Calculation of chi-square statistic for the out-degree values to test for geometric distribution

Observations	Observed frequency	Relative frequency	Expected frequency	chi-square
0	42	0.40	85.20	21.904
1	50	0.24	51.12	0.024
3	48	0.144	30.67	9.789
3	29	0.086	18.40	6.102
4	44	0.051	11.04	98.387
	213			136.206

The data categories used for this test and the calculation are given in Table 52. The calculated test statistic is compared to the upper value on a chi-square table with 2 degrees of freedom with alpha equal to 0.05. The reason 2 degrees of freedom was used is because we had 4 categories for the data and the mean has to be estimated, since we estimated one parameter mean for each application from the data, the degrees of freedom would be 4 minus 2 degrees of freedom. The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is less than the calculated value 136.206. We can reject the null hypothesis and conclude that the data of out-degrees did not come from a geometric distribution.

Since all of the hypotheses for well-known discrete distributions that we tested for were rejected, we went back to trying continuous distributions. We know that the data are not continuous data but perhaps their underlying distributions can be approximated by continuous distribution. We have already tried the normal and log-

normal distributions and the null hypothesis that the number of in-degrees (or out-degrees) followed a normal distribution was rejected. This was also true for the log-normal distribution. We will next test hypotheses that the number of in-degrees (or out-degrees) follows an exponential distribution.

The next test that was conducted was to test whether the out-degrees followed an exponential distribution. The data of out-degrees of all the modules were taken together to perform the test. The mean was estimated for the out-degrees. The mean was estimated to be (2.25) for the out-degrees. The intervals for out-degrees were distributed symmetrically with a difference of (1.01). The function $(1/\beta\ e^{(-x/\beta)}$ where β = mean of the out-degrees, was integrated over the intervals to get the relative frequencies for out-degrees. The following set of hypotheses is tested:

H_0: The number of out-degrees follows an exponential distribution.

H_1: The number of out-degrees does not follow an exponential distribution.

The calculation of the test statistic is given in Table 53. The data categories used for this test and the calculation are given in Table 53. The calculated test statistic is compared to the upper value on a chi-square table with 2 degrees of freedom with alpha equal to 0.05. The reason 2 degrees of freedom was used is because we had 4 categories for the data and the mean has to be estimated, since we estimated one parameter mean which is (2.25) from the data, the degrees of freedom would be 4 minus 2 degrees of freedom. The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is greater than the calculated value 5.38. Since the calculated value of the test statistic, (5.38), is not greater than the upper value on a chi-square table with 2 degrees of freedom and .05 significance level,(5.99) we cannot reject the null hypothesis at α = 0.05. Therefore at α = .05 we do not have enough evidence to conclude that the distribution of out-degrees is not exponential.

Table 53. Calculation of chi-square statistic for the out-degrees to test for exponential distribution

	Range	Observed frequency	Relative frequency	Expected frequency	Chi-square
1	0 to 1.01	92	0.361663	77.0342	2.907471
2	1.01 to 2.01	48	0.229048	49.842	0.068074
3	2.01 to 3.01	29	0.146861	31.2813	0.166385
4	>3.01	44	0.258786	55.121	2.243886
		213			5.385

Therefore at $\alpha = .05$ we do not have enough evidence to conclude that the distribution of out-degrees is not exponential. However at $\alpha = .10$, the null hypothesis would be rejected and there would be enough evidence to conclude that the distribution is not exponential.

We followed the same procedure for the in-degrees that were followed for the out-degrees. The mean for the in-degrees of all applications was estimated to be (2.30). The following set of hypotheses is tested:

H_0: The number of in-degrees follows an exponential distribution.

H_1: The number of in-degrees does not follow an exponential distribution.

The calculation of the test statistic is given in Table 54. The data categories used for this test and the calculation are given in Table 54. The calculated test statistic is compared to the upper value on a chi-square table with 2 degrees of freedom with alpha equal to 0.05. The reason 2 degrees of freedom was used is because we had 4 categories for the data and the mean has to be estimated, since we estimated one parameter mean which is (2.30) from the data, the degrees of freedom would be 4 minus 2 degrees of freedom. The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is greater than the calculated value 5.69.

Table 54. Calculation of chi-square statistic for the in-degrees to test for exponential distribution

	Range	Observed frequency	Relative frequency	Expected frequency	chi-square
1	0 to1.01	83	0.355403	75.7009	0.70378
2	1.01 to 2.01	57	0.227281	49.94968	0.995142
3	2.01 to 3.01	31	0.147143	31.34149	0.003721
4	>3.01	42	0.268081	57.10135	3.993791
		213			5.696434

The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is greater than the calculated value 5.69. Since the calculated value of the test statistic, (5.69), is not greater than the upper value on a chi-square table with 2 degrees of freedom and .05 significance level,(5.99) we cannot reject the null hypothesis at $\alpha = 0.05$. Therefore at $\alpha = .05$ we do not have enough evidence to conclude that the distribution of out-degrees is not exponential. However at $\alpha = .10$, the null hypothesis would be rejected and there would be enough evidence to conclude that the distribution is not exponential.

The next test that was conducted was to test whether the standardized out-degrees followed a standard exponential distribution. Following was the method used: each out-degree of a module is divided by the mean number of out-degrees for the application. The procedure is followed for all the 12 applications. All the standardized out-degree values were used to perform the test. In a chi-square goodness of fit test, intervals (or groups) are decided ahead of time. In this case, each interval had a width of 1.01 with the exception of the last interval. The function $f(x) = e^{-x}$, x >0 was integrated over the intervals to get the relative frequencies for the out-degrees. The following set of hypotheses is tested:

H_0: The standardized out-degrees for each module follow a standard-exponential distribution.

H_1: The standardized out-degrees for each module do not follow a standard-exponential distribution.

The calculation of the test statistic is given in Table 55. The data categories used for this test and the calculation are given in Table 55. The calculated test statistic is compared to the upper value on a chi-square table with 2 degrees of freedom with alpha equal to 0.05. The reason 2 degrees of freedom was used is because we had 4 categories for the data and the mean has to be estimated, since we estimated one parameter mean for each application from the data, the degrees of freedom would be 4 minus 2 degrees of freedom.

Table 55. Calculation of chi-square statistic for the standardized out-degree values to test for the standard exponential distribution

	Range	Observed frequency	Relative frequency	Expected Frequency	Chi-square
1	0 to 1.01	129	0.632	134.616	0.234
2	1.01 to 2.01	58	0.230	48.990	1.657
3	2.01 to 3.01	20	0.084	18.041	0.212
4	>3.01	6	0.053	11.289	2.477
		213			4.580

The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is greater than the calculated value 4.580. Since the calculated value of the test statistic, (4.580), is not greater than the upper value on a chi-square table with 2 degrees of freedom and.05 significance level, (5.99) we cannot reject the null hypothesis at $\alpha = 0.05$. Therefore at $\alpha = .05$ we do not have enough evidence to conclude that the distribution of out-degrees is not exponential. However at $\alpha = .10$, the null hypothesis would be rejected and there would be enough evidence to conclude that the distribution is not exponential.

A similar test was conducted for the in-degrees. The following set of hypotheses is tested:

H_0: The standardized in-degrees for each module follow a standard-exponential distribution.

H_1: The standardized in-degrees for each module do not follow a standard-exponential distribution.

The calculation of the test statistic is given in Table 56.

Table 56. Calculation of chi-square statistic for the standardized in-degree values to test for the standard exponential distribution

	Range	Observed frequency	Relative frequency	Expected Frequency	chi-square
1	0 to 1.01	129	0.632	134.616	0.234
2	1.01 to 2.01	66	0.23	48.990	5.906
3	2.01 to 3.01	15	0.0847	18.041	0.512
4	> 3.01	3	0.053	11.289	6.086
		213			12.738

The data categories used for this test and the calculation are given in Table 56. The calculated test statistic is compared to the upper value on a chi-square table with 2 degrees of freedom with alpha equal to 0.05. The reason 2 degrees of freedom was used is because we had 4 categories for the data and the mean has to be estimated, since we estimated one parameter mean for each application from the data, the degrees of freedom would be 4 minus 2 degrees of freedom. The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is less than the calculated value 12.738. We can reject the null hypothesis and conclude that the standardized in-degree values do not follow a standardized exponential distribution.

The next distribution that was tested was the chi-square distribution. The sample means for the in-degrees and out-degrees were 2.30 and 2.25 respectively. The function $f(x) = \frac{1}{2}e^{-x/2}$, x>0 was evaluated over the intervals to get the relative frequencies. The intervals taken for this distribution are in the range of [0, 1) which indicates that the value of 1 is not included in the interval. We tested for a $\chi^2(2)$ since the sample means were close to 2. The same procedure has been followed for the out-degrees. The following set of hypotheses is tested:

H_0: The number of in-degrees follows a chi-square distribution.

H_1: The number of in-degrees does not follow the chi-square distribution.

The calculation of the test statistic is given in Table 57.

Table 57. Calculation of chi-square statistic for in-degrees to test for chi-square distribution

Observations	Observed frequency	Relative frequency	Expected Frequency	chi-square
0 to 1	83	0.3934	83.7942	0.007527
1 to 2	57	0.239	50.907	0.729264
2 to 3	31	0.145	30.885	0.000428
≥3	42	0.223	47.499	0.636624
	213			1.373844

The data categories used for this test and the calculation are given in Table 57. The calculated test statistic is compared to the upper value on a chi-square table with 2 degrees of freedom with alpha equal to 0.05. The reason 2 degrees of freedom was used is because we had 4 categories for the data and the mean has to be estimated. Since we estimated one parameter mean (2.30) for the in-degrees, from the data the degrees of freedom would be 4 minus 2 minus degrees of freedom. The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is greater than the

calculated value 1.373844. We can accept the null hypothesis and conclude that the data of in-degrees did come from a chi-square distribution.

A similar test was done for the out-degrees: The following set of hypotheses is tested:

H_0: The number of out-degrees follows a chi-square distribution.

H_1: The number of out-degrees does not follow the chi-square distribution.

The calculation of the test statistic is given in Table 58.

Table 58. Calculation of chi-square statistic for out-degrees to test for chi-square distribution

Observations	Observed frequency	Relative frequency	Expected frequency	chi-square
0 to 1	92	0.3934	83.7942	0.803578
1 to 2	48	0.239	50.907	0.166002
2 to 3	29	0.145	30.885	0.115047
≥ 3	44	0.223	47.499	0.257753
	213			1.342379

The data categories used for this test and the calculation are given in Table 58. The calculated test statistic is compared to the upper value on a chi-square table with 2 degrees of freedom with alpha equal to 0.05. The reason 2 degrees of freedom was used is because we had 4 categories for the data and the mean has to be estimated for each program size. Since we estimated one parameter mean which was (2.25) for the out-degrees, from the data the degrees of freedom would be 4 minus 2 minus degrees of freedom.

The chi-square value at 2 degrees of freedom and .05 level of significance is 5.99 which is greater than the calculated value 1.342379. We can accept the null

hypothesis and conclude that the data of out-degrees did come from a chi-square distribution

CHAPTER 5: CONCLUSION

Our work formed a pilot study of the pattern of in-degrees and out-degrees of modules within a single application. We restricted the study to twelve, similar-sized applications. However, we were able to establish methods that could be employed in a larger study with more, larger applications. We found that the distribution of in-degree numbers and of out-degree numbers for the modules in an application was similar. Both followed a chi square distribution with two degrees of freedom. We will need to test these results with larger applications including hundreds of modules each. We hypothesize that the number of degrees of freedom will increase with application size (in terms of number of modules), but that the distribution will remain a chi-square.

In this pilot study we found that the distributions of in-degrees and out-degrees were each similar chi-square. A full study would need to address whether or not this similarity remains as application size grows. We considered both discrete and continuous probability distributions. While the number of in-degrees and the number of out-degrees clearly is discrete, the large number of modules in a large application should allow the possibility that a continuous distribution might be a good fit for observations.

If this larger study finds that the in-degrees and out-degrees actually follow specific probability distributions, this result could be used in several ways. For example, software engineers could take advantage of the fact that in-degrees and out-degrees are determined during the architectural design phase of software development. This phase occurs very early in either prescriptive or agile methodologies. A figure of merit could be developed for architecture by computing the distance between the distribution of in-degrees and of out-degrees for the specific architecture and the expected distribution. Perhaps a T-test or other statistical test could be employed. The

figure of merit actually would be the inverse of this distance. If the figure of merit was smaller than a predetermined standard, the architecture could be rejected.

The figure of merit could be used to compare two or more candidate architectures. The architecture with the larger figure of merit should be selected. If an application consisted of several large sub-systems, the figure of merit could be computed separately for each such sub-system. Sub-systems with the lowest figures of merit would then be given the most attention either in software construction or in testing.

Another potential use of a distribution of in-degrees and of out-degrees would be to study how an application changes during development or during maintenance. During development, the application architecture often changes significantly based on discoveries made during that development. Since both the pilot study reported here and the much larger study proposed use applications alter development is complete, it could be very interesting to observe how the distribution of in-degrees and of out-degrees converges towards the expected distribution during development. During maintenance, the distributions might diverge from the expected distribution as significant changes are made to the completed application. A figure of merit like that defined in the previous paragraph could be used to evaluate the need for refactoring of the application architecture as the architecture evolves through maintenance. If the figure of merit becomes too small, refactoring would be indicated.

All of these potential applications require that the distribution of in-degrees or of out-degrees be correlated with an application characteristic such as complexity. In-degree or out-degree has been used as a proxy for complexity in some projects and other studies. More work is needed to establish a strong link between this proxy and a

practical property of an application such as likelihood of failure in making a modification or performance.

Therefore, the work reported in this paper is the start of exciting research which could have significant, practical implications for the software industry. Further, results of this research could make a substantial contribution to the development of sorely needed theories to guide and explain the software development process.

REFERENCES

1. Chapine, N. (1979). A measure of software complexity. *National Computer Conference* , 995-1002.

2. *ERLANG.* (2007). Retrieved from http://www.erlang.se/doc/programming_rules.shtml on (11/05/07)

3. Gill, N. S., & Balkishan. (2008). Dependency and interaction oriented complexity metrics of component-based systems. *ACM SIGSOFT Software Engineering Notes , 32* (2), 1-5.

4. Halstead, M. H. (1977). *Elements of software science.* New York: Elsevier.

5. Henry, S., & Kafura, D. (1981). Software structure metrics based on information flow. *IEEE Transactions on Software Engineering , 7* (5), 510-518.

6. Khimta, S., Sandhu, P. S., & Brar, A. S. (2008). A Complexity Measure for JavaBean based Software Components. *World Academy of Science, Engineering and Technology , 42*, 449-452.

7. Lungu, M., & Lanza, M. (2007). Exploring Inter-Module Relationships in Evolving Software Systems. *11th European Conference on Software Maintenance and Reengineering (CSMR'07)* , 91-102.

8. Ma, Y., He, K., & Du, D. (2005). A Qualitative Method for Measuring the Structural Complexity of Software Systems Based on Complex Networks. *Proceedings of the 12th Asia-Pacific Software Engineering Conference* , 257 - 263 .

9. McCabe, T. J. (1976). Complexity Measure. *IEEE Transacions on Software Engineering , 2* (4), 308-320.

10. Phukan, A., Kalava, M., & Prabhu, V. (2005). Complexity metrics for manufacturing control architectures based on software and information flow. *Computers and Industrial Engineering , 49* (1), 1-20.

11. Schneberger, S. L., & McLean, E. R. (1996). Distributed computing environments: Effects on software maintenance difficulty. *Proceedings of the 29th Annual Hawaii International Conference on System Sciences* , 129-138.

12. Schneberger, S. L., & McLean, E. R. (2003). The complexity cross: implications for practice. *Communications of the ACM , 46* (9), 216 - 225.

13. Sharma, A., Kumar, R., & Grover, P. S. (2007). Empirical Evaluation and Critical Review of Complexity Metrics for Software Components. *Proceedings of the 6th WSEAS Int. Conf. on Software Engineering, Parallel and Distributed Systems, Corfu Island, Greece* , 16-19.

14. *Sourceforge*. (1999). Retrieved from www.sourceforge.net on (11/05/07).

15. Washizaki, H., Yamamoto, H., & Fukazawa, Y. (2003). A Metrics Suite for Measuring Reusability of Software Components. *Proceedings of the 9th International Symposium on Software Metrics* .

16. Yang, H., Tsujino, Y., & Tokura, N. (2007). The complexity measure of program based on the inter-module dependency. *Systems and Computers in Japan* , *22* (13), 10-19.