2010

Power law in XML schema metrics

Yanyin Zhang
University of Windsor

POWER LAW IN XML SCHEMA METRICS

by
Yanyin Zhang

A Thesis
Submitted to the Faculty of Graduate Studies
through School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science at the
University of Windsor

Windsor, Ontario, Canada
2010

Your file Votre référence
ISBN: 978-0-494-62742-6
Our file Notre référence
ISBN: 978-0-494-62742-6

Author's Declaration of Originality

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

Abstract

Software metrics are vital for the management of software development, especially when a new technology is being adopted and its best practice has yet to be established. XML Schema is a relatively new technology that has been widely adopted in software development. Despite its widespread usage in almost all different kinds of programming platforms, its usage patterns are not yet fully investigated. From two large sets of real XML Schemas, this thesis studies the distribution of some of the schema metrics and the structure of some large schemas. Elements in a schema are connected by their usage links. The interconnected elements can be viewed as a network of elements or a graph. This thesis also studies the structural properties of the network of the schema elements, including the scale free property, the connection of the graph, and its small world effect.

Acknowledgements

I wish to express my gratitude to Dr. Jianguo Lu, my supervisor, for his valuable assistance and support during my thesis work, and for his persistent guidance through out my graduate study at University of Windsor.

I also would like to express my appreciation to Dr. B. Boufama, Dr. Huiming Zhang and Dr. Dan Wu for having the patience to read drafts and offering me valuable comments and suggestions.

I am grateful to my friends who give me consistent help and insightful suggestions over the past three years in Windsor. The memory of time spent here would be kept in my mind forever.

Finally, I would like to thank my brothers who always give me support and encouragement over the past years.

Contents

Author's Declaration of Originality iii

Abstract iv

Acknowledgements v

List of Figures viii

List of Tables ix

1 Introduction 1

2 Review of Literature 4
2.1 XML Schema Metrics 4
2.2 Power Law Distribution 9
2.2.1 Power Law in Complex Networks 9
2.2.2 Power Law in Software 10
2.2.3 Small World Networks 13
2.3 Lognormal Distribution 14
2.3.1 Why Power Law and Lognormal Distribution? 15

3 Size Metrics 16
3.1 Data Collection 16
3.2 Power Law Distribution of Schema Metrics 17
3.2.1 Experiment Analysis 23
3.3 Lognormal Distribution 25
3.3.1 The Definition of Lognormal Distribution 26
3.3.2 Distribution Graph 29

4 Structure Metrics 34
4.1 Graph Model of XML Schema 34
4.1.1 Modeling XML Schema as Directed Acyclic Graph 34

4.1.2 Schema Length 38
4.2 In/Out Degree and Structure of Eight Schemas 44
4.2.1 Data Collection 44
4.2.2 In/Out Degree and Structure of Eight Schemas 45
4.3 Structure Metrics of Datasets 51
4.3.1 Data Collection 51
4.3.2 Number of Nodes and Edges 54
4.3.3 Small World 55

5 Conclusions **57**

Bibliography **59**

Appendices **66**

A Structure Graphs of Schemas **66**

Vita Auctoris **73**

List of Figures

3.1 The distribution of XSD file size 18
3.2 Histograms for dataset1 21
3.3 Histograms for dataset2 22
3.4 Histograms with Power Law/Lognormal distribution for dataset1 30
3.5 Histograms with Power Law/Lognormal distribution for dataset2 31

4.1 XML Schema example (top) and its graph 37
4.2 Histograms of in-degree for eight schemas 47
4.3 Histograms of out-degree for eight schemas 48
4.4 Structure of Schema CIM 49
4.5 Structure of Schema SIF 50
4.6 Histograms of structure metrics for two sub-datasets 52
4.7 Histograms of number of edges and number of nodes 54
4.8 The relationship between average path length and number of nodes 56

A.1 Structure of Schema eBay 67
A.2 Structure of Schema PDBML2 68
A.3 Structure of Schema PDBML1 69
A.4 Structure of Schema Purchase Order 70
A.5 Structure of Schema NIEM 71
A.6 Structure of Schema UN/CEFACT 72

List of Tables

2.1 Summary of existing studies on XML Schema metrics 5
2.2 Summary of Power Law Distribution in Complex Networks 10
2.3 Summary of Power Law Distribution in Software 12

3.1 Summary of Dataset 17
3.2 Summary of the statistics for size metrics of two datasets 32
3.3 Summary of size metrics R^2 value for two datasets 33

4.1 Summary of in/out degrees of eight schemas with Power Law fitting 46
4.2 Summary of the statistics for structure metrics of two subdatasets 53

Chapter 1

Introduction

Software metrics are vital for the management of software development, especially when a new technology is being adopted and its best practice has yet to be established. Chidamber and Kemerer, pioneers of the metrics of Objected-Oriented programming, said, "The need for (software) metrics is particularly acute when an organization is adopting a new technology for which established practices have yet to be developed" [8].

XML Schema is a relatively new technology that has been widely adopted in software development. Despite its widespread usage in almost all kinds of programming platforms, its usage patterns are not fully investigated yet.

The first question this study addresses is what a schema looks like in general. This can be divided into several sub questions, such as what is the size of schema in terms of its file size and how many elements it defines. Among the elements, how many simple types and complex types are defined? Since this study looked at a very large number of schemas, it focuses on the distributions of such statistics in order to show the state of the art as for what schemas really look like. It is observed that similar to other software components, the size of schema in various forms such as file size or elements follows power-law or log normal

distribution, i.e. a large percentage of the schemas are very small. Although it is commonly accepted that class sizes of Object-Oriented(OO) programs follow power-law or log normal distribution, possibly due to OO design principles, it is yet unknown as for the underlying mechanism for generating the large number of small schemas.

The second question this thesis intends to answer is what is the structure of a schema, especially the structure of a large schema. Elements in a schema are connected by their usage links. The interconnected elements can be viewed as a network of elements, or a graph. For this graph we would like to know:

In- and Out-degrees In general there are two kinds of networks. One is the random network where nodes are connected randomly. The other is the scale free network whose degrees follow a power-law distribution. We observe that without exception none of the schema forms random networks. Instead, their degrees follow power-law distribution. More specifically, two kinds of degrees are distinguished: in-degree and out-degree. In-degree corresponds to the number of times an element is used by other components, while out-degree corresponds to the number of subcomponents contained in the current element. It is observed that both in- and out-degrees distributions follow power-law, albeit their exponents are different. Most in-degrees have exponent around 1.5, while out-degrees have exponent around 1. This kind of asymmetry between in- and out-degrees are also observed in OO programs. However, in OO programs the exponents of out-degrees are in general larger than that of in-degrees.

Network connection We would like to know how densely the graph is connected. We observe that most of the graphs are sparse, in that the number of edges grows almost linearly with the number of nodes.

Small world effect In many natural and artificial networks, the average path length is

small. In other words, the path length grows logarithmically with the number of nodes. This is the so-called small world phenomenon. We observe that all the schemas exhibit such a phenomenon. For a schema with two thousands nodes, typically the path length is 4.

The implication of the in-degree distribution is that most of the elements have one or two degrees, while there are some elements having very large degrees. This reveals that elements are not adequately reused. In real practice, most elements are used only once while there are some "hub" elements that are used many times. For example, in our CIM schema, the number of reference time for most often used element is 200.

The implication of the out-degree distribution is that most of the elements contain only one sub-component. One example of such component relationship in CIM schema is:

```
<xs:element name="xCIM_PreconfiguredTransportAction"
            type="CIM_PreconfiguredTransportAction"/>
```

In addition to the general understanding of existing XML Schemas, our study finds applications in areas including:

- XML test data generation;
- Estimating the number of elements from the file size;
- Estimating the relation between number of elements and its referred numbers;

In this thesis, Chapter 2 gives an overview of the current work on schema metrics done by other authors. Also the related work in power law and lognormal distribution is discussed. Chapter 3 presents the power law distribution and the lognormal distributions of these schema metrics and a comparison between two of them are discussed. Chapter 4 describes the metrics for in-degree and out-degree for an individual Schema and structure metrics for two subdatasets. Also the small world effect is discussed. Chapter 5 provides the conclusions and proposes some suggestions for future work.

Chapter 2

Review of Literature

2.1 XML Schema Metrics

XML Schemas are widely used in software and web applications, which need to be properly designed so that they can be easily maintained. For this purpose, schema metrics need to be developed to enable quantification of schema size, complexity, quality and the other properties. In software engineering, XML Schema documents have a great impact on the overall quality of the software. Their metrics for predicting the quality and complexity of the software development process are important components [22]. Various metrics have been proposed in papers [5] [9] [18] [22] [23] [27] [37]. Table 2.1 is a summary of the studies on XML Schema metrics.

XML Schema metrics can be classified into two categories, i.e., size and structure of the schemas. Size metrics include line of code, number of elements, number of certain kinds of elements as well as the file size in KB. Structure metrics include number of nodes, number of edges, average path length, diameter, in-degree and out-degree. In this thesis, we describe these metrics in Chapter 3 and Chapter 4.

Study	Size							Structure metrics					
Paper	# of Ele.	# of Attr.	LOC	# of Anno.	# of CT.	# of ST.	File Size in KB	Depth	In-degree	Out-degree	MCC	Complexity new definition	Note
Klettke et al. 2002	yes	yes	no	no	no	no	no	yes	yes	yes	yes	yes	DTD
Lammel et al. 2005	yes	yes	yes	yes	yes	yes	yes	yes	no	no	yes	no	XML Schema
Visser 2006	yes	no	no	no	no	no	no	no	yes	yes	no	no	XML Schema
McDowell et al. 2004	yes	yes	no	yes	yes	yes	no	no	yes	yes	no	yes	XML Schema
Mignet et al. 2003	yes	yes	no	yes	no	no	yes	yes	no	yes	no	no	XML Web
Our study	yes	yes	yes	yes	yes	yes	yes	no	yes	yes	no	no	XML Schema

Table 2.1:
Summary of existing studies on XML Schema metrics. Note:CT–ComplexType; ST–SimpleType

In [19] Lammel et al.'s work is the most comprehensive study on the metrics of XML schemas. It tries to answer the following three questions for schemas used in real applications: a) how large and complex real-world XML schema are; b) what schema feature are used in practice; c) which of the known styles of schema organization are used in practice. The definition of depth is based on content model. A code-oriented depth basically measures the nesting of constructs for content model. Different datatype and element declarations contribute to corresponding value for total depth value. The instance-oriented depth is defined as the longest path from the root to a leaf in an XML tree which represents the schema. Also it draws the histogram of depths to show that depth may follow power law although they did not mention that. The result is not very conclusive due to small data set (63 schema projects).

They draw the co-relation between various size metrics, showing the loc grows linearly with file size and node number. Also the MCC has a linear relation with the node number.

The in/out degree distribution is not analyzed in this paper.

In [23], Mignet et al. studied the XML web and analyzed the depth and element/attribute fan-out to describe the structural property for XML files. Modeling XML file as tree representation, the depth of a certain node is defined as the longest distance from the root node to this node in XML documents. The element/attribute out is defined as the number of children per element/attribute. From the study, it shows that both depth and element/attribute out degree follow power law distribution(in degree is not analyzed). And 99% of the documents have fewer than 8 in their depth value. Besides, in XML Web, the out-degree of an XML document is defined as the number of attributes nodes labeled href, xmlhref, or xlink:href in that document which is represented as XML Web graph and it follows power law distribution with exponent 1.8. The experiments in this paper is based on total 200,000

XML documents on the web.

In [22], McDowell et al. discussed eleven metrics to measure the quality and complexity of XML Schema. The metrics discussed in this paper includes number of ComplexType declarations, SimpleType declarations, Annotations, Derived ComplexType, average number of Attributes per ComplexType declaration, number of Global Type declarations, number of Global Type references, number of Unbounded Elements, average Bounded Element Multiplicity Size, average number of Restrictions per SimpleType declaration and element fanning. For element fanning, it includes the in- and out- degree (The degree distribution with its regularity is not analyzed in this paper). Based on these eleven metrics, a quality index and complexity index were developed to evaluate the quality and complexity of schema file. The experiment in this paper is only based on one schema file. The formula for index is defined as:

Complexity Index=(number of unbounded elements)*5+(element fanning)*3+(number of complexType declarations)+(number of simpleType declarations)+(number of attributes per complexType declarations)

In [37], Visser proposed another suite of eleven structure metrics for the XML Schemas based on the directed graph representation of schema structure. The metrics include tree impurity, in/out degree, instability based on fan, afferent coupling, internal edges, coherence, normalized count of modules and count of nodes per module. The data in this paper is collected by a tool developed by the author himself and up to nearly two megabytes in ascii file size with nine schemas. The regularity of metrics is not analyzed in this paper.

In [18], Klettke et al proposed a set of five metrics to measure the quality, i.e., reusability and maintainability of XML schemas. The metrics including size, in/out degree, depth, complexity, and the rationale for using these metrics are described. Rather than doing a

statistical study of the metrics on a large set of schemas, this paper focuses on the definition of five metrics and their relations to usability and maintainability. The experiments in this paper are on several small DTDs. The depth and in/out degree is defined on the tree representation of Schema. The in/out degree in this paper is called fan-in/fan-out.

Complexity is an important metric defined in XML Schema to measure the complexity of the program. MCC is first proposed. It is a measure of control flow that links the number of logical branches or decisions in a module to the difficulty of programming[21]. It is also applied in XML Schema. In [19], MCC is redefined, and some important 'decision' nodes are considered, including the node of element references and the multiplicity of root element declarations. Summing up all decisions for each schema results in the MCC value.

Other than MCC, [32] mainly focused on different ways of determining the complexity of XML documents. These are based on different syntactic and structural aspects to decrease the complexity of XML documents and improve their reusability and maintainability. The metric value was evaluated on the basis of the internal complexities of major building components. Meanwhile, a Weight Allocation Algorithm is proposed, which assigns weights to the elements of XML trees according to their place from the root node (element) to evaluate their complexity. This algorithm provides means of gauging the quality and comprehensibility of XML documents. The limitation of this method is that the weight values are assigned randomly without validation. Consequently, the complexity value for schema can only reflect the structure rather than the content.

Conversely, in [5], the author suggests that the complexity of a given schema document closely depends on the internal complexities of its building components. This means that each component contributes its complexity values, on the basis of its design architectures, to the schema document's overall complexity. Also, a Weight Allocation Algorithm

is proposed and a weight value for each component that reflects the complexity of each component complexity degree.

2.2 Power Law Distribution

In this section, we introduce the power law distributions observed in complex networks. In computer science, it is mainly observed in internet-related webs and software programming.

2.2.1 Power Law in Complex Networks

Many real systems have been described as complex networks, where nodes represent specific parts of the system and connections represent relationships among them. Examples of such networks come from different areas. For example, networks like the Internet [15], the World Wide Web [7], and the North American Power Grid [3] have all been analyzed as complex networks. Many other examples are seen in the social sciences, such as networks of friendships between individuals, have also been modeled and analyzed as complex networks [29]. Similarly, biological systems such as neural and metabolic networks and protein interaction webs have been modeled as complex networks [17].

Complex systems can be represented as graphs. With graphs, statistical properties in systems can be displayed visually. More precisely, almost all of complex systems can be found to show a high degree of self-organization into a scale-free state [4]. In practice, these systems are modeled as graphs where the vertices represent the entities of the system, and the edges represent the relationship between them. The distribution of edges connected to vertices follows a power law. This can be seen in the internet topology [15] and the webpage links in World Wide Web [1]. Power law is not a characteristic that requires a

graph in order to be identified. It is also found in the distribution of features related to a single properties. For example, as stated in the previous chapter, the distribution of words in a corpus of documents and the population size of US cities [16] all follow a power law. The following is a summary of studies in complex networks with the k value and the general length.

Table 2.2: Summary of Power Law Distribution in Complex Networks

paper	Networks	k	length
Watts and Strogatz, 1998	Movie Actors	2.3	3.48
Newman, 2001	Phys. Coauthorship	–	6.19
Newman, 2001	Biol. Coauthorship	N/A	4.92
Broder et al., 2000	WWW Altavista	2.1/2.7	16.18
Render, 1998	Paper Citations	3.0/	–
Faloutsos, 1999	Internet	2.5	3.31
Amaral et al., 2000	Power Grid	–	18.99
Jeong et al., 2001	Protein Interaction	2.4	6.8

2.2.2 Power Law in Software

In the area of information and computer science, power law distributions have been observed in many areas including software metrics, network and social networks, web mining, IR etc.

In [31], Potanin et al. studied the object graphs of java ArgoUML, Java Forte, Java Jinsight, Java Satin GCC and SmallTalk programs. The node of the object graph represents an object instance in the program and link is the relationship between the object instances. The result showed that these systems are scale-free networks and the power law distribution is observed for the incoming and outgoing links on the object graph. The average k value

of fitted line for incoming and outgoing links are close to 2.5 and 3 respectively.

In [35], Valverde et al. studied the emergence of scaling in software architecture graphs constructed from the component in JDK1.2 where a class corresponds to a node and relationships exists between classes corresponds to edge. The power law is found for the degree distribution of the two largest components with the gradient between $2.5 - 2.65$. Also the small world phenomenon was found with the average distance of 6.39 and 6.91 between any two nodes on the graph.

In [28], Myers studied some open source system of VTK visualization library, Digital Material (DM), AbiWord word processing program, Linux operating system, MySQL relational database, and XMMS multimedia system with their class collaboration network. Class collaboration is defined to include the interaction of classes both through inheritance and aggregation. On the graph, the node represents the class and the edge represent the relationship between two classes. All collaboration networks studied exhibit scale-free (power law) and/or heavy-tailed degree distributions. The k value for in/out degree distribution of all systems is between $1.9 - 3.1$. For most of systems, this value is around 2.5.

In [39], Wheeldon et al. studied the power law distribution of class properties and relationships in JDK, Tomcat and Ant systems. The relationships mainly are focused on the number of methods, number of constructs, and number of fields as well as the coupling and inheritance relationship between them. The k value of power law distribution for those properties are between $0.906 - 3.663$ and the average is close to 1.1. The corresponding R-Square value is between $0.787 - 0.959$ and the average is 0.95. An extension of this paper is [6], where some structural metrics of different Java programs display power law distributions. The metrics studied in this paper includes number of methods, fields, constructors, subclasses, implemented interfaces, interface implementations, etc. The power law k value

for these metrics are around $0.91 - 3.29$ and the average value is around 1.8.

Similar work was done in [20], for which the relationship at class and function level in different systems show power-law distribution. All of these works model the system as a graph and analyze the feature of edge distributions in the graph. The in-degree and out-degree of the graph are considered and the k value is between $1.22 - 3.5$ and the average is close to 2. The corresponding R-Square value is above 0.95.

A different approach was taken in [41]. The total number of the distribution of lexical tokens in 24 real world systems written by Java, C++ and C language exhibits Zipf's law and the growth of the distinct tokens follows Heap's law. Table 2.3 summarizes power-law distributions in software systems.

Table 2.3: Summary of Power Law Distribution in Software

Paper	Dataset	Graph Model	Study Focus	k	R^2
Valverde et al. 2002	JDK1.2	Archi. graph	In/Out dehree	2.5-2.65	N/A
Wheeldon et al. 2003	JDK/TOMCAT/Ant	Class Diagram	OO coupling type	0.9-3.66	0.79–0.96
Myers 2003	Open source sys.	Class diagram	In/Out degree	1.9-3.1	N/A
Potanin et al. 2005	Java system	Object graph	In/Out degree	2-3	N/A
Baxter et. al. 2007	java/apache	Class diagram	OO metrics	0.91-3.29	N/A
Louridas et al. 2008	Java System	Module Graph	In/Out degree	1.22-3.50	0.58–0.99
Theoharis et al. 2008	Semantic Web	Class Diagram	properties	0.5-1.23	N/A
Zhang 2009	Java, C/C++	N/A	Lexical tokens	1.16-1.30	0.92-0.98

Finding the statistical properties in software is a popular topic among computer scientists. In recent years, interest in applying complex network theories and models in order to represent large software systems has emerged. Indeed, many software systems have reached such huge dimensions that it seems reasonable to treat them as complex networks [34]. Software is built up of many interacting subsystems and components at different levels of granularity (functions, classes, interfaces, libraries, source files, packages, etc.), and the various kinds of interactions among those components can be used to define graphs to

form a description of a system. Moreover, some entities can be studied to look for a certain behavior with its edge distribution.

2.2.3 Small World Networks

The small world effect has a great impact and implication on real networks. Especially it plays a fundamental role on the dynamics of network, because it affects the spread speed of network itself. For example, in [38], the small world model explains why the diameter of real graphs can remain very small when the number of vertices increases. In addition to "six degrees of separation", various numbers have emerged associated with many networks. In [4], it is estimated that any two webpages are at most 19 clicks away from one another. In [7], the average path length is about 16 for crawlers travel on a webgraph of 200 million nodes and 1.5 billion links.

In [38], a new *small world network* model is introduced and it is characterized by a small minimum length path but displaying a large clustering coefficient. This model first starts on lattices of any dimension or topology. If we take a one-dimensional lattice of L vertices on a ring, and join each vertex to its k neighbors with total Lk edges, the small world model is then created by taking a small fraction of the edges in this graph and "rewiring" them. The rewiring procedure involves going through each edge in turn and, with probability p, moving one end of that edge to a new location chosen uniformly at random from the ring, except that no double edges or self-edges are ever created.

The small world model proposed by [38] is able to get both clustering and small world effect. But still some problems exists. The most distinct feature between real networks and the early graph model proposed by [14] is about the shape of the degree distribution. While theoretical model suggests the Poisson distribution is observed, many real networks in fact

display heavy tail power law distribution.

2.3 Lognormal Distribution

Lognormal distributions are mostly analyzed in file size, either in web files or common files in any server. In [13] [12], the authors analyze the root cause of file size in lognormal distribution. In practice, the common methods of producing files include copying, translating or editing an existing file. In this way, the size of new file is proportional to old files at a fraction rate. At log-log scale, this multiplicative process is the reason for displaying lognormal distribution. In [33], the power law is unable to best fit the head of distributions for mobile calling data flow, but lognormal fits it well. In [42], the class size in large Java programs is better fitted with lognormal distribution. In [3], the skew of the head of the distribution graph for US western city power transmission lines and airport connections throughout the world indicates that they could be suitably fitted with power law distribution, however, lognormal distribution fits it very well.

In [42] [43], the author conducted an empirical study of class sizes for large Java Systems. The LOC is considered as the program size and it exhibits the Lognormal distribution for different Java systems. For all systems, the average 57.04% of the classes are smaller than 65 LOC and 75.09% of the classes are smaller than 129 LOC. This phenomenon is called the small class phenomenon. The possible causes and implications of the small class phenomenon lie in the adoption of OO decomposition and reuse techniques. A natural consequence of good OO development is that there are a large number of small classes.

The lognormal distribution is a parabola in log-log scales. Sometimes it may seem like a power law if appropriately fitted. The difference between them lies in the fact that power law is a scale-free network while lognormal distribution is single-scale because of

the distortion of the head in the distribution line [3].

2.3.1 Why Power Law and Lognormal Distribution?

Power law and lognormal distribution are two common models to fit the empirical data. The two are intrinsically connected. Very similar basic generative models can lead to either power law or lognormal distributions depending on trivial variations. In [25], the author explains some basic generative models that can lead to both power law and lognormal distribution, then further points out that small variations can change the result from one to the other. These models include: power law via preferential attachment; power laws via optimization; multiplicative processes; monkeys typing randomly; and double Pareto distribution [26] combining both power law and lognormal distribution. In [30], the authors provide a full explanation for the mechanisms of power law distributions. These mechanisms are: combination of exponentials; inverse of quantities; random walks; the Yule process; phase transition and critical phenomena and self-organized criticality.

From the above we can see that there is no absolute difference between generating power law and lognormal distribution. The internal property of empirical data itself decides which model is best suited. In this thesis, both power law and lognormal distribution are applied in analyzing the metrics for total property of datasets and individual schema files. We will determine which distribution can better fit the metrics resuling in this thesis.

Chapter 3

Size Metrics

In this chapter, we first give an overview of the data collected for our study. We then analyze the power law distribution of size metrics of those data.

3.1 Data Collection

For our experiments, there are two sets of XML Schemas. The first dataset contains 10,879 XSD files which were collected in 2007 and 9606 of which are valid files. Here the word 'valid' means that the file can be parsed by a DOM XML parser. The total size of the dataset is 610MB. The average size for each XSD file is 58.6KB. The second dataset was collected in 2009 and contains 18,046 XSD files with 14,454 valid files. The total size of the second data collection is 1.31GB. The average file size is 68.6KB. The details of the data are tabulated in Table3.1. During the data collection, we encountered the same problem as documented by other researchers, i.e., there are a significant number of errors in the collected data. These errors in Schema files include: bad encoding, missing end tags, missing elements, unescaped special characters, incorrect use of namespaces, incomplete

file structure, etc. The DOM XML parser detects file errors and these files are discarded when we implement analysis.

Table 3.1: Summary of Dataset

Name	total # of files	# of valid files	Size	Avg. file size
Dataset1	10879	9606	610M	58.6KB
Dataset2	18,046	14,454	1.31G	68.6KB

3.2 Power Law Distribution of Schema Metrics

A power law states that small values are extremely common, whereas large values are extremely rare. The definition of power law is expressed in following form:

$$p(x) \propto x^{-k} \tag{3.1}$$

where p(x) is the probability that x occurs.

Power law distribution is usually plotted with double logarithmic axes because we cannot identify well the characteristics of the distribution with linear axes. Alternatively, the equation (3.1) can be transformed into the equation (3.2) and we can see that the plotted values form a straight line whose gradient is $-k$.

$$logp(x) \propto -klog(x) \tag{3.2}$$

The illustration of power law distribution is seen in Figure 3.1. Due to the heavy-tailed nature of file size, as shown in Figure 3.1 A, we perform a logarithmic transformation of both axes to view data more clearly, as shown in Figure 3.1 in B, the file size distribution

on such log-log plots in which the graph approximately exhibits a straight line. But the tail does not fit a straight line very well. In order to smooth such noise tail, we use the method to increase bin size exponentially, see panel C.

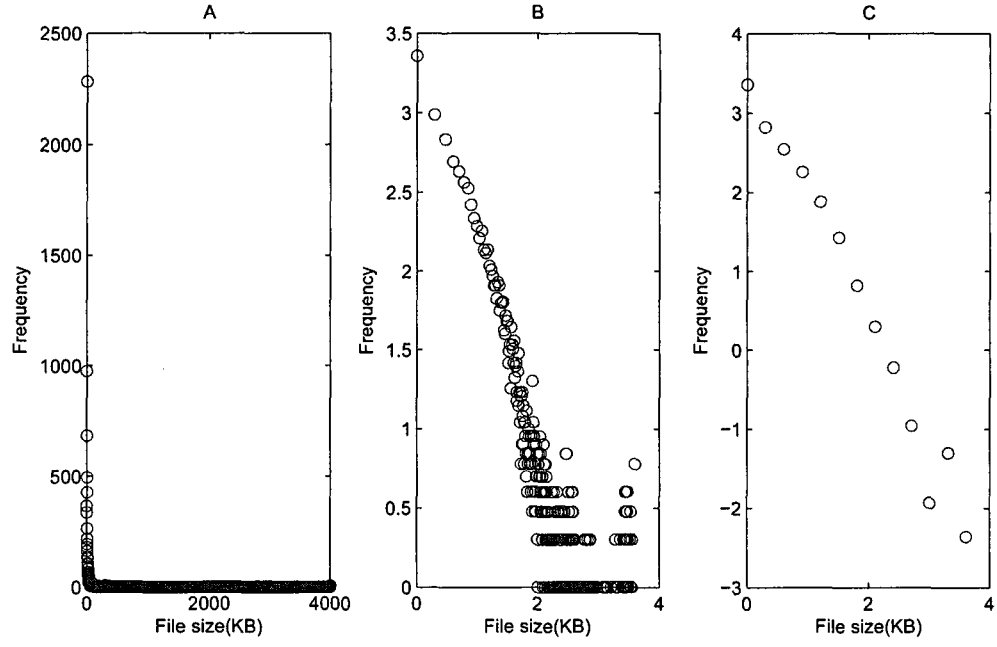

Figure 3.1: The distribution of XSD file size

Let random variable X be the frequency of a metric value we are interested in. Given a metric value x, the fraction of the schemas that has value x is denoted as $p(x)$, and is defined as

$$p(x) = Pr(X = x) = \frac{number\ of\ schemas\ having\ value\ x}{total\ number\ of\ schemas} \tag{3.3}$$

The metric value x can be the file size, the number of elements in a schema etc.

What we are interested in is what are the best functions to describe $p(x)$ for various metrics. It was observed that many metrics in software systems follow power-law distribution.

We will study:

- What metrics follow power-law distribution?
- If they follow power-law distribution, how well do they fit the power-law, and what are the exponent?
- What are the alternative distributions that may fit the distributions better?

Cumulative Distribution

In addition to the method of increasing data bin size exponentially, another method to reduce the noise at the tail of data plotted in log-log scale is to draw the cumulative distribution $P(x)$, which is defined as

$$P(x) = \sum_{j=x}^{\infty} p(x) \tag{3.4}$$

Note that $P(1) = 1$ and

$$P(x) \propto \sum_{j=x}^{\infty} \propto x^{-(k-1)} \tag{3.5}$$

While some studies [10] [30] use cumulative distribution, our study will use the method of increasing bin-size exponentially to remove noise data.

In this thesis, we examine the different metrics in size, including:

- $p_{FileSize}(x)$: File Size in KB
- $p_{LOC}(x)$: Line of Code
- $p_{ElementNumber}(x)$: Element Number
- $p_{ComplexType}(x)$: ComplexType Declaration Number
- $p_{SimpleType}(x)$: SimpleType Declaration Number
- $p_{annotation}(x)$: Annotation Number

To evaluate the goodness of fitting effect, we use the statistic of R-Square to measure how successful the fit is in explaining the variation of the data. R-Square can take on any value between 0 and 1, with a value closer to 1 indicating that the fitting is better. In our analysis, we get this value for each diagram fitting from matlab curve fitting directly.

The log-log diagram of data for each metric was plotted as shown in Figure 3.2 and Figure 3.3. The power law fitting is in Figure 3.4 and Figure 3.5(red-line). Table 3.2 is the summary of k-value and R-Square value for power law fitting.

These metrics are obtained from schema itself without graph modeling. None of these metrics reveal any schema-specific properties, however they provide general measures for understanding the overall size of datasets.

[2] [4] summarize that power law is caused by two reasons in real systems networks. One is the growth in which new nodes appear at random times. The other one lies in preferential attachment in which a new node connects to an existing node with probability proportional to the number of connections already at that node. In other words, well-connected nodes tend to attract more connections than poorly-connected nodes. Any network with these two conditions will tend to exhibit a scale-free state and has a power-law distribution. The fact that so many different kinds of networks are scale-free can be explained by growth and preferential attachment.

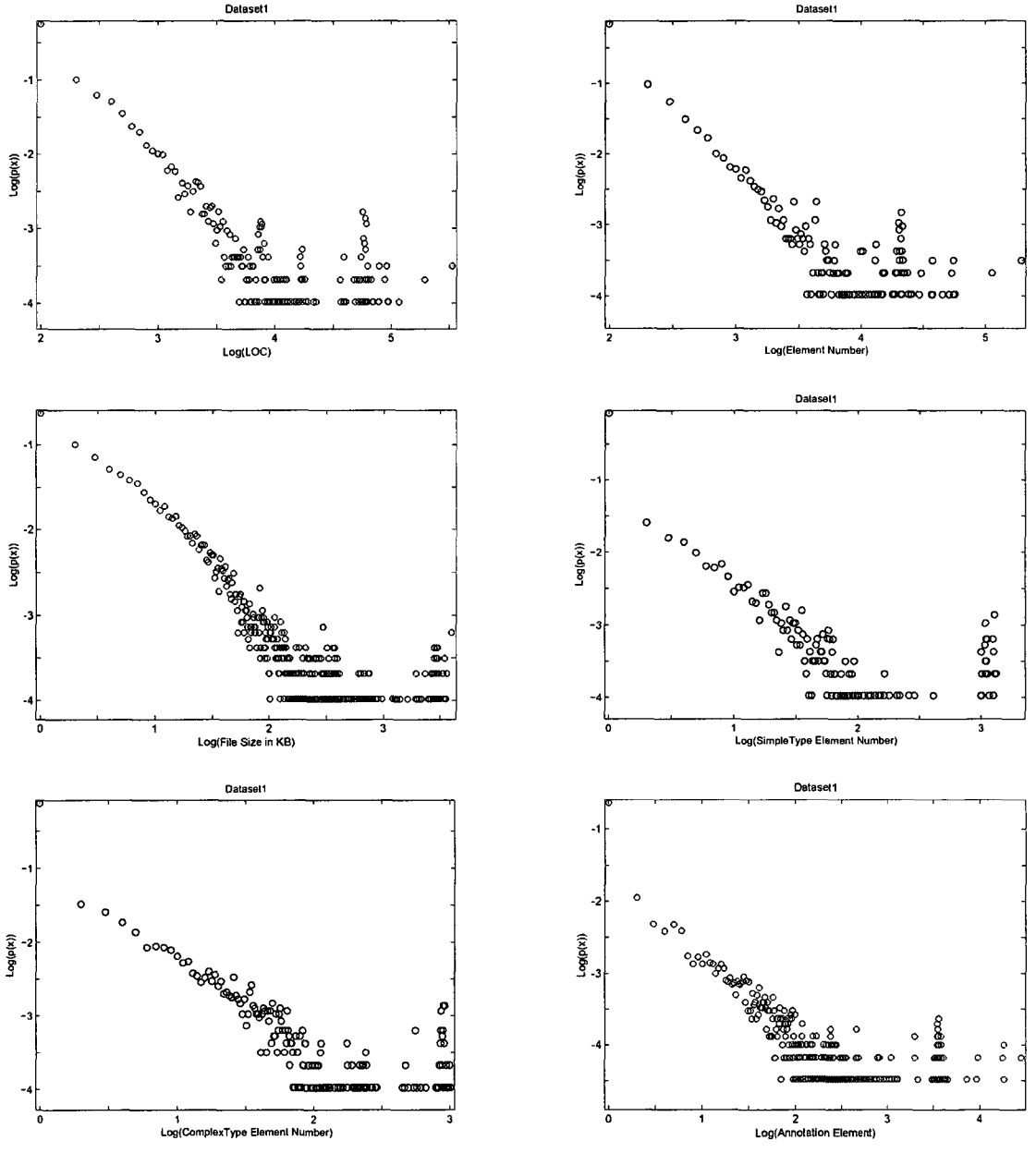

Figure 3.2:
Histograms in log-log scale for dataset1

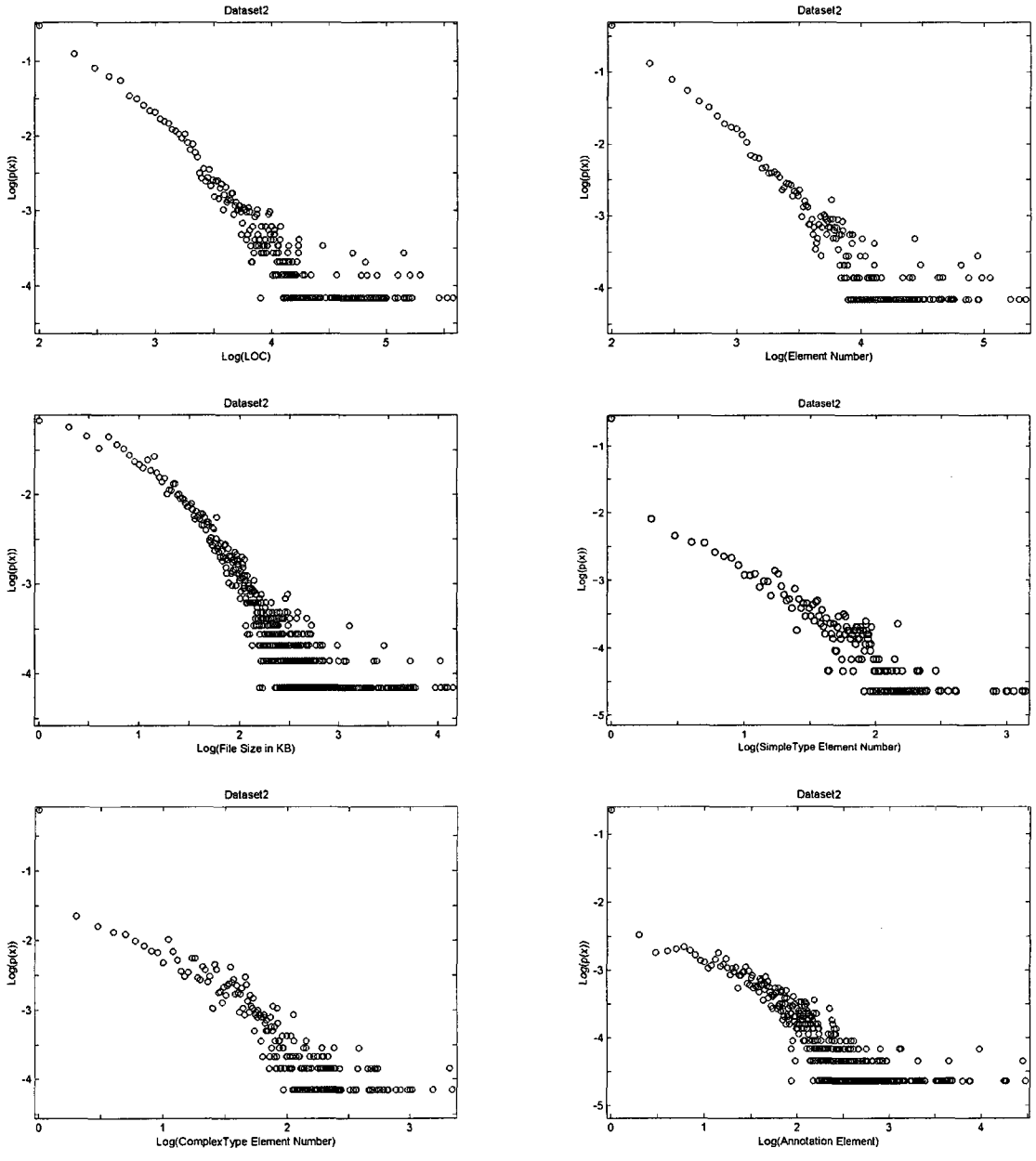

Figure 3.3:
Histograms in log-log scale for dataset2

3.2.1 Experiment Analysis

In this section, we give some analysis for each size metric.

Lines of Code

The first property we studied was the distribution of the size of schemas, computed as Lines of Code (LOC). The LOC here does not include blank lines, but includes comment line in the XSD file. For dataset1, The maximum LOC of a schema in this dataset is 337,527 and the average number of LOC for a schema is 1,407. For dataset2, The maximum LOC is a schema with 382,327 lines and the average number is 1,643. For its power law distribution, the k and R^2 is 1.19 and 0.80 for dataset1, and 1.17, 0.83 for dataset2.

LOC is often used as a size metric to evaluate XML Schema. In software system, LOC is also used to measure the program size. In [42], class size is measured in the form of LOC for some Java systems and their size distribution exhibits lognormal distribution.

Number of Elements

The second property we studied is the number of elements in each XML Schema. In a XML DOM tree, each node represents an element in schema and we count the total number of nodes on DOM tree as the number of element in schema. This number directly indicates the size of each schema. It includes all kinds of elements, including complexType elements, simpleType elements, documentation, etc. in schema. In dataset1, the schema with the maximum number of elements has 192,880, while the minimum number is 1. For dataset2, the maximum and minimum are 218,472 and 1, respectively. The average number of elements for each dataset is 683 and 959. For the power law distribution of this metric, the k and R^2 is 1.37, 0.91 for dataset1, and 1.37, 0.88 for dataset2. From the diagram of

power law distribution for this metric, we can see that some small value number at the head of the curve is not close the fitted straight line and this affects the R^2 value.

File Size in KB

The third property we studied is file size in KB. In both datasets, the majority schemas size is less than 1MB. There are only 1ess than 200 schemas with their size greater than 1MB. The largest size of a schema is $12.1MB$ in dataset1 and $13.8MB$ in dataset2. The average size is $58.6KB$ and $68.6KB$ for two datasets. We can see from the diagram of its power law distribution, the k and R^2 is 1.60, 0.97 for dataset1, and 1.46, 0.96 for dataset2.

Number of ComplexType Declarations

The fourth property we studied is the the number of ComplexTtype declarations (including the number of derived ComplexTypes from other ComplexTypes by either expanding or restricting the definitions of their parent types). XML Schema allows the definition of ComplexTypes. ComplexTypes allow elements in their contents and can have attributes. They are used to define elements with child elements in their content models. As a result, the presence of more ComplexTypes usually indicates more complex XML schema structures. From the result of experiments, we find that nearly 56.8% of files in dataset1 and 69% of files in dataset2 have no ComplexType declarations. The maximum number of ComplexType declaration in a schema is 1014 and 2222 for dataset1 and dataset2, respectively. This number is far less than the element number. This indicates that a large XSD file does not necessarily have very complex structures. For its power law distribution, the k and R^2 is 1.63, 0.96 for dataset1 and 1.84, 0.96 for dataset2.

Number of SimpleType Declarations

The fifth property we studied is the number of SimpleType declarations. The SimpleType declaration defines a simple type and specifies the constraints and information about the values of attributes or text-only elements. In dataset1, the majority of XSD files, i.e. nearly 80% of files, have no SimpleType declarations. In dataset2, this fraction is 84%. For its power law distribution, the k and R^2 is 1.85 and 0.90 for dataset1, and 1.94, 0.95 for dataset2.

Number of Annotations

The annotation element in XML Schema allows documentation for the benefit of both human readers and applications. The element may contain document elements for human readers. Having more annotation elements present in the XML Schema document usually implies that the XML Schema is better documented. Therefore, the overall quality of the XML Schema is better. Compared to SimpleType and ComplexType declaration number, the annotation element number is much more than two of them. For annotation element, the maximum number of a schema is 28051 in dataset1 and 29891 in dataset2. For its power law distribution, the k and R^2 is 1.60 and 0.97 for dataset1, and 1.57, 0.96 for dataste2.

3.3 Lognormal Distribution

In this section, we introduce the lognormal distribution and analyze this distribution for size metrics.

3.3.1 The Definition of Lognormal Distribution

When the logarithm of a variable x follows a normal distribution, the distribution of x is called a lognormal distribution, and the probability density function of the lognormal distribution is given by

$$f(x) = \frac{1}{\sigma x\sqrt{2\pi}} exp(\frac{-(\log(x)-\mu)^2}{2\sigma^2}), \quad x > 0; \ \sigma > 0 \tag{3.6}$$

where μ and σ^2 are the mean and variance of the associated normal distribution, and exp(x) is the exponential function of x, e^x. μ is also called the scale parameter and σ the shape parameter. The corresponding complementary cumulative distribution function for a lognormal distribution is in the form of

$$Pr[X \geq x] = \int_{z=x}^{\infty} \frac{1}{\sigma z\sqrt{2\pi}} exp(\frac{-(\log(z)-\mu)^2}{2\sigma^2})dz, \quad x > 0; \ \sigma > 0 \tag{3.7}$$

A lognormal distribution has finite mean and variance. Compared to normal distribution, the lognormal distribution is skewed, with mean $e^{\mu+\frac{1}{2}\sigma^2}$, median e^{μ}, and mode $e^{\mu-\sigma^2}$.

Despite its finite moments, the lognormal distribution is somewhat similar in shape to power law. For example, if x has a lognormal distribution, then in a log-log plot of the cumulative density function or complementary cumulative distribution function, its behavior will appear to be nearly a straight line for a large portion of the distribution. Taking the natural logarithm on both sides of the equation 3.6, we get:

$$\ln f(x) = -\ln(x) - \ln(\sigma) - 1/2\ln(2\pi) - (\ln(x)-\mu)^2/2\sigma^2 \tag{3.8}$$

which has the same form as the quadratic equation:

$$y = \ln f(x) = \beta_0 + \beta_1 \ln(x) + \beta_2 (\ln(x))^2 \tag{3.9}$$

where $\beta_1 = \mu/\sigma^2 - 1$ and $\beta_2 = -1/(2\sigma^2)$. For the above equation, the first term is a constant, the second represents a straight line, and the last term shows a parabolic line. Thus, a lognormal distribution can be seen as a quadratic function curve on a log-log plot. If σ is large enough, the effect of the last term is weakened, and the distribution approaches a power law. If the logarithm values of empirical data result in a quadratic function curve, we can reason that the data is lognormally distributed [43]. Lognormal distribution is widely used in file size distribution analysis.

Power law distribution has the property of being scale-free [35] [30] [15]. It is better that the data distribution in the above equation is referred to as a Gaussian distribution or quadratic distribution. The difference between the two is that the Gaussian distribution is not scale-free, but rather is single-scale compared to power law distribution [3]. In power law distribution, the fitted straight line decays at a fixed proportion from the head to the tail. However, in lognormal distributions, while there is a distortion at the head, its tail decays in the same way as a power law in a straight line.

The lognormal distribution can be parameterized by μ and σ^2, which we can estimate from the expectation *E(X)* and variance *V(X)* of an actual distribution. From the properties, *E(X)* and *V(X)* of the lognormal distribution are given as follows:

$$E(X) = \exp(\mu + \frac{\sigma^2}{2}), \tag{3.10}$$

$$V(X) = \exp(2\mu + 2\sigma^2) - \exp(2\mu + \sigma^2). \tag{3.11}$$

Then μ and σ^2 are evaluated as follows:

$$\sigma^2 = \ln(\frac{V(X)}{E(X)^2} + 1), \tag{3.12}$$

$$\mu = \ln(E(X)) - \frac{\sigma^2}{2}. \tag{3.13}$$

The lognormal distribution is a parabola in log-log scales, but may seem like a power law, if appropriately fitted. The most recognized reason for producing a lognormal distribution is the multiplicative process. The study undertaken in [13] indicates that the most common methods of producing files include copying, translating or editing. These methods are all based on a existing file which is used to produce a new file. In this way, the size of the new file is equal to the size of existing files plus or minus a random fraction. The size of the new file can be expressed by the existing file size multiplied by a certain value. The mathematical explanation is provided below.

In [25], the multiplicative processes are explained. For example, in biology, the multiplicative process is used to describe the growth of an organism. Suppose the size of an organism starts with X_0. At each step p, the size of organism may increase or decrease at a random fraction of F_p, so that

$$X_p = F_p X_{p-1}. \tag{3.14}$$

The idea behind this is that the random growth of an organism is expected as a percentage of its current weight, and is not related to its current actual size. If the F_q, $1 <= q <= p$ all

exhibit lognormal distributions, then each F_p, inductively, exihibits lognormal distribution, since the product of lognormal distributions is, again, lognormal [26].

3.3.2 Distribution Graph

In our study, we analyze the size metrics and plot the data with the fitting of lognormal distribution. Figure 3.4 and Figure 3.5 are diagrams of size metrics with fitting of lognormal distribution (blue line). The result shows that lognormal fitting appears better than power law fitting. To plot the diagram of each metric, the bin size of data increases in 2^n, $n = 0, 1, 2, ...$ Table 3.3 is the summary of R^2 of sizes metrics with lognormal fitting along with a comparison of that value with power law fitting.

Similar to the study in [13], we analyze that some files in our dataset are related to each other. Some files are produced based on the other files through re-editing or through the addition of new contents. This can explain the lognormal distribution of file size that has been illustrated in other papers. Other metrics, such as the number of elements, LOC, and element number, are related to the file size. The larger the file size in KB, the bigger the metrics value.

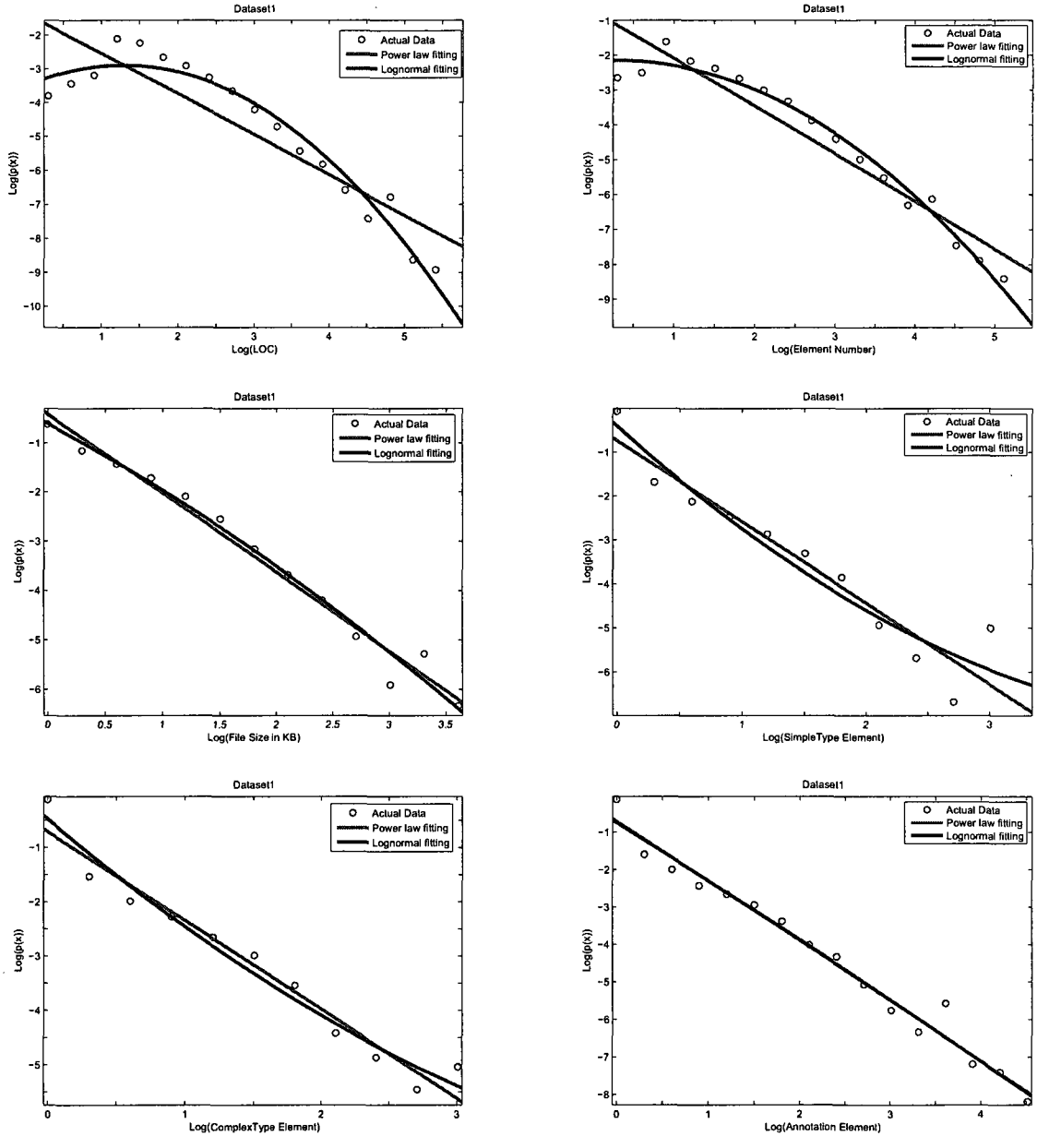

Figure 3.4:
Histograms in log-log scale with Power Law/Lognormal fitting for dataset1. For all diagrams, the bin size of data increases in 2^n, n=0,1,2....

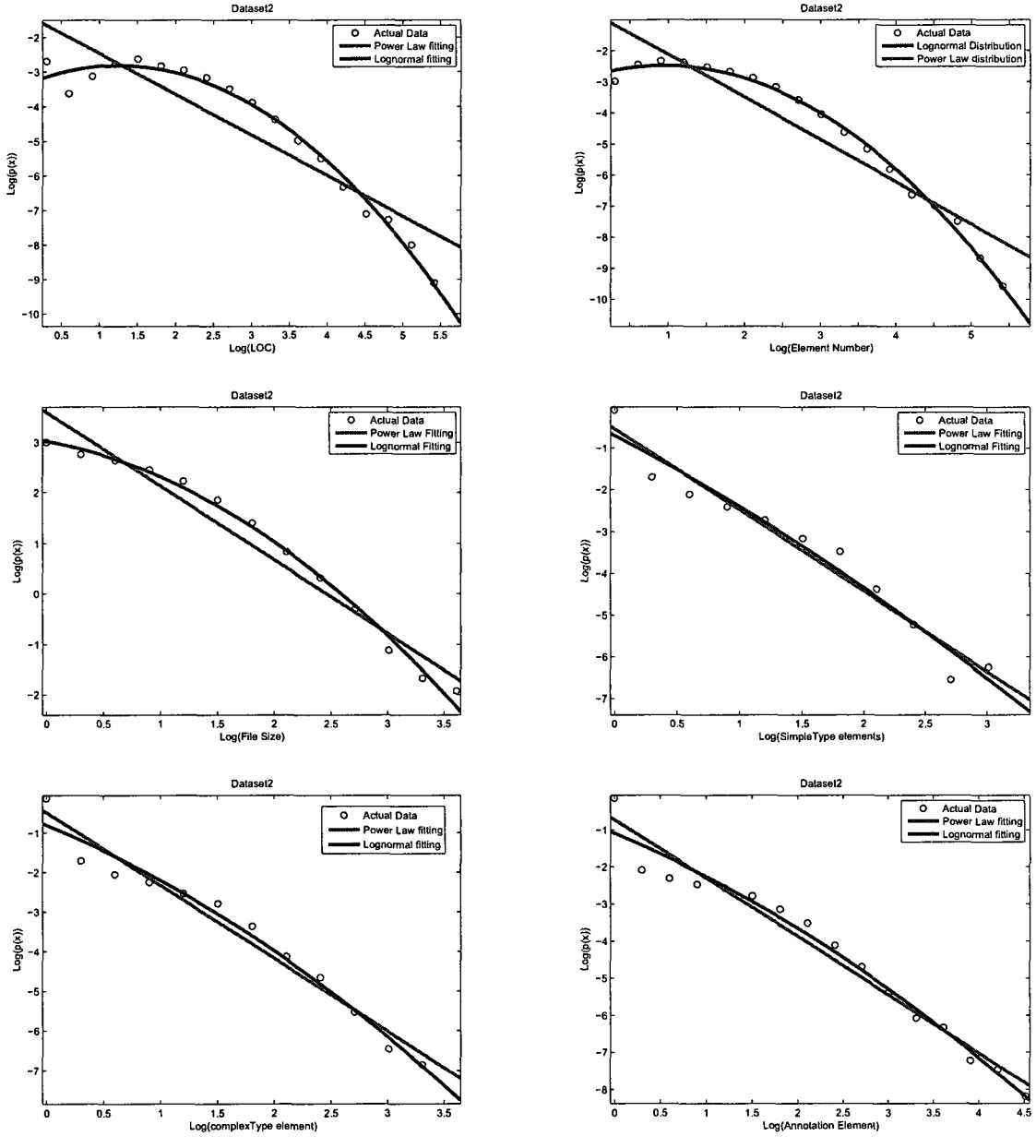

Figure 3.5:
Histograms in log-log scale with Power Law/Lognormal fitting for dataset2. For all diagrams, the bin size of data increases in 2^n, n=0,1,2....

Table 3.2: Summary of the statistics for size metrics of two datasets

	Dataset1					Dataset2				
Name	Maximum	Minimum	Mean	k	R^2	Maximum	Minimum	Mean	k	R^2
File size	12.1M	112B	58.6KB	1.60	0.97	13.8M	112B	68.6KB	1.46	0.96
LOC	337,527	2	1407	1.19	0.80	382,327	1	1643	1.17	0.83
Number of Elements	192,880	1	684	1.37	0.91	218,472	1	959	1.37	0.88
Number of Annotations	28,051	0	76	1.60	0.97	29,891	0	58	1.57	0.96
ComplexType Elements	1,014	0	15	1.63	0.96	2,222	0	10	1.84	0.96
SimpleType Elements	1,320	0	14	1.85	0.90	1368	0	5	1.94	0.95

Table 3.3: Summary of size metrics R^2 value for two datasets

	Dataset1		Dataset2	
Name	R^2 Lognormal	R^2 Power law	R^2 Lognormal	R^2 Power law
File Size	0.98	0.97	0.99	0.96
LOC	0.95	0.80	0.98	0.83
Element Number	0.98	0.81	0.99	0.88
ComplexType Element	0.97	0.96	0.97	0.96
SimpleType Element	0.91	0.90	0.96	0.95
Annotation Element	0.98	0.98	0.97	0.96

Chapter 4

Structure Metrics

4.1 Graph Model of XML Schema

In this section, we introduce the graph model of XML Schema and some definitions for small world phenomenon.

4.1.1 Modeling XML Schema as Directed Acyclic Graph

XML Schema is modeled as Directed Acyclic Graph [11], where each element and complexType of the schema is translated into a node, There is an edge from node A to node B if either one of the following is true:

- B is a component in the content model of A. e.g.,

```
<element name="order">
    <complexType>
        <element name="address">
```

Element address is nested in element of order, then on the DAG graph from node of order,

there is an edge to node address.

- B is the type of A. e.g.,

  <element name="book" type="CourseReserveType">

there is an edge linking "book" to "CourseReserveType" because "CourseReserveType" is a type of "book" element.

- B is the reference of A. e.g.,

  <element name="ID" type="IDType">

  <sequence>

  <element ref="CopyIndicator" minOccurs="0">

The element "CopyIndicator" has already been defined in schema, and then there is an edge connects it to the node "ID".

- B is the dataType of A. e.g.,

  <xsd:ComplexType name="PurchaseOrderType" datatype="MailingAddressType">

The datatype "MailingAddressType" is defined at the top level of the schema, then there is a link between the node of "PurchaseOrderType" to the node of "MailingAddressType".

- B is an extension of A. e.g.,

  <element name="RoomType" minOccurs="0">

  <complexContent>

  <extension base="BuildingDesign">

From the node of "RoomType" to the node of "BuildingDesign", there is a link.

Intuitively, each node represents an entity, and the edge indicates that there is a relationship between the two entities. In XML_OO binding such as JAXB, those nodes are mapped to classes and edges to the relations between classes. Hence, our graph corresponds roughly to a class diagram in UML.

With that motivation, in our analysis we do not translate annotations and attributes into nodes, because in a class diagram they are normally mapped into fields of a class. To focus more on user-defined software artifacts, we also ignore primitive XSD data types such as String and int, and simpleTypes such as Strin32 that are directly derived from primitive data types.

There are several difficulties while processing the data.

- Import: To simplify the problem, we ignore the import statements. In our analysis, when an element declaration refers to a different schema, we treat this element as a new node on graph.
- Recursive: Since the schema is modeled as DAG, there is no cycle on the graph. When the program adds the edges between nodes, it can detect whether a cycle will be formed with the addition of this edge. If it forms a cycle, the edge will not be added.
- Name similarity: In schema, sometimes it happens that the value of name attribute and value of type attribute is same. For example, <element name="table" type="orm:table">. Here, the namespace orm is defined in this schema and we treat the table and orm:table as one node. To avoid cycles on the graph, we add a prefix "pre_"on the value of table and treat it as a different node.

To calculate the in/out degree for each node on the graph, we make the following definitions:

In-degree: the in-degree of a node v is the number of incoming edges that v has.

Out-degree: the out-degree of a node v is the number of out-going edges v has.

Figure 4.1 is an schema example and the corresponding DAG.

For this DAG, the in-degree, out-degree for each node is following (the node order is:

```
<?xml version="1.0"?>
 <xsd:schema
 targetNamespace=
  "http://cs.uwindsor.ca/schemaExample"
 xmlns:xsd=
  "http://www.w3.org/2001/XMLSchema">
 <xsd:annotation>
   <xsd:documentation>
      This is an example XML Schema for a library.
   </xsd:documentation>
 </xsd:annotation>
 <xsd:simpleType name="string32">
   <xsd:restriction base="xsd:token">
     <xsd:maxLength value="32"/>
   </xsd:restriction>
 </xsd:simpleType>
 <xsd:complexType name="TextBookType">
   <xsd:sequence>
    <xsd:element name="BookTitle" type="string32"/>
    <xsd:element name="author" type="xsd:string"
      maxOccurs="unbounded"/>
   </xsd:sequence>
   <xsd:attribute name="isbn" type="xsd:string"/>
 </xsd:complexType>
 <xsd:complexType name="libraryBookType">
   <xsd:complexContent>
    <xsd:extension base="TextbookType">
    <xsd:attribute name="CallNumber"
       type="xsd:string"/>
   </xsd:extension>
   </xsd:complexContent>
 </xsd:complexType>
 <xsd:element name="library">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="book"
      type ="libraryBookType"
       maxOccurs="unbounded"/>
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
</xsd:schema>
```

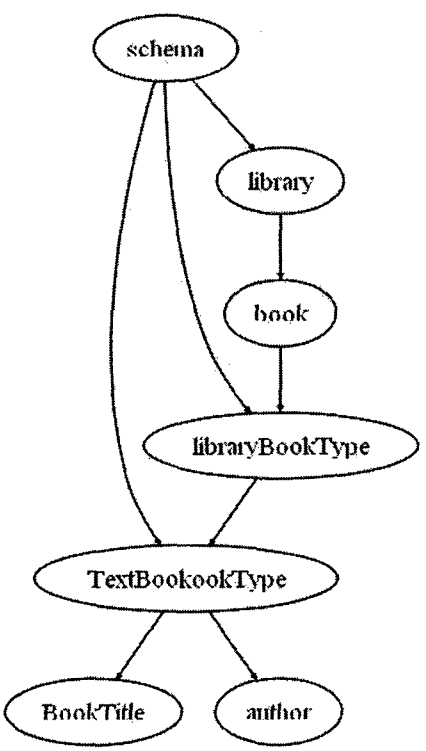

Figure 4.1: XML Schema example (top) and its graph

schema; library; book; libraryBookType; TextBookType; BookTitle; author):

In-degree: 0 1 1 2 2 1 1

Out-degree: 3 1 1 1 2 0 0

4.1.2 Schema Length

For the purpose of analyzing small world phenomenon, following the method in [36], from the above modeled DAG, we remove the direction of DAG, then the graph becomes undirected and each node pair becomes reachable. There is one shortest path between each node pair. The total number of shortest path on a connected graph with the node number of n is: $n(n-1)/2$.

From the undirected graph, we give following definitions:

Distance: A Graph $G(V,E)$ is a pair of sets V and E, where V is a set of vertices, and E is a set of edges connecting the vertices. The geodesic, or the shortest path between two nodes i and j is denoted as d_{ij}.

The distance, or the average path length of G, is the average of all the shortest paths between any two pairs of nodes.

$$d_{mean} = \frac{2\sum_{i>j} d_{ij}}{n(n-1)} \tag{4.1}$$

d_{mean} can be found using Floyd-Warshall algorithm in time complexity of $O(|V|^3)$ [40].

The diameter of G is the maximum of all the shortest paths between any two pairs of nodes. i.e.,

$$d_{max} = Max_{i>j} d_{ij} \tag{4.2}$$

When the graph G is not connected, there are pairs of nodes whose shortest path is infinite. Therefore both d_{mean} and d_{max} are infinite. In that case, we can only calculate the distance for connected graphs.

A network shows a small world effect if the average path length increases logarithmically or slower with graph size for a fixed average degree, i.e.,

$$d_{mean} = a + b log(n) \tag{4.3}$$

We discuss the small world phenomenon in later section.

To find the shortest path length for each node pair on the graph, we implement Floyd-Warshall algorithm to obtain the result. Floyd-Warshall algorithm is used to obtain all possible shortest paths through the graph between each pair of nodes. The following is an explanation of this algorithm and an example to calculate the average shortest path.

Floyd-Warshall algorithm

To implement Flod-Warshall algorithm. Initially, a graph G with node n and edge set E can be represented by an $n \times n$ matrix with its edge cost,

$$d_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \text{ and } (i,j) \in E \\ \infty & \text{if } i \neq j \text{ and } (i,j) \notin E \end{cases}$$

After the execution of the algorithm, the output is a matrix $D = [d_{ij}]$ where d_{ij} is the shortest path from node v_i to v_j.

On graph G, the nodes v_2, v_3,..., v_{l-1} are called the intermediate nodes of the path p=< v_1, v_2,...v_l >.

Let $d_{ij}^{(k)}$ be the length of the shortest path from node v_i to v_j such that any intermediate nodes on the path are chosen from the set{ v_1, v_2,...v_k}.

Let $D^{(k)}$ be the n × n matrix $[d_{ij}^{(k)}]$. At each iteration step k, the aim is to compute $D^{(k)}$ from $D^{(k-1)}$ for $k = 0, 1, \ldots, n$.

For a shortest path from i to j such that any intermediate nodes on the path are chosen from the set { v_1, v_2,...v_k}, there are two possibilities:

1. v_k is not a node on the path,

the shortest such path has length d_{ij}^{k-1}.

2. v_k is a node on the path.

The shortest path has length $d_{ik}^{k-1} + d_{kj}^{k-1}$.

Combining the two cases we get

$$d_{ij}^{k} = \min\{d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1}\}.$$

The matrix is calculated in following pseudo code:

Start with all single edge path

```
For i = 1 to n do
    For j = 1 to n do
        d(i,j) = edgeCost(i,j)
```

```
d(i,j) is "best" distance so far from node i to node j.
    For k = 1 to n do (k is the 'intermediate' node)
        For i = 1 to n do
            For j = 1 to n do
                if (d(i,k) + d(k,j) < d(i,j))
                    d(i,j) = d(i,k) + d(k,j)
```

Example

We use Figure 4.1 as an example. First, we start with the adjacency matrix of the undirected graph that there is an edge between two nodes. If there is an edge between two nodes, it is indicated with 1, otherwise, it is ∞. The node itself is indicated with 0.

Initially, the adjacency matrix of the example graph is following (the node order is same with above):

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

Then start from first node of the node set, (on this graph, it is the node schema), each

node is added as an intermediate node between any two nodes in hope to find a shorter path. On this matrix, after the first node of schema is added, the matrix becomes the following:

$$\begin{pmatrix} 0 & 1 & \infty & 1 & 1 & \infty & \infty \\ 1 & 0 & 1 & 2 & 2 & \infty & \infty \\ \infty & 1 & 0 & 1 & \infty & \infty & \infty \\ 1 & 2 & 1 & 0 & 1 & \infty & \infty \\ 1 & 2 & \infty & 1 & 0 & 1 & 1 \\ \infty & \infty & \infty & \infty & 1 & 0 & \infty \\ \infty & \infty & \infty & \infty & 1 & \infty & 0 \end{pmatrix}$$

After the node schema is added as an intermediate node between any other node pairs, only two temporary shortest paths are found between node library and libraryBookType through node schema. The path length is 2. No other paths are founded.

Then the second node library is added as an intermediate node based on last step, it connects the path from node schema to node book. The length is 2. In addition, it connects the path from book to libraryBookType through node library and node schema. The length is 3. But the original path from book to libraryBookType has the length value of 1, it's shorter than the new path length, then the length of 1 is used as the shortest path between book and libraryBookType. At same time, it connects the path from book to TextBookType with the length of 3. The matrix becomes the following:

$$\begin{pmatrix} 0 & 1 & 2 & 1 & 1 & \infty & \infty \\ 1 & 0 & 1 & 2 & 2 & \infty & \infty \\ 2 & 1 & 0 & 1 & 3 & \infty & \infty \\ 1 & \infty & 1 & 0 & 1 & \infty & \infty \\ 1 & 2 & 3 & 1 & 0 & 1 & 1 \\ \infty & \infty & \infty & \infty & 1 & 0 & \infty \\ \infty & \infty & \infty & \infty & 1 & \infty & 0 \end{pmatrix}$$

Similarly and repeatedly, adding the third node book, the fourth node libraryBookType, and the fifth node TextBookType is added as intermediate node, the matrix becomes:

$$\begin{pmatrix} 0 & 1 & 2 & 1 & 1 & 2 & 2 \\ 1 & 0 & 1 & 2 & 2 & 3 & 3 \\ 2 & 1 & 0 & 1 & 2 & 3 & 3 \\ 1 & 2 & 1 & 0 & 1 & 2 & 2 \\ 1 & 2 & 2 & 1 & 0 & 1 & 1 \\ 2 & 3 & 3 & 2 & 1 & 0 & 2 \\ 2 & 3 & 3 & 2 & 1 & 2 & 0 \end{pmatrix}$$

Still, the last two nodes BookTitle and author is added, but there is no change with the result. The final matrix for the length of shortest is the same as the above. There are total 21 pairs of nodes on the graph and the average path length is 1.81.

4.2 In/Out Degree and Structure of Eight Schemas

In this section, we discuss and analyze the in/out degree distributions for individual XML Schema.

4.2.1 Data Collection

We choose the schemas size larger than 2MB in both datasets. For such kind of schemas, there are a total of 112 in dataset1 and 128 in dataset2. Also we choose 3 schemas which has large average path value but their sizes are less than 2MB. We are interested to find the difference of their structures with other large size schemas. Out of these large schemas we randomly choose 8 schemas from two datasets for detailed analysis. These 8 schemas are introduced in following (we have two PDBML schemas).

1) Schema UN/ECEFACT: UN/ECEFACT is a schema of specification and codelist about business information in UN Economic Commission For Europe.

2) Schema PDBML: PDBML is the abbreviation of Protein Data Bank Markup Language. It provides a representation of PDB data in XML format.

3) Schema NIEM: NIEM stands for the National Information Exchange Model. It provides the foundation and building blocks for national-level interoperable information sharing and data exchange.

4) Schema SIF: SIF Association is a membership organization where various software vendors or education departments have come together to create a set of rules and definitions which enable software programs from different companies to share information. This set of platform-independent, vendor-neutral rules and definitions is called the SIF Implementation Specification. The Schema of SIF message lists the names and data type for this specification.

5) Schema e-Bay: eBay schema is a specification about an API for online shopping.

6) Schema CIM: CIM is the abbreviate of Common Information Model. It is a standard that defined how managed elements in an IT environment are represented as a common set of objects and relationships between them. CIM Schema lists all management elements and its format.

7) Schema JUSTICE: The JUSTICE Schema specifies all codes from the National Crime and Information Center in Japan.

4.2.2 In/Out Degree and Structure of Eight Schemas

We analyze the above 8 schemas with their DAG graph to get in-degree and out-degree number. We plot the diagrams of in-degree and out-degree distribution with the power-law fitting in Figure 4.2 and Figure 4.3. The diagrams are plotted in log-log scale with the bin size increasing logarithmically. Table 4.1 is a statistic summary of in/out degrees with power law fitting.

In our experiment, we observe that the k value of power law fitting for data in both datasets is around $0.8 - 2.0$. This value is consistent with the k value of power-law distributions in other areas, like internet topology, Java class relationship, etc.

On graph representation of schema, the nodes with high in-degrees means that an element or ComplexType declaration has broad references. The high out-degrees happen at the node representing elements, like root elements, complexType element which has a great number of child elements. The higher the out-degree, the wider the structural graph. For both in and out degree, it couldn't be increased without limitation as it causes the increment of complexity and increases the cost of whole schema [36].

Figure 4.4 and Figure 4.5 are visual structure graphs for schema SIF Information and

CIM which has the average path value with 4.0 and 3.3 but the diameter value is 10 and 6, respectively. From the schema itself, we observe that the schema CIM has many elements and complexType declarations with primitive datatype which is not considered in our program. In addition, the majority element and complexType declarations are directly under the root node "schema". This causes the graph structure grows "wider" but not "deeper". For schema SIF Information, there are less primitive datatype uses for element and complexType declaration but more referred to other places which are already declared. This causes the length to grow and increases the average path length of the schema. We append the structure graphs for other six schemas in appendix.

Table 4.1: Summary of in/out degrees of eight schemas with Power Law fitting

Schema Name	Nodes	Ave. path len.	Diameter	Edges	Ave. degree	k (in/out)	R^2 (in/out)	std. dev.(in/out)
e-Bay	3326	3.34	6	5880	1.78	1.61/1.03	0.88/0.89	4.6/22.4
PDBML1	3048	4.89	6	4377	1.44	1.92/0.98	0.97/0.87	1.9/9.7
PDBML2	3122	4.93	6	4591	1.47	1.67/0.94	0.95/0.86	2.2/9.5
Purchase Order	651	2.24	4	2722	4.18	0.88/0.84	0.85/0.9	1.1/22.3
NIEM	2277	2.08	4	6081	2.67	1.0/0.65	0.72/0.51	22.5/49.4
UN/CEFACT	1877	3.2	6	4838	2.58	1.34/1.19	0.93/0.67	8.4/17.4
SIF Information	684	4.0	10	1220	1.79	1.83/1.06	0.93/0.96	2.2/7.6
CIM	1000	3.3	6	3479	3.5	1.05/1.0	0.87/0.89	12.1/14.0

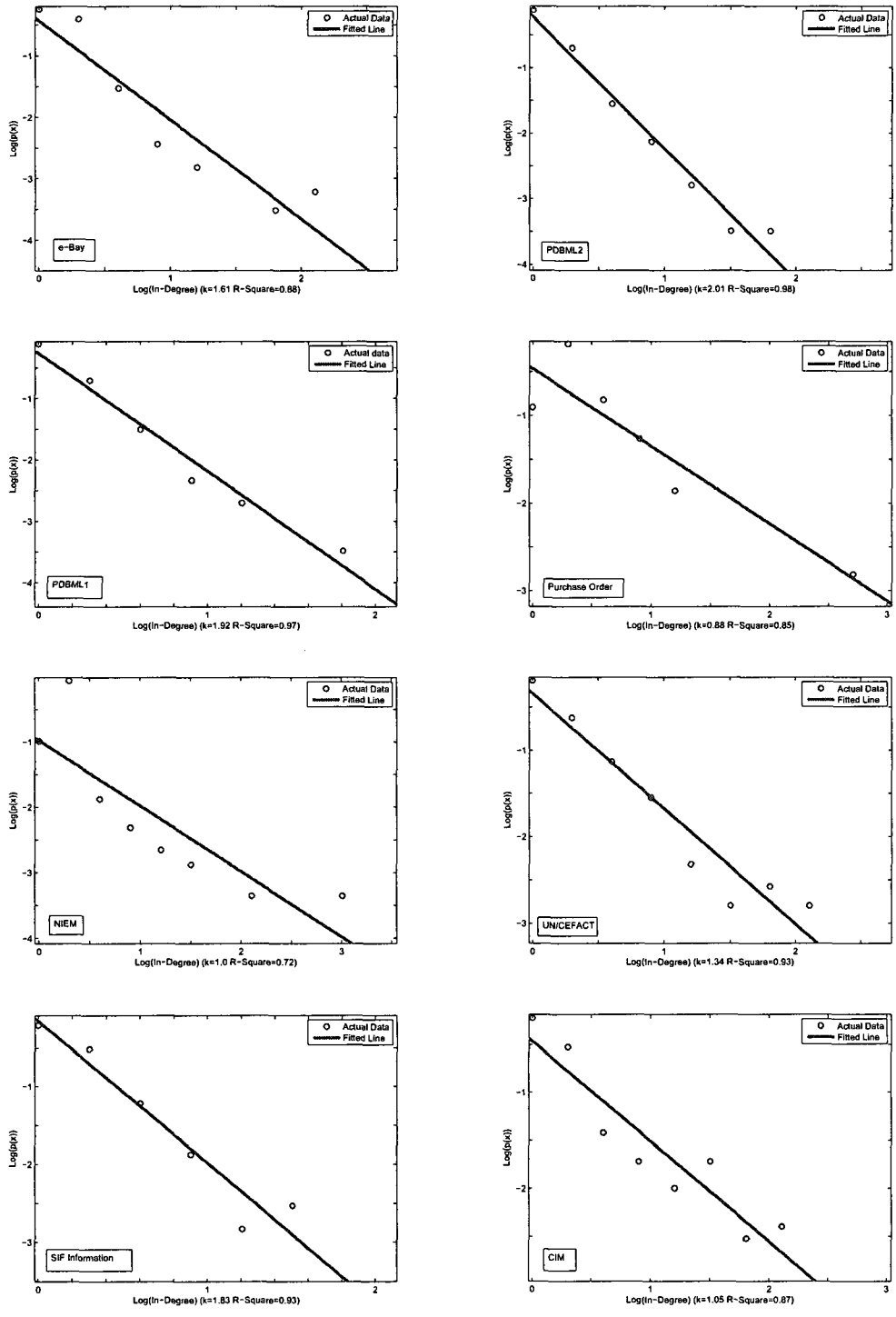

Figure 4.2: Histograms of in-degree for eight schemas listed in Table 4.1. Plots are in log-log scale. Bin size for in-degree increases in 2^n, n=0,1,2,....

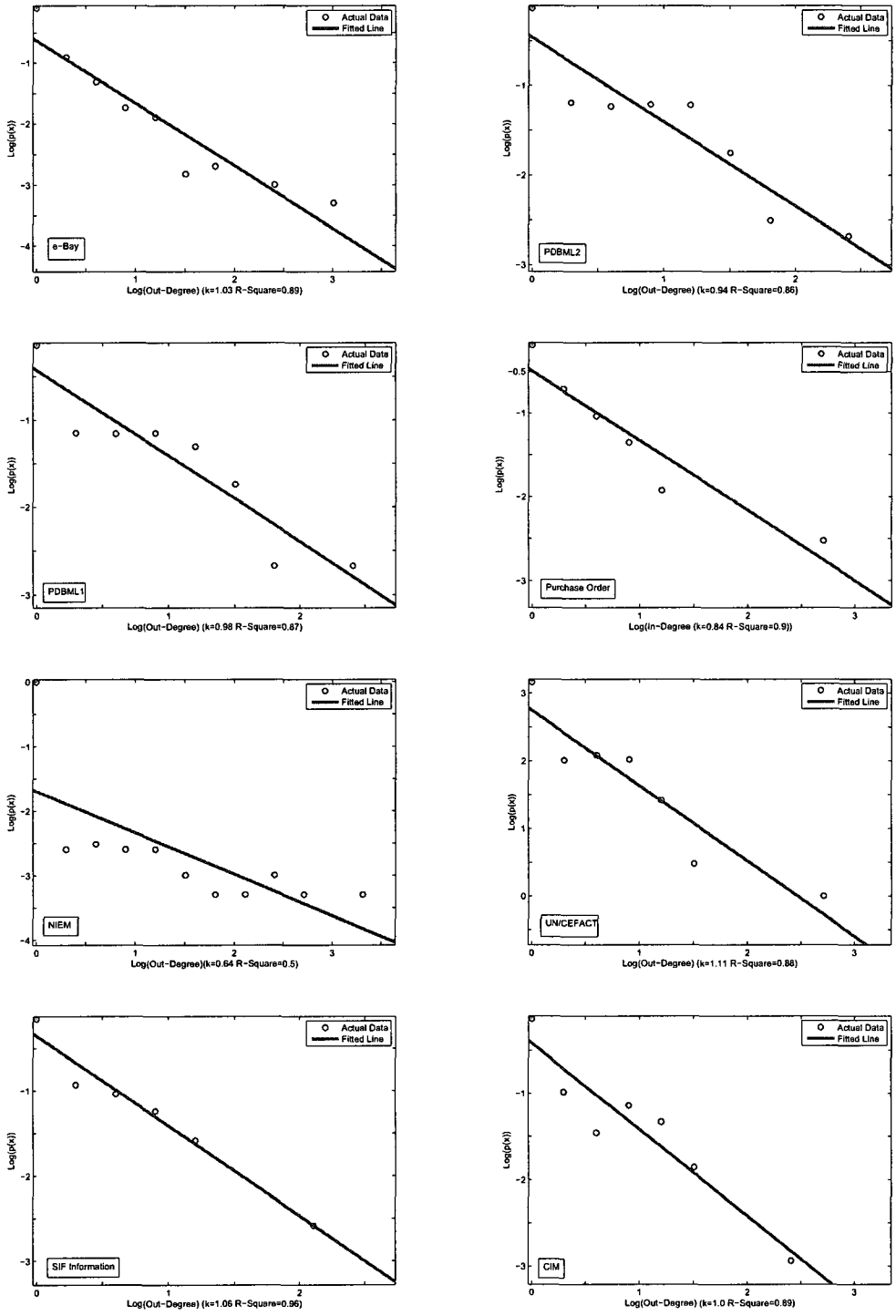

Figure 4.3: Histograms of out-degree for eight schemas listed in Table 4.1. Plots are in log-log scale. Bin size for out-degree increases in 2^n, n=0,1,2,....

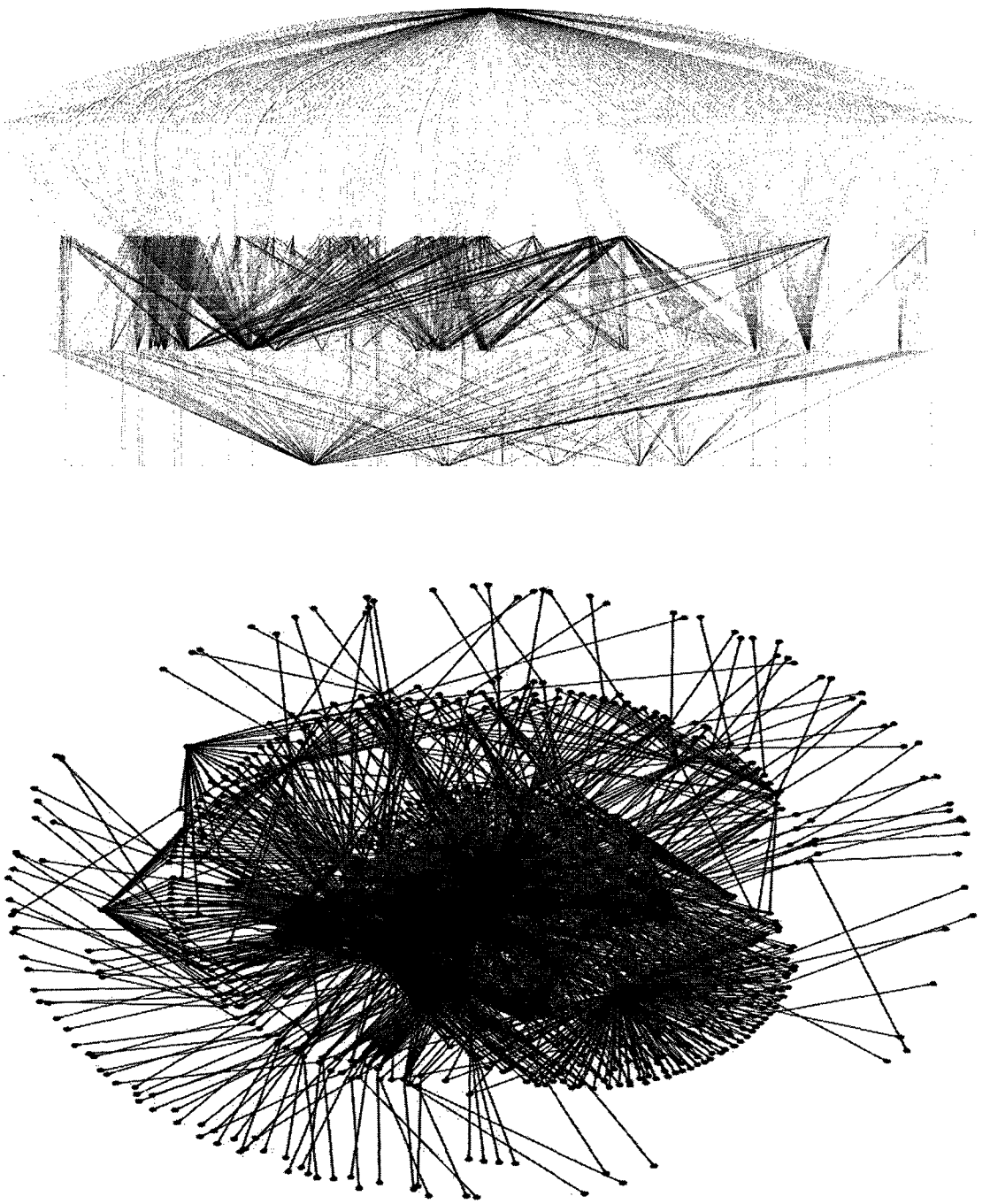

Figure 4.4: Structure of Schema CIM. two diagrams are the same but have different layout. Top: plotted by GraphViz DOT. Bottom: Plotted by GraphViz NEATO

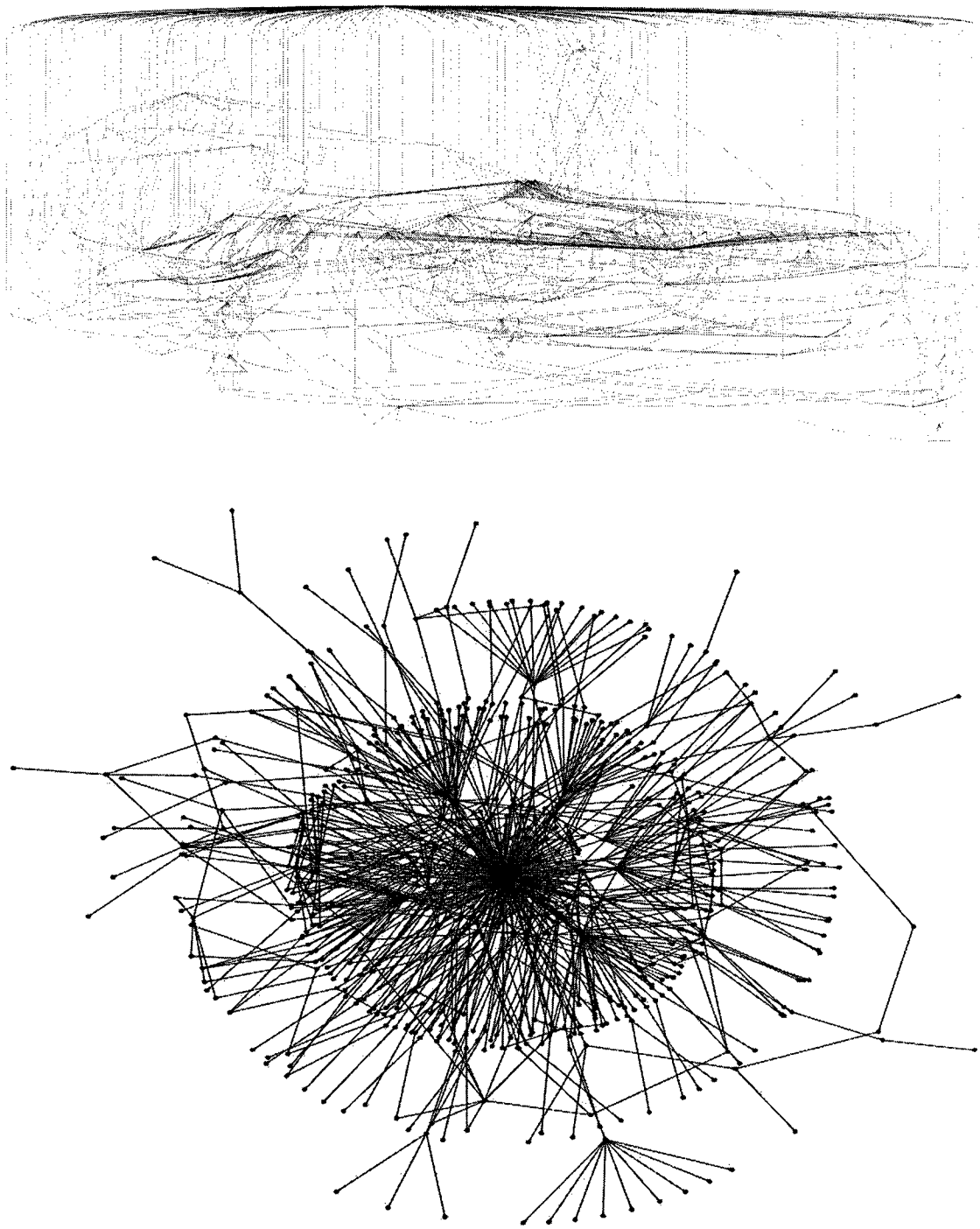

Figure 4.5: Structure of Schema SIF Information. two diagrams are the same but have different layout. Top: plotted by GraphViz DOT. Bottom: Plotted by GraphViz NEATO

4.3 Structure Metrics of Datasets

In this section, we analyze the structure metrics for two subdatasets and discuss small world phenomenon.

4.3.1 Data Collection

Structure metrics include the number of nodes, edges and average path length or diameter of the schemas. These metrics are based on schemas' graph model. On graph representation of a schema, the total number of in-degree and out-degree is the same as its edges, we analyze the in-degree as its edge distribution. Due to the restriction of the program, we only analyze the schemas with the size less than 1MB for their graph model in two datasets. We rename the two datasets as sub-dataset1 and sub-dataset2. In sub-dataset1, there are 5586 such schemas. In sub-dataset2, there are 11151 such schemas. Figure 4.6 is the histogram of the number of nodes, the number of edges, average path length and diameter distribution with the power law fitting for the two subdatasets. Table 4.2 is the statistical summary for these metrics.

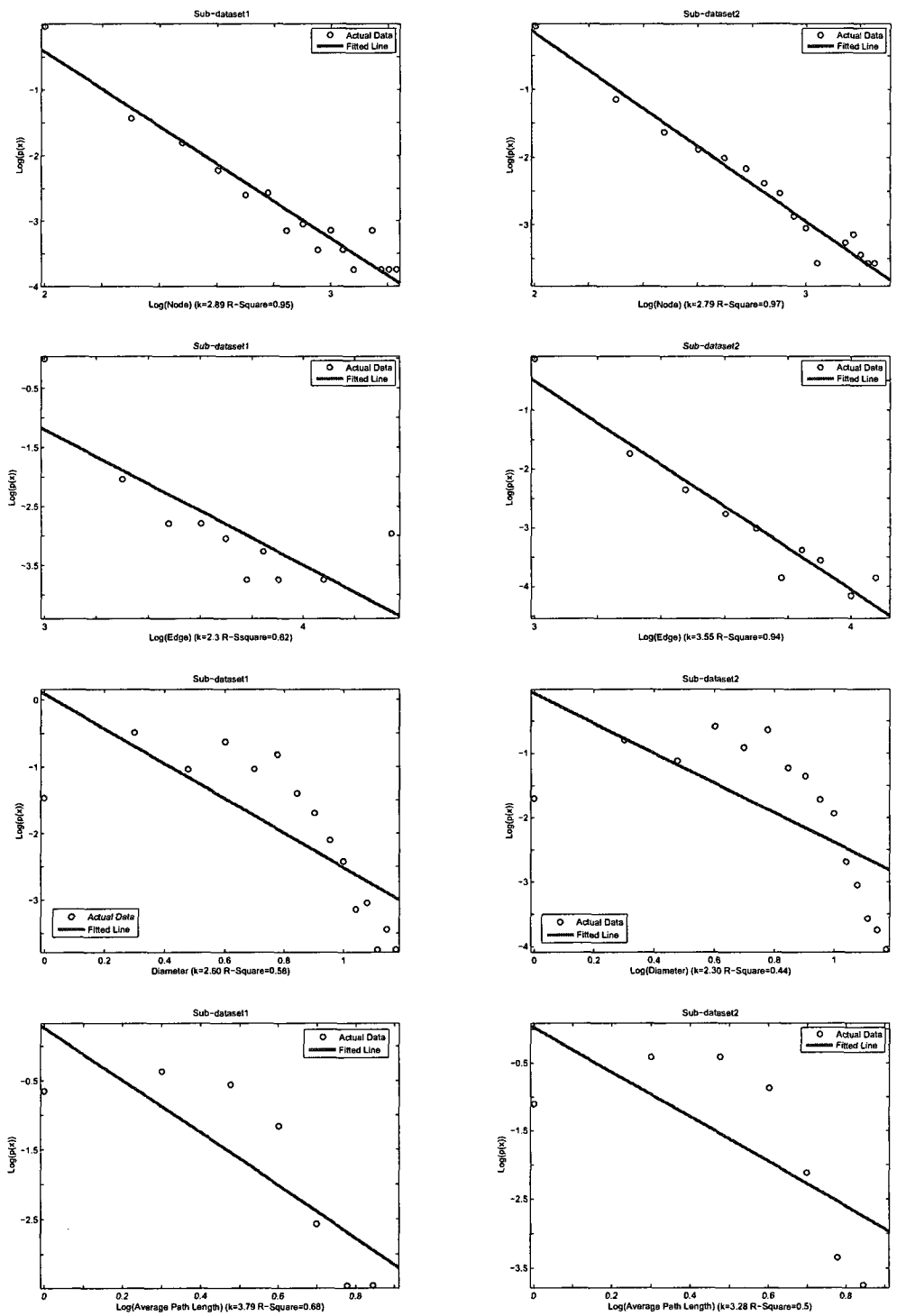

Figure 4.6: Histograms of number of nodes, edges, diameter and average path length for two sub-datasets. Plots are in log-log scale. Bin size for number of nodes increases with 100; edges: 1000; diameter and average path length: 1

Table 4.2: Summary of the statistics for structure metrics of two subdatasets

	Sub-dataset1						Sub-dataset2					
Name	Max.	Min.	Mean	Std.dev.	k	R^2	Max.	Min.	Mean	Std.dev.	k	R^2
Nodes in a schema	1685	2	48	98.9	1.76	0.89	1961	2	82	138.9	2.79	0.97
Diameter	15	1	3.84	1.88	2.60	0.58	15	1	4.67	1.88	2.30	0.44
Average Path Length	7.17	1	2.26	0.76	3.7	0.68	7.3	1	2.61	0.79	3.28	0.50
Edges in a schema	22166	1	191	839.7	2.32	0.62	12266	1	280	557.8	3.57	0.94

4.3.2 Number of Nodes and Edges

For those small size schemas in sub-dataset1, the maximum number of nodes for a schema is 1685, and the average is 48. For sub-dataset2, the two values are 1961 and 82. The minimum number of node for a schema in both subdataset is 2. From our modeling, we decide to only extract the node with elements and complexType which have name and type declarations. This is why the node number is much less than the metrics of number of elements which is obtained from schema itself. On a graph, the total in-degrees is the same with the total out-degrees and this number is the total edge number. For sub-dataset1, the maximum edge number of a schema is 22166. The average is 191 and the minimum is 1. For sub-dataset2, these values are 12266, 280 and 1 respectively.

For the two subdatasets, a linear growth relation can be found between the number of edges and number of nodes(see Figure 4.7). The relation is in the form of $edge \sim node^{1.22}$ for sub-dataset1 and $edge \sim node^{1.17}$ for sub-dataset2.

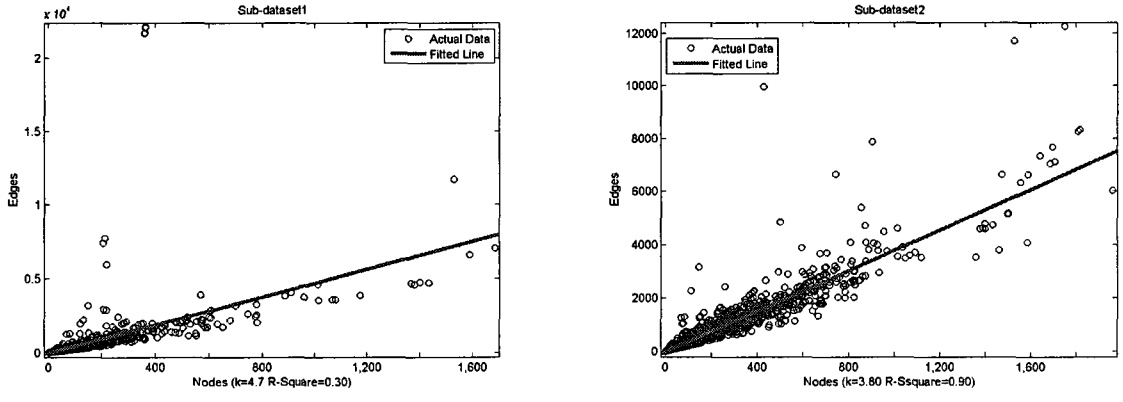

Figure 4.7: Histograms of number of edges and number of nodes

4.3.3 Small World

The small world phenomenon was first proposed by [24] where a series of experiments were conducted and the goal of each experiment was to find short chains of acquaintances linking pairs of people in the United States who didn't know each other. There were 60 letters were sent to a destination with the restraint that a letter was passed from person to person by hand and one individual had to pass the letter only to personal acquaintances who they thought might be able to reach the destination - whether directly or via a "friend of friend". The experiment showed that the average number of intermediate steps in a successful chain lies between five and six. This result was also known as the *six degree of separation.*. Usually, small world refers that the size of diameter of the network is not a large value.

Diameter and Average Path Length

Diameter is the maximum among all shortest paths on a graph. For sub-dataset1, the maximum diameter of a schema is 15 and the mean value is 3.84. The minimum is 1. The maximum average path length of a schema is 7.17 and the mean value is 2.26. For subdataset2, the maximum and mean value are 15, 4.67 for diameter and 7.3 and 2.61 for average path length. The minimum of both values for two subdatasets is 1.

To analyze the small world phenomenon, we use matlab to do linear regression analysis and obtain the equation for the relation between number of nodes and average path length for two subdatasets. See equation 4.4 and equation 4.5. The diagram for this relationship is in Figure 4.8.

$$d_{mean} = 1.0018 + 0.9821 log(n) \qquad for\ sub-dataset1 \tag{4.4}$$

and

$$d_{mean} = 1.0861 + 0.9798log(n) \qquad for\ sub-dataset2 \tag{4.5}$$

From the above equation, we can see that the average path length is not a large value even if it is a big schema and has lots of nodes on its graph model. If a schema has 1000 nodes on its graph, from the above equation, it can be seen that the average path length is around 4. In our experiment result, the average path length for the majority of small size schemas (less than 1MB) is around this value.

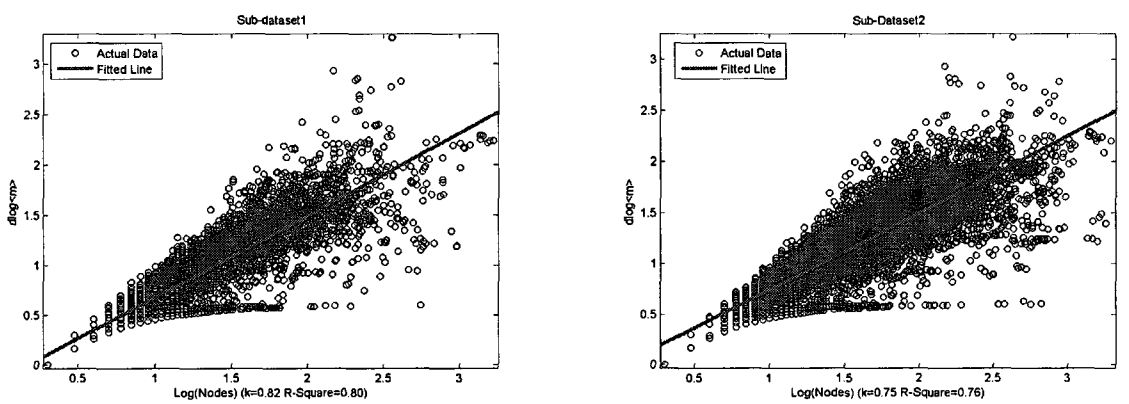

Figure 4.8: The relationship between average path length and number of nodes. In figure, d is the average path length and <m> is the average degree of a schema

For many systems whose structure can be represented in graph form, it is helpful to keep the average path length at a low value [36]. Low average path length has been observed in many real systems, including the Internet or social networks, though sometimes they have a large number of nodes. In [2], the authors claim that two randomly chosen documents on the Internet are, on average, 19 clicks away from each other.

Chapter 5

Conclusions

In this thesis, we analyzed size metrics and structure metrics for two different XML schema datasets. For each metric, we analyzed its power law distribution and lognormal distribution. From the experiment, we found that the lognormal distribution appears to be a better fit for the size metrics than power law distribution. Also we analyzed the in-degree and out-degree statistical distributions for individual schema from its graph model. We found that power law distributions is a better fit for both metrics, indicating that the schema creating activity can be simply modeled as a random addition of new elements. However, this addition exhibits strong dependence on what has been already created. There is a high rate of re-definition or references among the element and complexType declarations.

We also analyzed the small world effect of schemas for which the sizes are less than 1MB in two datasets. The relationship between number of nodes and average path length is expressed with an equation. From the diagrams we can see a trend: with the increment of the schema size, the average path length increases accordingly. But this increment has a certain limit. For the two datasets, the maximum average path length is 15. For the majority of schemas, this value is around 4.

Future Work

There are two directions that the research presented in this thesis could be taken further. First, it could be useful to repeat the analysis on other software systems to see whether the above conclusions are consistent. For most of the in/out-degrees in different systems, the power law distribution has been confirmed. We hope to give a further detailed comparison of metrics distribution between XML schema and other software systems.

Second, it is necessary to expand the datasets and analyze more kinds of schema files. Researchers interested in the topic could crawl more schemas from internet and broaden the metrics analysis. Then following the theory of small world, it could be interesting to verify the exact relationship between the node number and the value of average path length existing in XML schema.

Bibliography

[1] Lada A. Adamic, Bernardo A. Huberman;, A.-L. Barabasi, R. Albert, H. Jeong, and G. Bianconi;. Power-Law distribution of the world wide web. *Science*, 287(5461):2115a, March 2000.

[2] Reka Albert, Hawoong Jeong, and Albert-Laszlo Barabasi. Internet: Diameter of the World-Wide web. *Nature*, 401(6749):130–131, 1999.

[3] L. A. N. Amaral, A. Scala, M. Barthelemy, and H. E. Stanley. Classes of small-world networks. *Proceedings of the National Academy of Sciences of the United States of America*, 97(21):11149–11152, October 2000.

[4] Albert-Laszlo Barabasi and Reka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, October 1999.

[5] Dilek Basci and Sanjay Misra. Complexity metric for xml schema documents. In *Proceedings of Object-Oriented Programming, Systems, Languages, and Applications, Montreal Canada*, Oct. 2007.

[6] Gareth Baxter, Marcus Frean, James Noble, Mark Rickerby, Hayden Smith, Matt Visser, Hayden Melton, and Ewan Tempero. Understanding the shape of java software. *ACM SIGPLAN*, 41(10):397–412, 2006.

[7] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. Graph structure in the web. *Computer Networks*, 33(1-6):309–320, June 2000.

[8] S. R. Chidamber and C. F. Kemerer. A metrics suite for object oriented design. *IEEE Trans. Softw. Eng.*, 20(6):476–493, 1994.

[9] B. CHOI. Choi: What are real DTDs like?(technical report), 2002.

[10] Giulio Concas, Michele Marchesi, Sandro Pinna, and Nicola Serra. Power-Laws in a large Object-Oriented software system. *IEEE Trans. Softw. Eng.*, 33(10):687–708, 2007.

[11] Hong-Hai Do and Erhard Rahm. COMA: a system for flexible combination of schema matching approaches. In *Proceedings of the 28th international conference on Very Large Data Bases*, pages 610–621, Hong Kong, China, 2002. VLDB Endowment.

[12] Allen B. Downey. Evidence for long-tailed distributions in the internet. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 229–241, San Francisco, California, USA, 2001. ACM.

[13] Allen B. Downey. The structural cause of file size distributions. *SIGMETRICS Perform. Eval. Rev.*, 29(1):328–329, 2001.

[14] P Erdos and A Renyi. On random graphs, i. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[15] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *Proceedings of the conference on Applications,*

technologies, architectures, and protocols for computer communication, pages 251–262, Cambridge, Massachusetts, United States, 1999. ACM.

[16] Xavier Gabaix. Zipf's law and the growth of cities. *The American Economic Review*, 89(2):129–132, May 1999.

[17] H. Jeong, S. P. Mason, A.-L. Barabasi, and Z. N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411(6833):41–42, May 2001.

[18] Meike Klettke, Lars Schneider, and Andreas Heuer. Metrics for XML document collections. In *Proceedings of the Worshops XMLDM, MDDE, and YRWS on XML-Based Data Management and Multimedia Engineering-Revised Papers*, pages 15–28. Springer-Verlag, 2002.

[19] Ralf Lammel, Stan Kitsis, and Dave Remy. Analysis of xml schema usage. In *Conference Proceedings XML 2005*, Nov. 2005.

[20] Panagiotis Louridas, Diomidis Spinellis, and Vasileios Vlachos. Power laws in software. *ACM Trans. Softw. Eng. Methodol.*, 18(1):1–26, 2008.

[21] T.J. McCabe. A complexity measure. *Software Engineering, IEEE Transactions on*, SE-2(4):308–320, 1976.

[22] Andrew McDowell, Chris Schmidt, and Kwok-Bun Yue. Analysis and metrics of xml schema. In *SERP '04, Proceedings of the International Conference on Software Engineering Research and Practice*. CSREA Press.

[23] Laurent Mignet, Denilson Barbosa, and Pierangelo Veltri. The XML web: a first study. In *Proceedings of the 12th international conference on World Wide Web*, pages 500–510, Budapest, Hungary, 2003. ACM.

[24] Stanly Milgram. The small world problem. *Psychology today*, 2:60–67, 1967.

[25] Michael Mitzenmacher. A brief history of generative models for power law and log-normal distributions. *Internet Mathematics*, 1(2):226–251, 2004.

[26] Michael Mitzenmacher. Dynamic models for file sizes and double pareto distributions. *Internet Mathematics*, 1(3):305–333, 2004.

[27] Irena Mlynkova, Kamil Toman, and Jaroslav Pokorny. Statistical analysis of real xml data collections (technical report). In *Proceedings of COMAD-2006*, Delhi, India, 2006.

[28] Christopher R. Myers. Software systems as complex networks: Structure, function, and evolvability of software collaboration graphs. *Physical Review E*, 68(4):15, October 2003.

[29] M. E. J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences of the United States of America*, 98(2):404–409, 2001.

[30] MEJ. Newman. Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46:323–351, September 2005.

[31] Alex Potanin, James Noble, Marcus Frean, and Robert Biddle. Scale-free geometry in OO programs. *Commun. ACM*, 48(5):99–103, 2005.

[32] M.H. Qureshi and M.H. Samadzadeh. Determining the complexity of XML documents. In *Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on*, volume 2, pages 416–421 Vol. 2, 2005.

[33] Mukund Seshadri, Sridhar Machiraju, Ashwin Sridharan, Jean Bolot, Christos Faloutsos, and Jure Leskove. Mobile call graphs: beyond power-law and lognormal distributions. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 596–604, Las Vegas, Nevada, USA, 2008. ACM.

[34] Giancarlo Succi and Michele Marchesi, editors. *Extreme programming examined.* Addison-Wesley Longman Publishing Co., Inc., 2001.

[35] S. Valverde, R. Ferrer Cancho, and R. V. Sole. Scale-free networks from optimal design. *EPL (Europhysics Letters)*, 60(4):512–517, 2002.

[36] Sergi Valverde and Ricard V Sole. Hierarchical small worlds in software architecture. *IEEE Transaction on Software Engineering*, July 2004.

[37] Joost Visser. Structure metrics for xml schema. In *Proceedings of XATA, 2006*, 2006.

[38] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of /'small-world/' networks. *Nature*, 393(6684):440–442, June 1998.

[39] R. Wheeldon and S. Counsell. Power law distributions in class relationships. In *Source Code Analysis and Manipulation, 2003. Proceedings. Third IEEE International Workshop on*, pages 45–54, 2003.

[40] Wikipedia. FloydcWarshall algorithm. Website, 2010. `http://en.wikipedia.org/wiki/Floyd%E2%80%93Warshall_algorithm`

[41] Hongyu Zhang. Discovering power laws in computer programs. *Inf. Process. Manage.*, 45(4):477–483, 2009.

[42] Hongyu Zhang and Hee Beng Kuan Tan. An empirical study of class sizes for large java systems. In *Proceedings of the 14th Asia-Pacific Software Engineering Conference*, pages 230–237. IEEE Computer Society, 2007.

[43] Hongyu Zhang, Hee Beng Kuan Tan, and Michele Marchesi. The distribution of program sizes and its implications: An eclipse case study. *0905.2288*, page 10, May 2009.

Appendices

Appendix A

Structure Graphs of Schemas

In this part, we list the structure graph of schemas studied in Chapter 4.

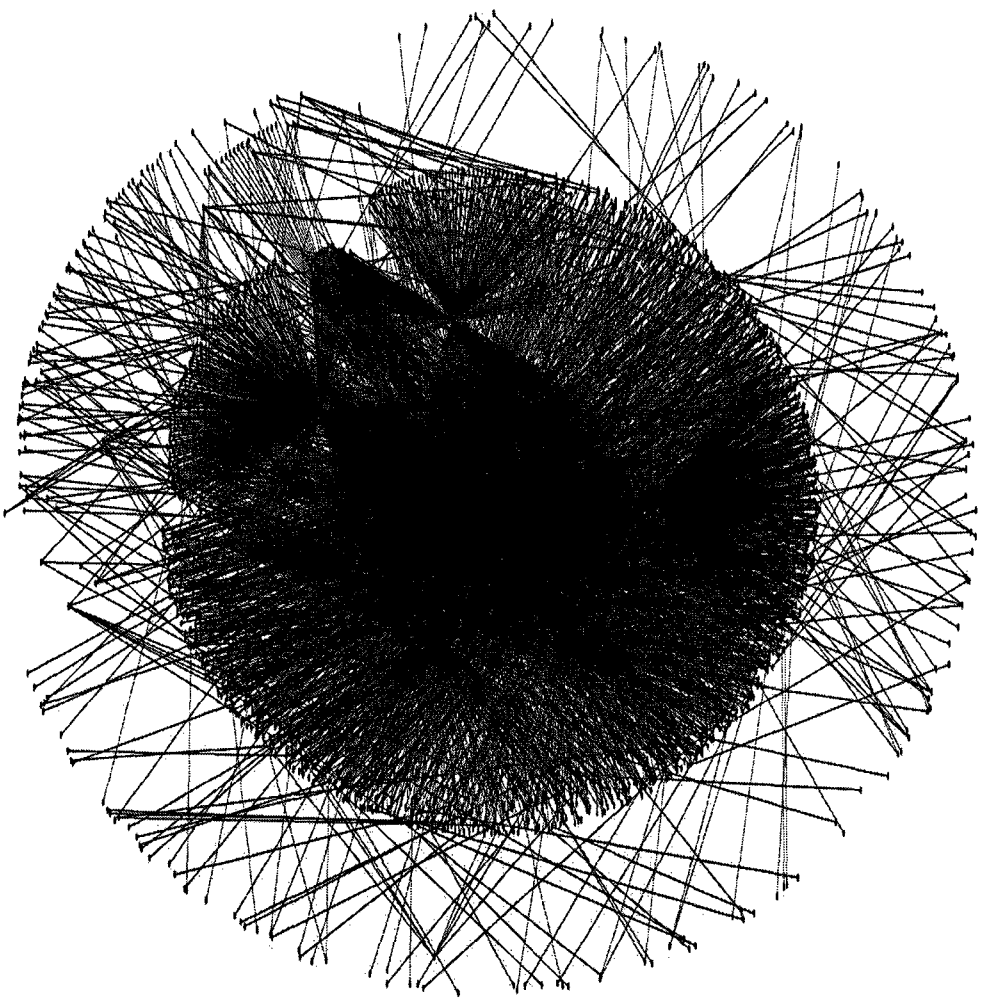

Figure A.1: Structure Graph of Schema eBay. The graph is plotted by GraphViz NEATO

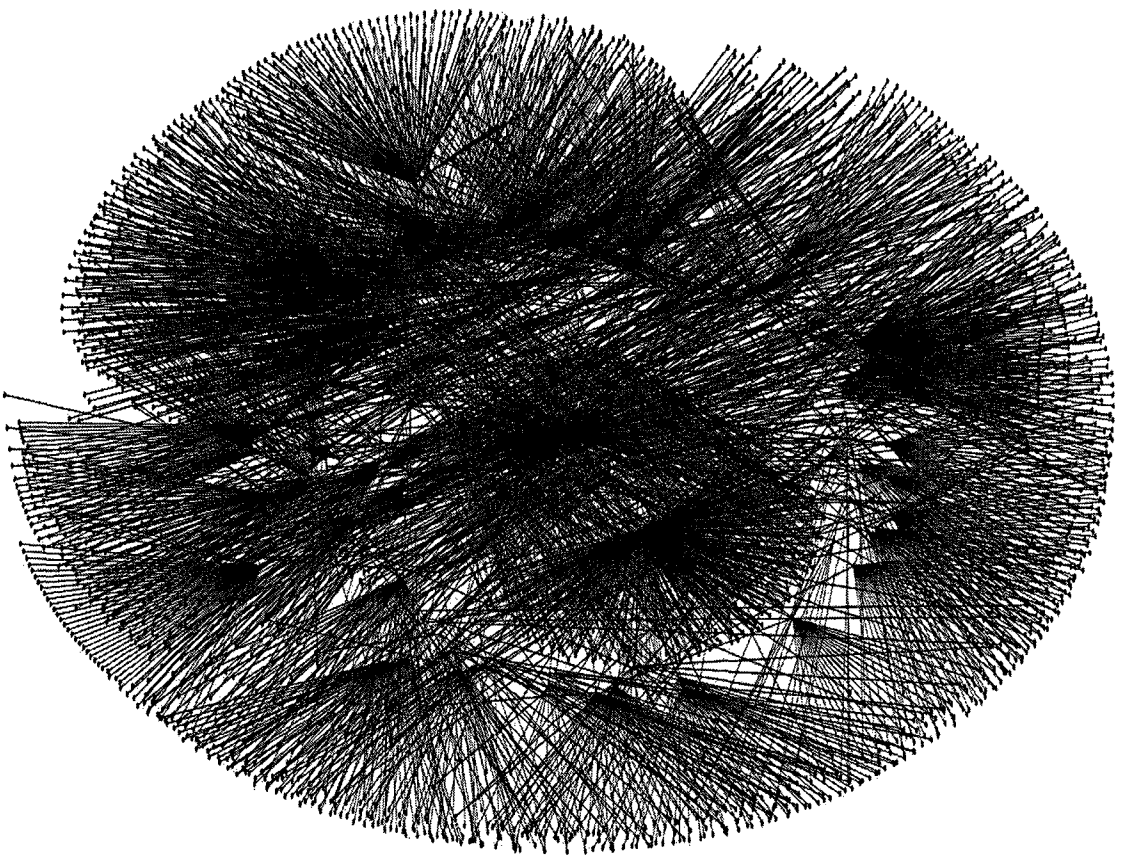

Figure A.2: Structure Graph of Schema PDBML2. The graph is plotted by GraphViz NEATO

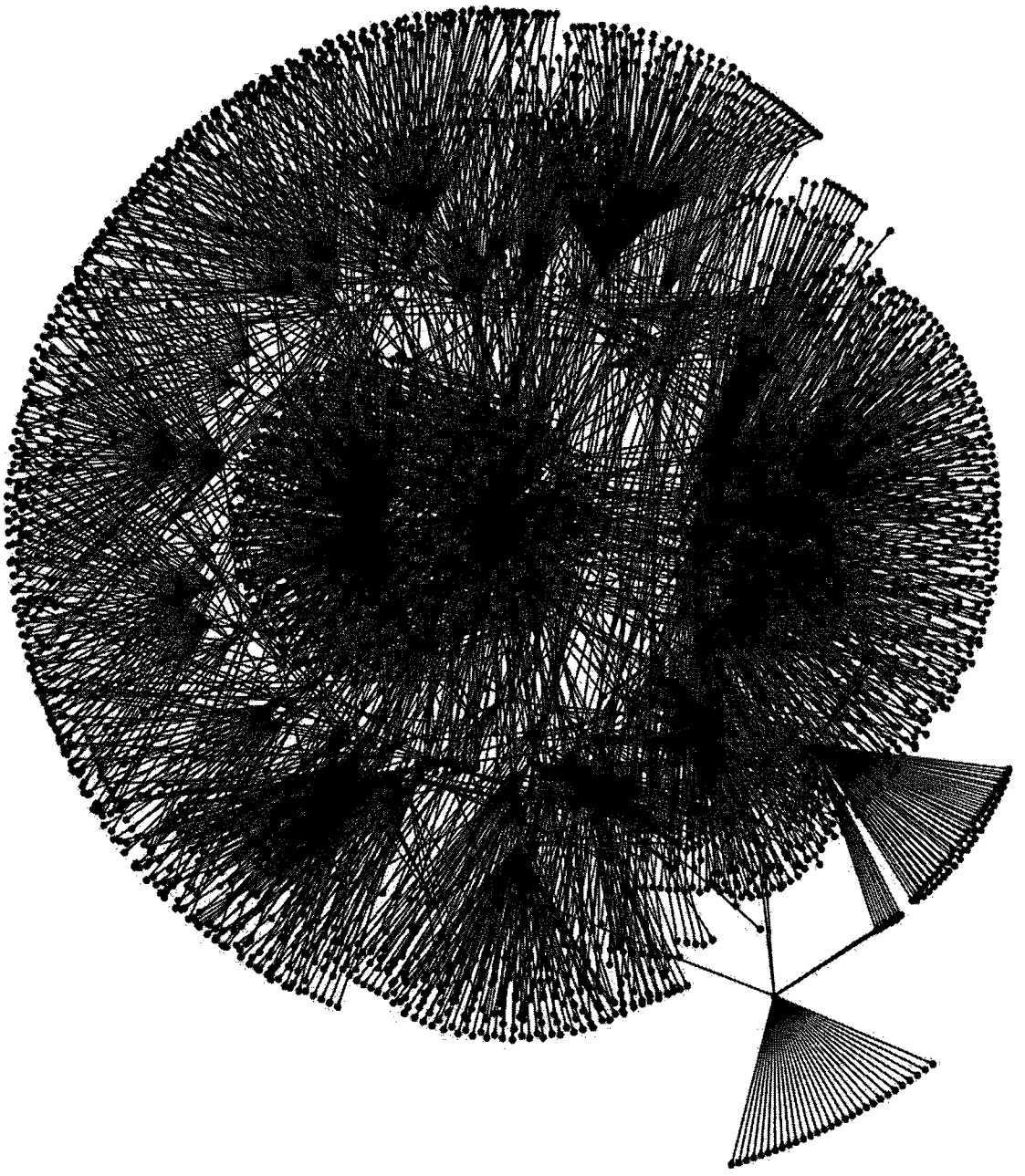

Figure A.3: Structure Graph of Schema PDBML1. The graph is plotted by GraphViz NEATO

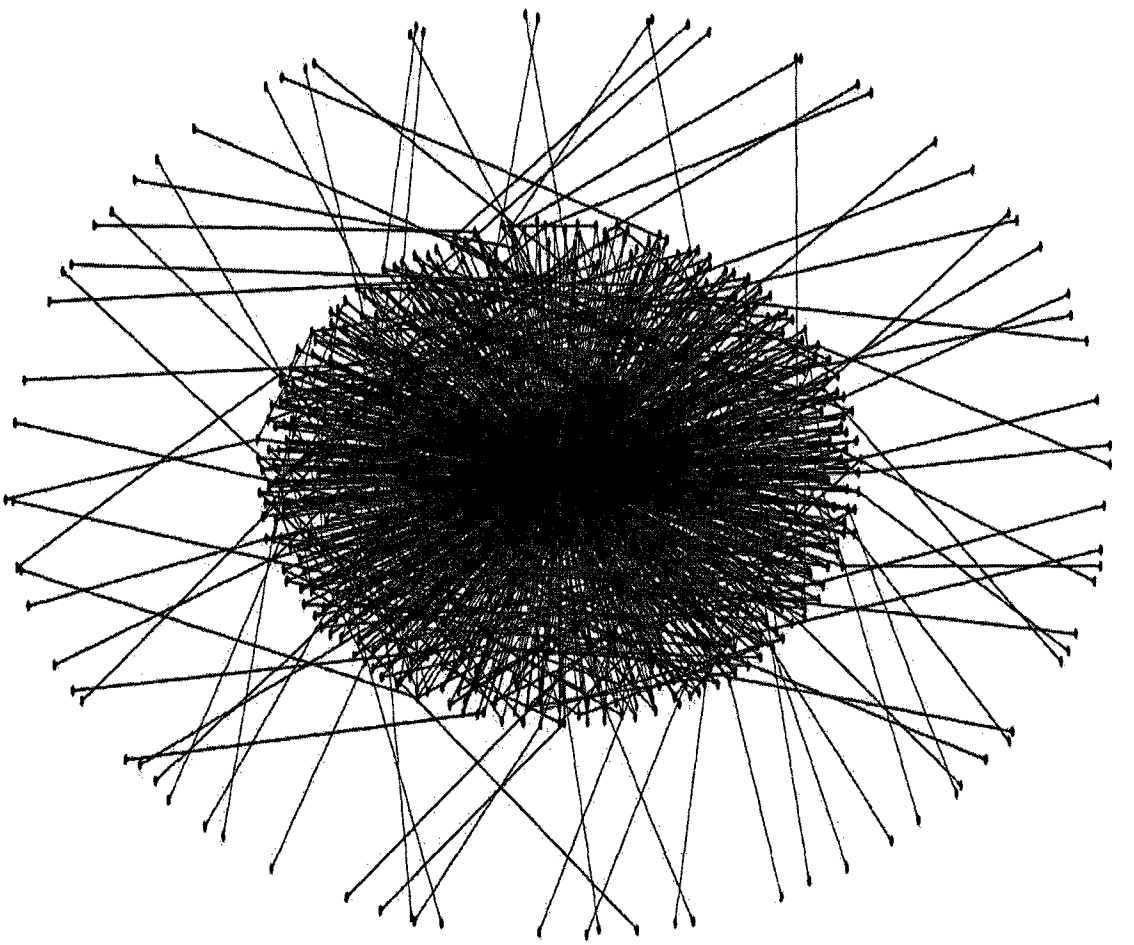

Figure A.4: Structure Graph of Schema Purchase Order. The graph is plotted by GraphViz NEATO

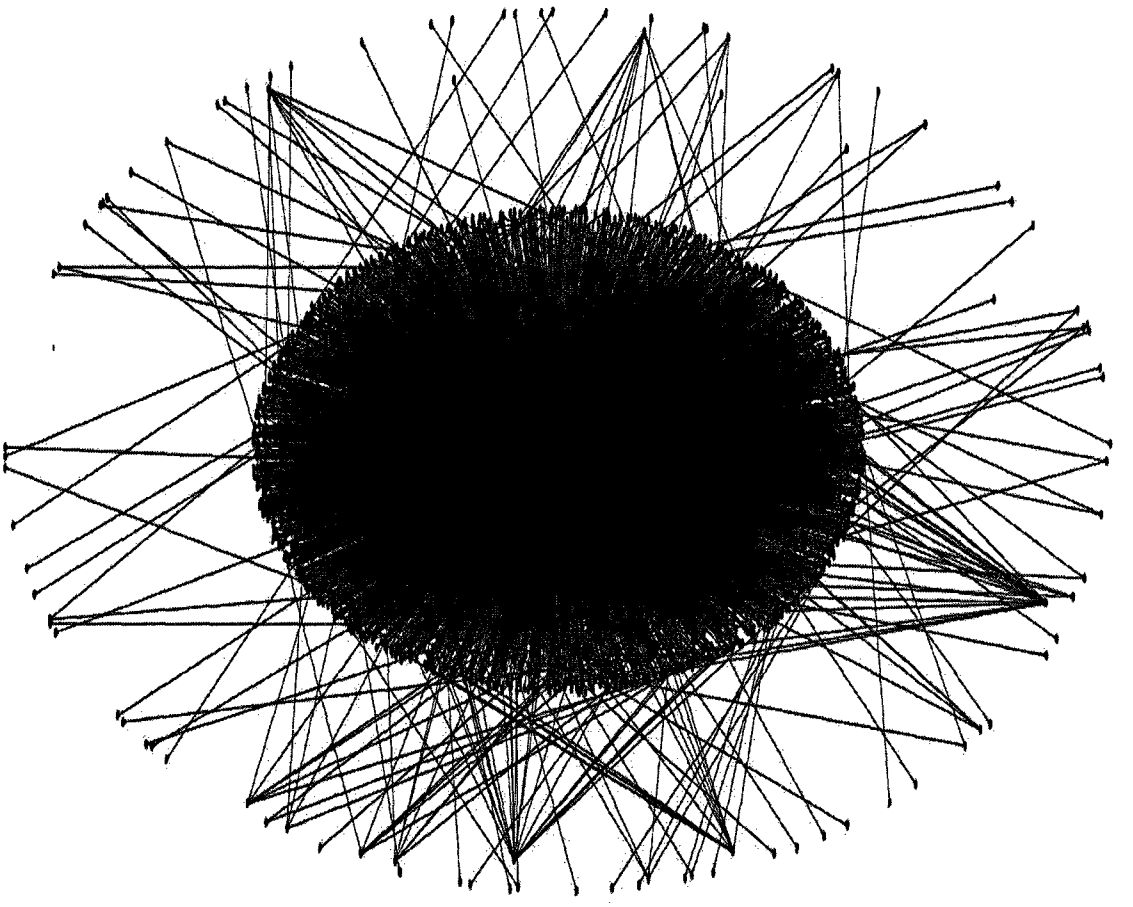

Figure A.5: Structure Graph of Schema NIEM. The graph is plotted by GraphViz NEATO

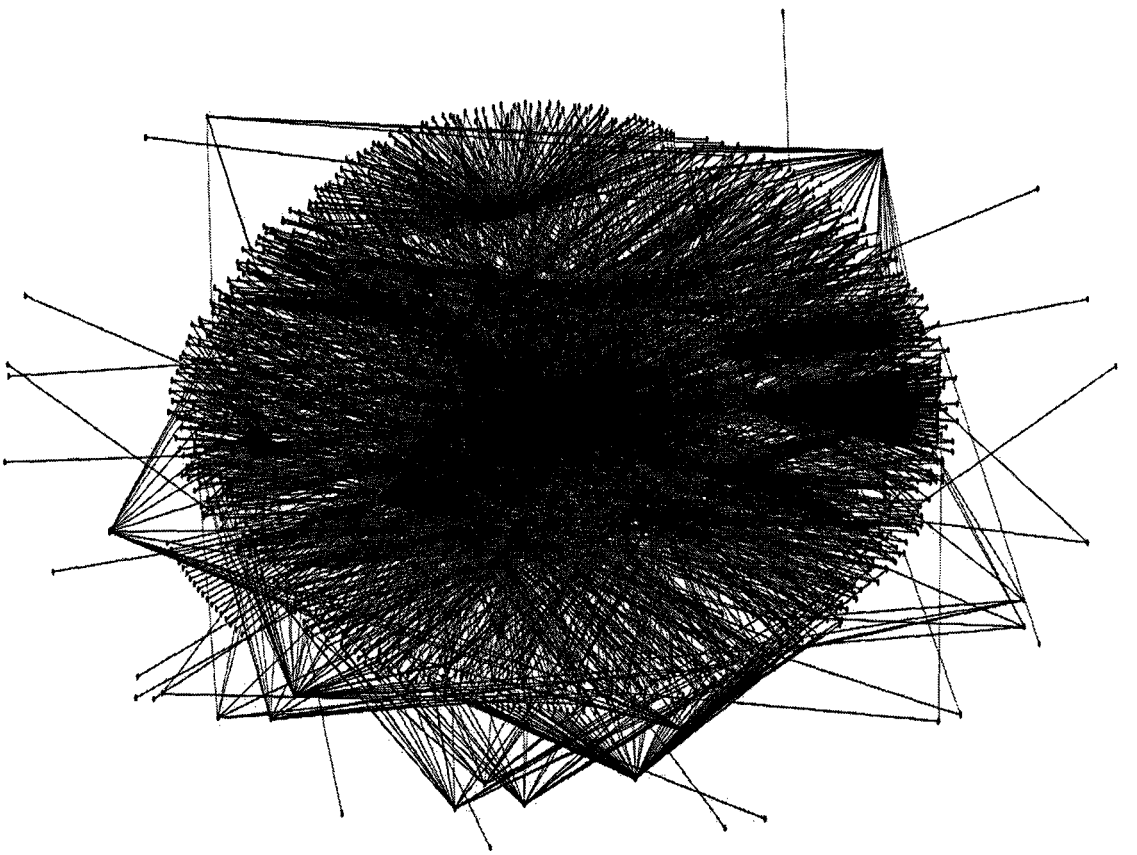

Figure A.6: Structure Graph of Schema UN/CEFACT. The graph is plotted by GraphViz NEATO

Vita Auctoris

Yanyin Zhang was born in 1976 in China. He went to Lanzhou Jiaotong University in Lanzhou, Gansu Province, China, where he obtained the degree in Bachelor of Engineering. He is currently a candidate for the Master's degree in Computer Science at the University of Windsor and will graduate in Summer 2010.